ULAŞIM AĞLARI İÇİN YÖNLENDİRME KONTROLU TASARIMI

Banu ATAŞLAR

Yüksek Lisans Tezi
Elektrik-Elektronik Mühendisliği
Anabilim Dalı
Temmuz 1998

ULAŞIM AĞLARI İÇİN YÖNLENDİRME KONTROLU TASARIMI •

BANU ATAŞLAR

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Yönetmeliği Uyarınca
Elektrik-Elektronik Mühendisliği Anabilim Dalı
Kontrol ve Kumanda Sistemleri Bilim Dalında
YÜKSEK LİSANS TEZİ
Olarak Hazırlanmıştır.

Danışman: Prof. Dr. Altuğ İFTAR

TEMMUZ 1998

Banu ATAŞLAR'ın YÜKSEK LİSANS tezi olarak hazırladığı "Ulaşım Ağları İçin Yönlendirme Kontrolu Tasarımı" başlıklı tez 27/07/1998 tarihinde aşağıdaki jüri tarafından Lisansüstü Öğretim Yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

Üye (Tez Danışmanı) : Prof. Dr. Altuğ İFTAR

Üye : Prof. Dr. Atila BARKANA

Üye : Doç. Dr. Abdurrahman KARAMANCIOĞLU

Anadolu Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 27/08/1998 tarih ve 13/7 sayılı kararıyla onaylanmıştır.

ÖZET

Yüksek Lisans Tezi

ULAŞIM AĞLARI İÇİN YÖNLENDİRME KONTROLU TASARIMI

BANU ATAŞLAR

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Altuğ İFTAR
1998

Bu çalışmada, ulaşım ağlarında ortaya çıkan tıkanıklığı önlemek amacıyla bir dışmerkezli kontrol yaklaşımı önerilmiştir. Akış kontrolu ve yönlendirme kontrolunun birlikte ele alındığı bu yaklaşım daha önce önerilmiş olan tıkanıklık ölçeğine dayalı olarak geliştirilmiştir. Önerilen algoritma her bir nodda yerel olarak uygulanabilmektedir. Bu uygulama sırasında ağdaki diğer nodlarla ilgili olarak gereken tek bilgi akış-aşağı nodlara ait tıkanıklık ölçeği miktarlarıdır. Böylece, önerilen algoritma nodlar arası çok az bilgi alış-verişi gerektirmesi anlamında dışmerkezlidir. Ayrıca önerilen algoritma nodlar arası senkronizasyon gerektirmemektedir. Önerilen yaklaşımın gerçek trafik ağlarında göstereceği performansı önceden saptayabilmek için benzetim çalışmaları da yapılmıştır. Bu amaçla *UNIX* ortamında *C* programlama dilinde bir benzetim programı hazırlanmış ve hazırlanan program kullanılarak iki örnek ağ için benzetim çalışmaları yapılmıştır. Ayrıca, aynı trafik ağlarına aynı tarfik koşulları altında bir alternatif kontrol yaklaşımı da uygulanmış ve elde edilen sonuçlar karşılaştırılmıştır.

Anahtar Kelimeler: Ulaşım ağları; yönlendirme kontrolu; akış kontrolu; dışmerkezli kontrol.

ABSTRACT

Master of Science Thesis

ROUTING CONTROL DESIGN
FOR TRANSPORTATION NETWORKS

BANU ATAŞLAR

Anadolu University
Graduate School of Natural and Applied Sciences
Electrical and Electronics Engineering Program

Supervisor: Prof. Altuğ İFTAR
1998

In this study, a decentralized control approach is proposed for transportation networks to prevent traffic congestion. The approach is developed by considering flow control and routing control together and a control algorithm based on a previously proposed congestion measure is presented. The algorithm can be implemented locally at each node. The only information required from the other nodes is the congestion measure values from the adjacent downstream nodes. Thus, the proposed controller is a decentralized controller which requires very limited information exchange among the nodes. The controller does not require any synchronization among the nodes either (each node may simply use the last received congestion measure values). In order to illustrate the performance of the proposed control approach, a simulation program is developed in the C programming language under the UNIX operating system. By using this program, the algorithm has been simulated for two different networks. Furthermore, an alternative control approach has also been applied to the same networks under the same traffic conditions and the performances of both approaches have been compared.

Keywords: Transportation networks; routing control; flow control; decentralized control.

TEŞEKKÜR

Çalışmalarımda beni yönlendiren ve benden yardımlarını esirgemeyen Hocam Sayın Prof. Dr. Altuğ İFTAR'a teşekkür ederim.

Bu çalışma EEEAG-173 sayılı araştırma projesi çerçevesinde Türkiye Bilimsel ve Teknik Araştırma Kurumu (TÜBİTAK) tarafından desteklenmiştir.

İÇİNDEKİLER

İÇİNDEKİLER (devam)

ŞEKİLLER DİZİNİ

ŞEKİLLER DİZİNİ (devam)

ŞEKİLLER DİZİNİ (devam)

ŞEKİLLER DİZİNİ (devam)

ŞEKİLLER DİZİNİ (devam)

ŞEKİLLER DİZİNİ (devam)

TABLOLAR DİZİNİ

1. GİRİŞ

Ulaşım ağlarında ortaya çıkan en önemli problemlerden biri yönlendirme problemidir. Ulaşım ağlarındaki yönlendirme problemi, araçların ağa katıldıkları noddan ulaşmak istedikleri noda varıncaya kadar hangi nodlar ve linkler üzerinden geçeceğinin belirlenmesi olarak tanımlanabilir. Ancak yönlendirme kontrolu probleminin çözümünün bulunması ulaşım ağlarından en etkin şekilde yararlanılabilmesi için yeterli değildir. Ulaşım ağına katılan araç miktarlarının kontrol edilmediği durumda ağ içinde aşırı oranda tıkanıklıklar ve gecikmeler oluşabilecektir. Bu sebeple, akış kontrolu da ulaşım ağları için büyük önem taşımaktadır.

Yönlendirme probleminin çözümü için günümüze kadar birçok algoritma geliştirilmiştir. [1]'de tek bir giriş ve tek bir varış nodunun yanında bu nodlar arası birden fazla alternatif yolun bulunduğu bir ağ yapısı ele alınarak, bu topolojiye sahip ağlar için trafiğin döngüye girmesini engelleyen dışmerkezli dinamik yönlendirme algoritması geliştirilmiştir. [2]'de yapılan çalışmada, birden fazla giriş ve varış noduna sahip ağlar için kontrol tasarımı önerilmiştir. Bu çalışmada, yönlendirme kontrolu ve akış kontrolu birlikte ele alınmıştır. Ancak önerilen dışmerkezli yönlendirme kontrolu ağa katılan araç miktarının fazla olduğu durumlarda iyi sonuçlar vermeyi garanti edememektedir. Yönlendirme kontrolu algoritması için bir doğrusal olmayan optimizasyon yaklaşımı [3]'de ele alınmıştır. Yapılan çalışmada birden fazla varış noduna sahip ağlar için ağa katılmak üzere nodlara gelen araçların zamana bağlı olarak değiştiği kabul edilmiştir. Ancak, bu algoritma dinamik yönlendirme için aşırı hesaplama yükü gerektirmektedir. Diğer yandan, doğrusal olmayan optimizasyon problemi ağ dinamiğinin basitleştirilmesi ile doğrusal bir optimizasyon problemine dönüşebilmektedir [4]. [5]'de nodlar arası en kısa yol kriteri ele alınarak bunu gerçekleştiren bir kontrol algoritması geliştirilmiştir. [6]'de ise ağa giriş bölümlerindeki akış oranında ortaya çıkan değişimlerin akış aşağı linkleri belli

bir zaman gecikmesi ile etkilediği göz önüne alınarak bir model geliştirilmiş ve bu modele dayalı olarak da ağa katılan araç miktarının maksimize edildiği bir kontrol algoritması önerilmiştir. [7]'de bir optimal kontrol probleminin çözümü ile elde edilen *tıkanıklık ölçeği*'ne dayalı olarak geliştirilen bir dışmerkezli dinamik yönlendirme kontrolu yaklaşımı yer almaktadır. Bu kontrolör dışmerkezli olması ve kolayca gerçeklenebilmesi yanında tüm yönlendirme kısıtlarını (sıra uzunluklarının ve akış oranlarının negatif olmaması ve akış oranlarının link kapasitelerini aşmaması) sağlamakta, trafiğin döngüye girmesini önlemekte ve ağa giriş oranlarının belli limitleri aşmaması durumunda tüm sıra uzunluklarını sonlu zamanda sıfırlayarak tüm araçları sonlu zamanda varış noktalarına ulaştırmaktadır. Ancak, önerilen kontrolör pratik açıdan önemli iki noktayı göz ardı etmektedir: (i) Kontrolörün uygulaması sürekli değişkenler kullanılarak sürekli zamanda yapılmıştır; bir başka deyişle, sistemin kesikli doğası dikkate alınmamıştır. (ii) Her bir nodda belli varış noktalı araçlar sıralarda bekletilirken başka varış noktalı araçların serbestçe yönlendirilebileceği kabul edilmiştir; dolayısıyla, ulaşım ağlarında sıra başında bekleyen bir aracın yönlendirilmesi yapılmadan o aracın gerisinde bekleyen araçların yönlendirilemeyeceği kısıtlaması göz ardı edilmiştir. Bu olumsuzluklardan arındırılarak geliştirilen [8]'daki algoritma nodlarda kuyrukta sıra bekleyen araçları sırasıyla ele alarak, her bir kontrol periyodunda tam sayıda araç yönlendirilmesini sağlamaktadır. Bu kontrolde her bir nodun akış aşağı nodlarındaki tıkanıklık ölçeği değerlerini bilmesi gerçek zaman içi hesaplamalar için yeterlidir. Bunun dışında diğer nodlarla hiç bir bilgi alış-verişine gerek duyulmadan gerçek zaman içi hesaplamalar yerel olarak nodlarda yapılabilmektedir.

Diğer yandan, akış kontrolu probleminin çözümü üzerine de birçok çalışma yapılmıştır. [9]'de yeni bir metod olarak *arttırma yöntemi* (augmentation method) kullanılarak bir kontrol yaklaşımı geliştirilmiştir. Bu metod orjinal sistemin dinamiğinden elde edilen birbirleriyle örtüşen altsistemlerin optimal kontroluna dayalı olarak elde edilen bir alt-optimal kontrol içerir. Bir başka akış kontrolu da [10]'da farklı bir yöntem olarak *ardarda sıralama tekniği* (cascading technique)

kullanılarak geliştirilmiştir. Yapılan çalışmada ardarda sıralama tekniğinin kullanılması ile geliştirilen kontrolun arttırma tekniği ile geliştirilen kontrole göre daha verimli olduğu gösterilmiştir. [11]'da doğrusal kuadratik (Linear-Quadratic - LQ) optimizasyon yaklaşımına dayalı olarak bir kontrol stratejisi geliştirilmiştir. LQ metodolojisinin bu uygulaması [9, 10]'de ele alınan klasik LQ-yaklaşımından farklıdır. [11]'da klasik LQ-yaklaşımı ile karşılaştırıldığında bazı avantajlara sahip olan integral kısımlarına sahip çok değişkenli bir regülatör kullanılmıştır.

Trafik ağları üzerine yapılan çalışmalarda, yönlerdirme ve/veya akış kontrolu üzerine geliştirilen kontrollerin gerçek trafik ağlarında göstereceği performansı önceden belirleyebilmek amacıyla benzetim çalışmaları da yapılmaktadır. Benzetim çalışmalarında kullanılan ağ modelinin gerçeğe yakınlılığı sonradan doğabilecek sorunları önlemek açısından çok önemlidir. Bu konuda yapılan belli başlı çalışmalarda ([3] ve [12]–[16]), ağlardaki trafik akışı ideal bir akışkana benzetilerek modellemesi yapılmaktadır. [16]'de her birinde iki temel kavramın (korunum denklemi ve ortalama hız-yoğunluk karakteristiği) temel alındığı üç farklı trafik modeli incelenmiştir. Bu çalışmadaki bir model ele alınarak trafik ağları için bir makroskobik benzetim programı da hazırlanmıştır [17].

Bu çalışmada ise, [8]'de önerilen algoritma temel alınarak, yönlendirme kontrolu yanında akış kontrolunu da birlikte ele alan yeni bir kontrol algoritması önerilmiştir. Önerilen algoritma her bir nodda yerel olarak uygulanabilmektedir. Bu uygulama sırasında ağdaki diğer nodlarla ilgili olarak gereken tek bilgi akışaşağı nodlara ait tıkanıklık ölçeği miktarıdır. Böylece, önerilen algoritma nodlar arası çok az bilgi alış-verişi gerektirmesi anlamında dışmerkezlidir. Ayrıca önerilen algoritma nodlar arası senkronizasyon gerektirmemektedir.

Ulaşım ağlarında ortaya çıkan yönlendirme ve akış kontrolu problemlerinin çözümü için hazırlanan bu çalışmanın ikinci bölümünde, gerek kontrolör tasarımı aşamasında, gerekse benzetim aşamasında kullanılan trafik akış dinamiği modeli tanıtılacaktır. Üçüncü bölümde ise geliştirilen kontrol algoritmasına yer verilecektir. Bu kontrol algoritmasının trafik ağlarında göstereceği performansı önceden

saptayabilmek için yapılan benzetim çalışmaları ile ilgili olarak, öncelikle dördüncü bölümde UNIX ortamında C programlama dilinde hazırlanmış olan benzetim programı tanıtılacak; ardından beşinci bölümde bu programın kullanılmasıyla gerçekleştirilen benzetim çalışmalarına yer verilecektir.

2. TRAFİK AKIŞ DİNAMİĞİ MODELİ

Ulaşım ağları üzerine yapılan çalışmalarda gerek kontrolör tasarımı aşamasında, gerekse benzetim aşamasında öncelikle trafik akış dinamiğinin modellenmesi gerekmektedir. Ulaşım ağlarındaki yönlendirme ve akış kontrolu problemlerini çözmek amacıyla yapılan bu çalışmada ele alınan trafik akış dinamiği modeli [17]'de kullanılan model temel alınarak hazırlanmıştır.

Ele alınan bu modelde, kavşaklar (ağ içindeki iki veya daha fazla yolun kesişim noktası veya ağa trafik girişinin ve/veya çıkışının olabileceği nokta) nodları, kavşaklar arasındaki yollar linkleri oluşturmaktadır. Nod ve linklerden oluşan ağ modeline ait trafik akış dinamiğinin belirlenebilmesi için trafiğin akışı ideal bir akışkana benzetilerek aşağıdaki değişkenler tanımlanmıştır [16]:

- *yoğunluk* $\rho(x,t)$: x konumunda t zamanındaki birim uzunluk başına düşen araç sayısı (*araç/km*)
- *ortalama hız* $v(x,t)$: x konumunda t zamanındaki araçların ortalama hızı (*km/saat*)
- *trafik akış oranı* $q(x,t)$: x konumunda t zamanındaki birim sürede geçen araç sayısı (*araç/saat*).

Trafiğin ideal bir akışkana benzetilmesiyle, linklerdeki akış oranı hidro-dinamik sistemlerdeki yapıya dayalı olarak

$$q(x,t) = \rho(x,t)\, v(x,t) \tag{2.1}$$

denklemi ile ifade edilmektedir [16]. Yine hidro-dinamik sistemlere olan benzerlikten yararlanarak trafiğin linkler üzerindeki akışı için aşağıdaki korunum denklemi yazılmaktadır [16]:

$$\frac{\partial \rho(x,t)}{\partial t} + \frac{\partial q(x,t)}{\partial x} = 0 \tag{2.2}$$

Ele alınan bu modelde, linklerdeki $x + \Delta x$ konumundaki t zamanına ait yoğunluk değerinin x konumundaki ortalama hız değerini τ zaman gecikmesi ile etkileyeceği kabul edilerek ortalama hız ile yoğunluk ifadeleri arasındaki ilişki

$$v(x, t+\tau) = V(\rho(x+\Delta x, t)) \tag{2.3}$$

şeklinde ifade edilmektedir [16]. Bu eşitliğin sol tarafı τ etrafında, sağ tarafı ise Δx etrafında Taylor serisine açıldığında ve yüksek dereceli terimler ihmal edildiğinde

$$v(x,t) + \frac{dv(x,t)}{dt}\,\tau = V(\rho(x,t)) + \left(\frac{\partial V(\rho(x,t))}{\partial \rho(x,t)}\,\frac{\partial \rho(x,t)}{\partial x}\right)\Delta x \tag{2.4}$$

ifadesi elde edilir. Burada $\Delta x \triangleq \frac{0.5}{\rho(x,t)}$ ve $\nu \triangleq -0.5\,\frac{\partial V(\rho(x,t))}{\partial \rho(x,t)} > 0$ tanımlamaları yapıldığında [16] yukarıdaki denklem,

$$v(x,t) + \frac{dv(x,t)}{dt}\,\tau = V(\rho(x,t)) - \frac{\nu}{\rho(x,t)}\,\frac{\partial \rho(x,t)}{\partial x} \tag{2.5}$$

şeklini almaktadır. Ayrıca, trafiğin akışı ile beraber hareket eden bir gözlemleyiciye ait ivme ifadesi

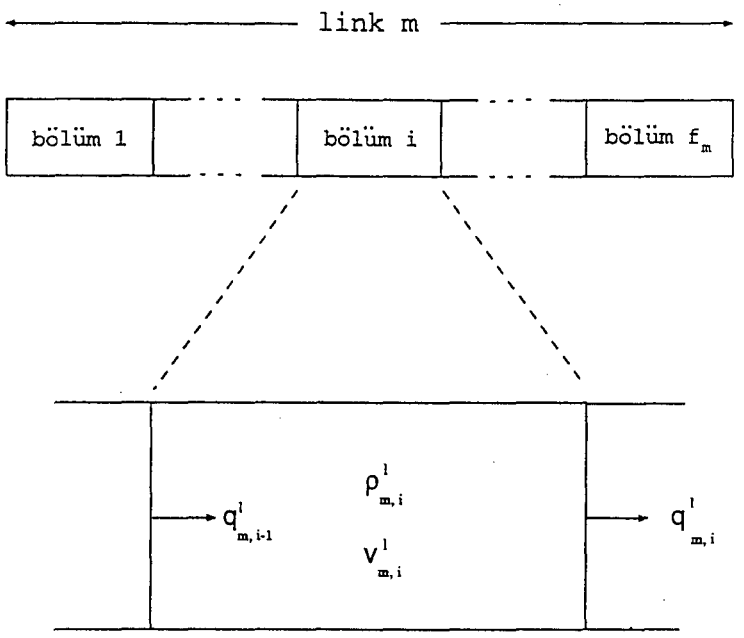
$$\frac{dv(x,t)}{dt} = v(x,t)\,\frac{\partial v(x,t)}{\partial x} + \frac{\partial v(x,t)}{\partial t} \tag{2.6}$$

şeklindedir. (2.6) denkleminin (2.5)'de yerine konulmasıyla ortalama hız değişkenine ait dinamik denklem

$$\frac{\partial v(x,t)}{\partial t} = \frac{1}{\tau}\left(V(\rho(x,t)) - v(x,t) - \frac{\nu}{\rho(x,t)}\,\frac{\partial \rho(x,t)}{\partial x}\right) - v(x,t)\,\frac{\partial v(x,t)}{\partial x} \tag{2.7}$$

olarak elde edilir.

Böylece elde edilen (2.1), (2.2) ve (2.7) denklemleri bir ağdaki trafik akış dinamiği modelini ifade etmektedir. Sürekli zaman ve konumda elde edilmiş olan



Şekil 2.1. Linklerin modellenmesi

bu model gerek kontrolör tasarımı ve gerekse benzetim çalışmaları sırasında kolaylık sağlaması açısından kesikli zaman ve konuma aktarılmıştır. Bu amaçla, her link eşit uzunlukta (genelde 300m. mertebesinde) bölümlere ayrılmış (Şekil 2.1) ve zamanın da belirli bir periyodla (genelde 10s. mertebesinde) örneklenmesi ile ağ modeli kesikleştirilmiştir. Bu modele ait dinamik denklemler de (2.1), (2.2) ve (2.7) denklemlerinin kesikleştirilmesi ile elde edilmiştir. Bu amaçla, yoğunluk, akış oranı ve ortalama hız değişkenlerinin kesikli konum ve zamandaki yeni tanımları:

$\rho^l_{m,i}(k)$: m linkinin i'inci bölümündeki ($i = 1, 2, ..., f_m$; f_m: m linkinin bölüm sayısı) varış nodu l olan araçların kT zamanındaki yoğunluğu (her bir şeritteki birim uzunluk başına araç sayısı) (*araç/km/şerit*).

$q^l_{m,i}(k)$: m linkinin i'inci bölümündeki varış nodu l olan araçların kT zamanındaki akış oranı (birim zamanda geçen araç sayısı) (*araç/saat*).

$v_{m,i}(k)$: m linkinin i'inci bölümünde kT zamanındaki ortalama hız değeri (*km/saat*).

şeklinde yapılmış ve yeni dinamik denklemler aşağıdaki şekilde elde edilmiştir:

$$q^l_{m,i}(k) = \lambda_m \, \rho^l_{m,i}(k) \, v_{m,i}(k) \tag{2.8}$$

$$\rho^l_{m,i}(k+1) = \rho^l_{m,i}(k) + \frac{T}{L_m \, \lambda_m} \left(q^l_{m,i-1}(k) - q^l_{m,i}(k) \right) \tag{2.9}$$

$$\rho_{m,i}(k) = \sum_{l \in J_m} \rho^l_{m,i}(k) \tag{2.10}$$

$$\begin{aligned} v_{m,i}(k+1) \;=\; & v_{m,i}(k) + \frac{T}{\tau} \left(V(\rho_{m,i}(k)) - v_{m,i}(k) \right) \\ & + \frac{T}{L_m} v_{m,i}(k) \left(v_{m,i-1}(k) - v_{m,i}(k) \right) \\ & - \frac{\nu \, T}{\tau \, L_m} \frac{\rho_{m,i+1}(k) - \rho_{m,i}(k)}{\rho_{m,i}(k) + \kappa} \end{aligned} \tag{2.11}$$

Bu denklemlerdeki değişkenler:

T: örnekleme periyodu

L_m: m linkine ait her bir bölümün uzunluğu

λ_m: m linkinin sahip olduğu şerit sayısı

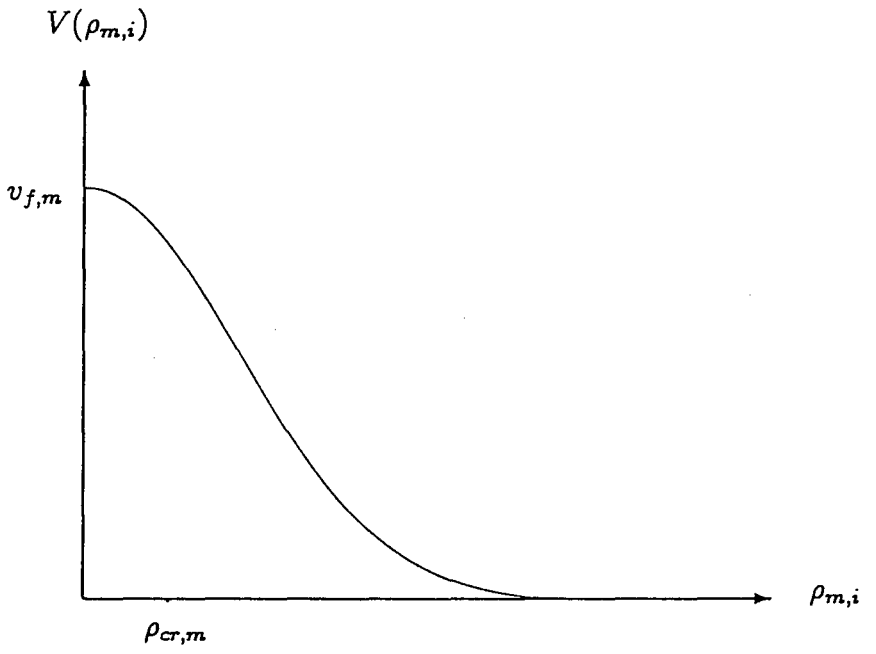
J_m: m linki ile ulaşılabilecek varış nodlarının kümesi.

şeklindedir.

Denklem (2.11) ile verilen ortalama hızın değişimini gösteren ifade üç terimden oluşmaktadır:

- *Birinci Terim:* $\left(\frac{T}{\tau} \left(V(\rho_{m,i}(k)) - v_{m,i}(k) \right) \right)$ Bu terim ortalama hız değerini asimtotik olarak, yatışkın durum karakteristiği olan $V(\rho_{m,i}(k))$ fonksiyonuna yakınlaştırır. Yatışkın durum karakteristiği $V(\rho_{m,i}(k))$ yoğunluk değeri arttıkça araçların hızının azalması mantığına dayalıdır. Bunun matematiksel ifadesi için literatürde yapılmış olan birden fazla çalışma vardır (bkz. [16] ve oradaki referanslar). Ele alınan bu modelleme çalışmasında kullanılan $V(\rho_{m,i}(k))$ ise

$$V(\rho_{m,i}(k)) = v_{f,m} \exp \left[-\frac{1}{a_m} \left(\frac{\rho_{m,i}(k)}{\rho_{cr,m}} \right)^{a_m} \right] \tag{2.12}$$



Şekil 2.2. $V(\rho_{m,i})$ fonksiyonunun grafiği

ifadesiyle tanımlanan monoton olarak azalan bir fonksiyondur (Şekil 2.2). Bu fonksiyondaki değişkenler:

$\rho_{cr,m}$: m linkine ait *kritik trafik yoğunluğu*

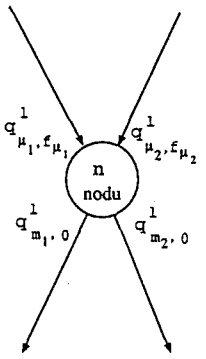
$v_{f,m}$: m linkine ait *serbest hız değeri*

Ayrıca, buradaki a_m her bir m linki için o linke ait C_m (m linkinin kapasite değeri (*araç/saat/şerit*)), $v_{f,m}$ ve $\rho_{cr,m}$ değerleri kullanılarak aşağıdaki denklem yardımı ile hesaplanan sabit değerli bir parametredir.

$$a_m = \left(\ln\left(\frac{v_{f,m} \times \rho_{cr,m}}{C_m}\right)\right)^{-1} \tag{2.13}$$

- *İkinci Terim:* $\left(\frac{T}{L_m} v_{m,i}(k) \left(v_{m,i-1}(k) - v_{m,i}(k)\right)\right)$ Linkteki bir önceki bölümden o bölüme olan hız yayılımını belirler.

- *Üçüncü Terim:* $\left(\frac{\nu T}{\tau L_m} \frac{\rho_{m,i+1}(k) - \rho_{m,i}(k)}{\rho_{m,i}(k) + \kappa}\right)$ Sürücünün göreceli yoğunluk değişimine olan tepkisini ν parametresi ağırlığında yansıtan terimdir. Denklem



Şekil 2.3. Akış oranlarının nodlardaki paylaşımı

(2.7)'nin kesikleştirilmesiyle (2.11)'un elde edilmesi sırasında üçüncü terime eklenen κ parametresi yoğunluğun çok küçük olduğu durumda paydanın sıfıra yaklaşarak bu terimin etkisinin gerçekte olması gerektiğinden çok daha büyük olmasını önler.

Trafik akış dinamiği modelini ifade eden (2.8)-(2.13) denklemlerinin kullanılabilmesi için her bir linkin başlangıcındaki akış oranlarının ($q^l_{m,0}(k)$), her bir linkin başlangıcındaki ortalama hız değerinin ($v_{m,0}(k)$) ve her bir linkin sonundaki yoğunluğun ($\rho_{m,f_m+1}(k)$) bilinmesi gerekmektedir. Bu değerler verilen dinamik denklemlerin sınır koşullarını oluşturmaktadır. Sınır koşullarından biri olan $q^l_{m,0}(k)$ (her bir linkin başlangıcındaki akış oranı) akış-yukarı linklerden nodlara gelen trafiğin o nodda akış-aşağı linklere paylaşımının yapılması sonucunda elde edilir (Şekil 2.3). Akış-yukarı linklerden ve/veya ağa katılmak üzere dışarıdan n noduna gelen araçların varış noduna ulaşmasını sağlayacak birden fazla akış-aşağı link varsa, bu araçlar söz konusu linkler arasında uygulanan yönlendirme kontrolörü tarafından belirlenen oranlarda ($\phi^l_{n,m}(k)$) paylaştırılır:

$$q^l_{m,0}(k) \;=\; \phi^l_{n,m}(k)\left(\hat{r}^l_n(k) + \sum_{\mu\in I_n} q^l_{\mu,f_\mu}(k)\right) \tag{2.14}$$

Burada, $\hat{r}^l_n(k)$, varış nodu l olan araçların kT zamanında n nodundan trafiğe katılma oranıdır (birim zamanda trafiğe katılan araç sayısı - bu değer akış kontrolu ile belirlenmektedir). $\phi^l_{n,m}(k)$ ise l varış noduna ulaşmak üzere kT zamanında n

noduna gelen araçlar için akış aşağı linklerden biri olan m linkine dağılım oranıdır ve bu değer yönlendirme kontrolu ile belirlenmektedir. Ayrıca, I_n n noduna giren linklere ait kümeyi göstermektedir.

Sınır koşullarından biri olan her bir linkin başlangıcındaki ortalama hız değeri ($v_{m,0}(k)$), o linkin akış yukarısındaki noda katılan linklerin son bölümlerindeki ortalama hız değerleri üzerinden alınan bir ağırlıklı ortalama ile hesaplanır:

$$v_{m,0}(k) = \frac{v_n(k)\hat{r}_n(k) + \sum_{\mu \in I_n} v_{\mu,f_\mu}(k) q_{\mu,f_\mu}(k)}{\hat{r}_n(k) + \sum_{\mu \in I_n} q_{\mu,f_\mu}(k)} \tag{2.15}$$

Eğer noda ağ dışından katılan bir link (on-ramp) varsa bu linkin son bölümündeki ortalama hız değeri ($v_n(k)$: kT zamanında n nodundan trafiğe katılan araçların trafiğe katıldıkları andaki ortalama hızları) [17]'deki yaklaşımdan farklı olarak aşağıdaki şekilde hesaplanır:

$$v_n(k) = min\left(v_n^M , \frac{\sum_{\mu \in O_n} v_{\mu,1}(k)}{s(O_n)} \right) \tag{2.16}$$

Burada, v_n^M, n nodunun giriş bölümü (on-ramp) için tanımlanmış olan en yüksek hız değeridir. Ayrıca, O_n, n nodundan ayrılan linklerin kümesi olmak üzere, $s(O_n)$ bu kümenin eleman sayısını ifade etmektedir.

Üçüncü sınır koşulu olan her bir linkin sonundaki yoğunluk değeri ($\rho_{m,f_m+1}(k)$), o linkin akış aşağısındaki noddan ayrılan linklerin ilk bölümlerindeki yoğunluk değerleri üzerinden alınan bir ağırlıklı ortalama ile hesaplanır:

$$\rho_{m,f_m+1}(k) = \frac{(\rho_n(k))^2 + \sum_{\mu \in O_n} (\rho_{\mu,1}(k))^2}{\rho_n(k) + \sum_{\mu \in O_n} \rho_{\mu,1}(k)} \tag{2.17}$$

Eğer noddan ayrılan linklerden biri ağ dışına çıkıyorsa (off-ramp) bu linkin ilk bölümündeki yoğunluk değeri ($\rho_n(k)$: kT zamanında n nodunda trafikten ayrılan araçların çıkış bölümündeki yoğunluğu) [17]'deki yaklaşımdan farklı olarak aşağıdaki şekilde hesaplanır:

$$\rho_n(k) = \frac{\sum_{\mu \in I_n} q^n_{\mu,f_\mu}(k)}{\lambda^o_n v^o_n} \tag{2.18}$$

Burada, v^o_n, n nodunun çıkış bölümü için tanımlanmış olan ortalama hız değerini, λ^o_n ise n nodunun çıkış bölümünün şerit sayısını göstermektedir.

$v_n(k)$ ve $\rho_n(k)$ değişkenlerinin değerlerinin [17]'deki yaklaşımdan farklı olarak seçilmesinin nedeni, [17]'de ağdaki nodlara ait giriş-çıkış linklerinin toplam sayısının en fazla üç olduğu (en fazla 1 giriş - 2 çıkış veya 2 giriş - 1 çıkış) kabul edilerek bir yaklaşım hazırlanmış olmasıdır.

Yukarıda temel denklemleri verilen modellemeye ek olarak [17]'de kullanılan yaklaşımda, ağda trafik akışı sırasında doğabilecek üç durum göz önüne alınarak bu durumlar için verilen denklemlerde iyileştirmeler önerilmiştir:

- Trafik ağında akış-aşağı yolların şerit sayısında azalma olduğu durumda, kavşaklara yaklaşan araçların hızlarını azaltmaları gerekmektedir. Bu azaltma oranı, linklerin şerit sayılarının yanısıra, linklerin o anki yoğunluk durumuna ve araçların hızlarına bağlı olarak değişir.

  n noduna katılan her bir m linki için n nodundan ayrılan linklerin şerit sayıları toplamı $(\lambda_{\mu_1} + \lambda_{\mu_2})$ m linkinin şerit sayısından (λ_m) daha küçük ise, $v_{m,f_m}(k+1)$ değeri hesaplanırken (2.11) denkleminin sağ tarafına ek olarak aşağıdaki terim eklenmektedir (ϕ sabit değerli bir parametredir):

  $$-\frac{\phi\, T}{L_m\, \lambda_m} \frac{(\lambda_m - (\lambda_{\mu_1} + \lambda_{\mu_2}))\, \rho_{m,f_m}(k)}{\rho_{cr,m}} (v_{m,f_m})^2$$

  [17]'de önerilen bu yaklaşımda çıkış linklerinden biri (μ_2) ağdan ayrılan bir link (off-ramp) olduğunda, bu linkin şerit sayısı dikkate alınmamaktadır. n noduna gelen araçlardan belli bir bölümünün μ_2 linkinden ayrılacağı göz önüne alınarak bu çalışma kapsamında ele alınan modelleme için bu durumda eklenecek olan terim aşağıdaki gibi değiştirilmiştir:

  $$-\frac{\phi\, T}{L_m\, \lambda_m} \frac{(\lambda_m - \lambda_{\mu_1})\, (\rho_{m,f_m}(k) - \rho^n_{m,f_m})}{\rho_{cr,m}} (v_{m,f_m})^2$$

- Trafik ağında ana yol ile farklı yönden gelen birer yol bir kavşakta birleşiyorsa ve akış-aşağı yolun şerit sayısı bu yolların şerit sayıları toplamından daha küçük ise, bu durumda kavşakta ve devamı olarak yolun ilk bölümünde araçlar hızlarını azaltmak zorunda kalırlar.

  Biri (μ_1) ana yoldan, diğeri (μ_2) ise farklı yönden gelerek bu noda katılan linkler olmak üzere bir n noduna ait iki giriş linki varsa ve n nodunun çıkış linklerinden biri olan m linkinin şerit sayısı (λ_m) giriş linklerinin şerit sayıları toplamından ($\lambda_{\mu_1} + \lambda_{\mu_2}$) daha küçük ise, $v_{m,1}(k+1)$ değeri hesaplanırken (2.11) denkleminin sağ tarafına ek olarak aşağıdaki terim eklenmektedir:

$$-\frac{\delta\, T}{L_m\, \lambda_m}\, \frac{q_\mu(k)\, v_{m,1}(k)}{\rho_{m,1}(k) + \kappa}$$

  Burada δ sabit değerli bir parametre olup, $q_\mu(k)$'nın değeri ise aşağıdaki şekilde hesaplanmaktadır:

$$q_\mu(k) = \begin{cases} \sum\limits_{l \in J_{\mu_2}} q^l_{\mu_2, f_{\mu_2}}(k)\,, & \lambda_m \leq \lambda_{\mu_1} \\ \\ \sum\limits_{l \in J_{\mu_2}} q^l_{\mu_2, f_{\mu_2}}(k) - C_m\,(\lambda_m - \lambda_{\mu_1})\,, & \lambda_m > \lambda_{\mu_1} \end{cases}$$

- Yolun belli bir bölümünde yoğunluk değeri, yolun taşıyabileceği bir maksimum yoğunluk miktarını aşarsa, trafik akışı durabilmektedir.

  m linkinin $i+1$'inci bölümüne ait yoğunluk değeri ağdaki yolların durumuna göre belirlenmiş olan bir ρ_{max} değerine ulaşmış veya belli bir aralıkta yaklaşmış ise, $q^l_{m,i}(k)$ değişkeninin değerinde o anda sahip olduğu değer üzerinden aşağıda verilen yaklaşım doğrultusunda hesaplanan oranda bir azaltma yapılmaktadır. Ayrıca, $\forall l \in J_m$ olmak üzere $q^l_{m,i}(k)$ değişkenlerinin yeni hesaplanmış olan değerleri kullanılarak $v_{m,i}(k)$ değişkeninin o anki değeri de değiştirilmektedir (Burada R ağdaki yolların durumuna göre belirlenmiş sabit

bir değerdir) :

$$q^l_{m,i}(k) = \begin{cases} q^l_{m,i}(k)\ \left(1 - \frac{\rho_{m,i+1}(k)-(\rho_{max}-R)}{R}\right), \\ \qquad\qquad (\rho_{max} - R) < \rho_{m,i+1}(k) < \rho_{max} \\ 0\,, \qquad\qquad \rho_{m,i+1}(k) \geq \rho_{max} \end{cases}$$

$$v_{m,i}(k) = \begin{cases} \dfrac{\sum\limits_{l \in J_m} q^l_{m,i}(k)}{\rho_{m,i}(k)\ \lambda_m}, \quad (\rho_{max} - R) < \rho_{m,i+1} < \rho_{max} \\ 0\,, \qquad \rho_{m,i+1} \geq \rho_{max} \end{cases}$$

3. ÖNERİLEN KONTROL ALGORİTMASI

[8]'da yapılan çalışmada, [7]'de geliştirilen algoritma temel alınmak üzere bu algoritmanın birinci bölümde bahsedilen olumsuzlukları ortadan kaldırılarak yeni bir kontrol algoritması önerilmiştir. Önerilen algoritma bir optimal kontrol probleminin çözümünden elde edilen ve yine ilk kez [7]'de kullanılmış olan "tıkanıklık ölçeği" yaklaşımına dayalı olarak geliştirilmiştir. Önerilen kontrolörün uygulaması kesikli zamanda yapılmıştır. Bu kontrolör dışmerkezli olması ve kolayca gerçeklenebilmesi yanında tüm yönlendirme kısıtlarını (sıra uzunluklarının ve akış oranlarının negatif olmaması ve akış oranlarının link kapasitelerini aşmaması) sağlamakta, trafiğin döngüye girmesini önlemekte ve ağa giriş oranlarının belli limitleri aşmaması durumunda tüm sıra uzunluklarını sonlu zamanda sıfırlayarak tüm araçları sonlu zamanda varış noktalarına ulaştırmaktadır. Önerilen algoritma nodlarda kuyrukta sıra bekleyen araçları sırasıyla ele alarak, her bir kontrol periyodunda tam sayıda araç yönlendirilmesini sağlamaktadır. Algoritmanın uygulaması sırasında her bir nodun akış aşağı nodlarındaki tıkanıklık ölçeği değerlerini bilmesi yeterlidir. Bunun dışında diğer nodlarla hiç bir bilgi alış-verişine gerek duyulmadan kontrol için gerekli hesaplamalar yerel olarak nodlarda yapılabilmektedir.

Bu çalışmada, ikinci bölümde tanıtılan matematiksel model temel alınarak, trafik ağlarında tıkanıklığı önleyici yeni bir kontrol yaklaşımı önerilmiştir. Yönlendirme ve akış kontrolunun bir arada ele alındığı bu yaklaşım, [8]'da önerilen kontrol yaklaşımı temel alınarak geliştirilmiştir.

Geliştirilen kontrol yaklaşımı için tıkanıklık ölçeği her bir n nodu ve l varış nodu için aşağıdaki şekilde tanımlanmıştır ($l \in J_n' := \cup_{m \in O_n} J_m$):

$$p_n^l(k) = \sum_{\mu \in I_n} \alpha_\mu^l \, \sigma_\mu^l(k) + \beta_n^l \, \zeta_n^l(k) \tag{3.1}$$

$$\sigma_{\mu}^{l}(k) = \begin{cases} \dfrac{\rho_{\mu,f_\mu}^{l}(k)}{\rho_{\mu,f_\mu}(k)} \left(\rho_{\mu,f_\mu}(k) - \rho_{cr,\mu}\right), & \rho_{\mu,f_\mu}(k) > \rho_{cr,\mu} \\ 0, & \rho_{\mu,f_\mu}(k) \leq \rho_{cr,\mu} \end{cases} \tag{3.2}$$

$$\zeta_{n}^{l}(k+1) = \zeta_{n}^{l}(k) + T\left(r_{n}^{l}(k) - \hat{r}_{n}^{l}(k)\right) \tag{3.3}$$

Burada $\alpha_{\mu}^{l} > 0$ $(\mu \in I_n)$ ve $\beta_{n}^{l} > 0$ tasarımcı tarafından seçilen sabit kontrolör parametreleridir. Bu parametreler, n nodundaki kuyruk uzunluğu ile akış-yukarı linklerdeki yüksek yoğunluk değerlerinin n nodunda oluşan tıkanıklığa olan göreceli katkılarını yansıtmaktadır. Diğer değişkenler ise aşağıdaki şekilde tanımlanmıştır:

$\zeta_{n}^{l}(k)$: Daha önce n nodundan ağa katılamamış olup kuyrukta bekleyen l varış noduna sahip araçların oluşturduğu kuyruğun kT zamanındaki uzunluğu.

$r_{n}^{l}(k)$: kT zamanında l varış noduna ulaşmak üzere n nodundan birim zamanda ağa katılmak isteyen araç sayısı.

$\hat{r}_{n}^{l}(k)$: Varış nodu l olan araçların kT zamanında n nodundan trafiğe katılma oranı (birim zamanda trafiğe katılan araç sayısı _ bu değer akış kontrolu ile belirlenir).

Tanımlanan bu tıkanıklık ölçeğine dayalı olarak aşağıda verilen algoritma önerilmektedir. Bu algoritmada yukarıda tanımlanmış olan $J_{n}^{'}$ kümesinin yanısıra bir de S_{n}^{l} kümesi kullanılmaktadır. S_{n}^{l} kümesi n nodundan l varış noduna uzanan alternatif yollardan her birinin ilk nodunu içeren bir kümeyi temsil etmektedir $(l \in J_{n}^{'})$. $J_{n}^{'}$ ve S_{n}^{l} kümelerinin her ikisinin de sıralı küme olduğu kabul edilmiştir ve $J_{n}^{'}(t)$ ve $S_{n}^{l}(h)$ sırasıyla $J_{n}^{'}$ kümesinin t'inci elemanını ve S_{n}^{l} kümesinin h'inci elemanını göstermektedir. $J_{n}^{'}$ kümesinin elemanları gelişigüzel sıralanabilmektedir (kullanılan sıralamanın sonuca her hangi bir etkisi yoktur). Ancak, S_{n}^{l} kümesinin sıralaması belirlenirken n nodundan l noduna gitmek için en uygun (en kısa veya tahmini seyahat sürelerine göre en hızlı) yola ait ilk nod bu kümenin

ilk elemanı olarak belirlenmelidir. Diğer yollara ait ilk nodlar da tanımladıkları yolların uygunluk sıralarına göre sıralanmalıdır. Ayrıca, aşağıda $r_{n,\max}$, n nodunda trafiğe katılma oranının alabileceği en büyük değeri göstermektedir. Ağdaki her bir n nodu için k'ıncı örnekleme anında uygulanması önerilen algoritma:

1. $\hat{r}_n^l(k) = 0, \quad \forall l \in J_n'$.

2. n nodundan ağa katılmak üzere bekleyen araç yoksa 8. adıma git.

3. n nodundan ağa katılmak üzere bekleyen araçlardan ilkinin l varış nodunu belirle.

   $h = 1$.

4. $j = S_n^l(h)$.

5. $p_n^l(k) \geq p_j^l(k)$ ise aracın trafiğe katılmasına izin ver:

   $\hat{r}_n^l(k) = \hat{r}_n^l(k) + \frac{1}{T}$

   7. adıma git.

6. $h = h + 1$.

   $h \leq s(S_n^l)$ ise 4. adıma dön.

   Aksi durumda 8. adıma git.

7. $\sum_{l \in J_n'} \hat{r}_n^l(k) < r_{n,\max}$ ise 2. adıma dön.

   Aksi durumda 8. adıma git.

8. $t = 1$.

9. $l = J_n'(t), \quad h = 1$.

10. $j = S_n^l(h)$.

11. $p_n^l(k) \geq p_j^l(k)$ ise, n nodunda bulunan ve varış nodu l olan araçları j noduna yönlendir:

$$\phi_{n,m}^l(k) = \begin{cases} 1\,, & m \in O_n \cap I_j \\ \\ 0\,, & m \notin O_n \cap I_j \end{cases}$$

14. adıma git.

12. $h = h + 1$.

$h \leq s(S_n^l)$ ise 10. adıma dön.

13. n nodunda bulunan ve varış nodu l olan araçları tıkanıklık ölçeği değerleri ile ters orantılı olacak şekilde paylaştırarak akış aşağı nodlara yönlendir:

$$\phi_{n,m}^l(k) = \begin{cases} \dfrac{\dfrac{1}{p_j^l(k)}}{\displaystyle\sum_{\nu \in S_n^l} \frac{1}{p_\nu^l(k)}}, & m \in O_n \cap I_j,\ j \in S_n^l \\ \\ 0, & m \notin O_n \cap I_{S_n^l} \end{cases}$$

14. $t = t + 1$, $t \leq s(J_n')$ ise 9. adıma dön.

Aksi durumda DUR.

Yukarıdaki algoritma ulaşım ağları için hem yönlendirme kontrolu hem de akış kontrolu problemlerine çözüm getirmektedir. Önerilen algoritma her bir nodda yerel olarak uygulanabilmektedir. Bu uygulama sırasında ağdaki diğer nodlarla ilgili olarak gereken tek bilgi akış-aşağı nodlara ait tıkanıklık ölçeği değerleridir. Böylece, önerilen algoritma nodlar arası çok az bilgi alış-verişi gerektirmesi anlamında dışmerkezlidir. Ayrıca önerilen algoritma nodlar arası senkronizasyon gerektirmemektedir (her bir nod kendisi için en son hesaplanan tıkanıklık miktarını kullanmaktadır).

Önerilen algoritmada, her bir örnekleme anında, her bir nod için öncelikle o örnekleme anında o noddan ağa katılacak araç miktarı belirlenir. Bu değer

akış kontrolu yardımıyla belirlenir. Akış kontrolu yaklaşımında kuyrukta bekleyen araçlar sırayla ele alınır. Bir başka deyişle, kuyrukta bekleyen bir aracın ağa katılıp katılamayacağının incelenebilmesi için öncelikle kuyrukta kendisinin önünde bekleyen aracın ağa katılmış olması gerekmektedir. Ağa katılabilecek araç sayısı yine tıkanıklık miktarlarına bağlı olarak belirlenir. Bir araç için, eğer kuyrukta bulunduğu nodun akış-aşağı nodlarından en az birinin tıkanıklık miktarı bulunduğu nodun tıkanıklık miktarından daha küçük veya eşit ise, ağa katılmasına izin verilir. Aksi halde, o araç arkasındaki tüm araçlar ile birlikte bir sonraki örnekleme anına kadar kuyrukta kalır.

Nodlardan ağa katılacak araç miktarlarının belirlenmesinin ardından, nodlarda bulunan araçların akış-aşağı linklere dağılım oranlarını veren değerler hesaplanır. Bu değerler, o noda ve akış-aşağı nodlara ait tıkanıklık miktarları kullanılarak yönlendirme kontrolu yardımıyla bulunur. n nodunda bulunan araçlar akış-aşağı nodlar içinden tıkanıklık miktarı n nodundaki tıkanıklık miktarına oranla daha küçük olan nodlar arasından en çok tercih edilen yolun (n nodundan l varış noduna uzanan yollar arasından) ilk noduna yönlendirilir. Eğer akış-aşağı nodların tümünün sahip olduğu tıkanıklık miktarları n nodununkinden daha büyük ise n nodunda bulunan araçlar akış-aşağı linklere tıkanıklık miktarları ile ters orantılı olacak şekilde paylaştırılır.

4. BENZETİM PROGRAMI

Bu çalışmada önerilen ve bir önceki bölümde tanıtılan kontrol algoritmasının gerçek trafik ağlarında göstereceği performansı önceden saptayabilmek için benzetim çalışmaları yapılmıştır. Bu amaçla, UNIX ortamında C programlama dilinde bir benzetim programı geliştirilmiştir. Prof. Markos Papageorgiou ve Dr. Albert Messmer tarafından hazırlanan METANET benzetim paket programı temel alınarak hazırlanan bu programın önerilen kontrol algoritmasına hizmet edebilmesi için özgün programa bir kontrol modülü eklenmiştir. Bu kontrol modülünde:

- Her bir örnekleme anında nodlardan ağa katılacak araç miktarı ve ağa katılan bu araçların l varış nodları önerilen akış kontrolu algoritması yardımıyla belirlenir. Ağa katılan araçların l varış nodları geliştirilmiş olan rastgele sayı atayan bir üreteç tarafından belirlenmektedir. Bu üreteç n nodunda kuyrukta bulunan araçlar (daha önceki örnekleme zamanlarında ağa katılamamış olup kuyrukta bekleyen araçlar ile birlikte o örnekleme anında ağa katılmak üzere n noduna gelen araçların tümü) içinden biri için o kuyrukta bekleyen l varış noduna ait araçların kuyrukta bekleyen tüm araçlara oranı değerindeki olasılıkla $(= (\zeta_n^l(k) + T\, r_n^l(k)) / \sum_{\nu \in J_n'} (\zeta_n^\nu(k) + T\, r_n^\nu(k)))$ l nodunu varış nodu olarak seçmektedir.

- Birden fazla akış aşağı linke sahip nodlarda bulunan araçların bu linklere dağılım oranını veren değerler (splitting rate) tıkanıklık ölçeği değerleri temel alınarak önerilen yönlendirme kontrolu yaklaşımı ile hesaplanır.

- Her bir örnekleme anında her bir n nodu ve her bir l varış nodu için örnekleme periyodu sonunda oluşan kuyruk uzunlukları hesaplanır. Bu değerlere bağlı olarak da, bir sonraki örnekleme periyodunda kullanılacak olan tıkanıklık

ölçeği değerleri hesaplanır. Tıkanıklık ölçeği tanımında yer alan sabit değerli parametreler kullanıcı tarafından belirlenir ve bir giriş dosyasında tanımlanır.

Geliştirilen bu kontrol modülünün dökümü Ek-1'de verilmiştir. Bu kontrol modülünün yanı sıra özgün programın diğer bölümlerinde de değişiklikler yapılması gerekmiştir:

- Özgün METANET programı her bir nodda en çok iki giriş ve bir çıkış veya bir giriş ve iki çıkış linkine izin vermektedir. Her ne kadar, daha fazla sayıda giriş/çıkış linki bulunan nodlar, sanal nodlar eklenerek gösterilebilse de, bu yaklaşım toplam nod sayısını arttırmakta, bu da hem programın kullanımını güçleştirmekte hem de benzetim süresini artırmaktadır. Bu dezavantajları gidermek amacıyla, program üzerinde değişiklikler yapılarak her bir nodun iki giriş ve iki çıkış linkine sahip olabilmesi sağlanmıştır.

- Ağa giriş bölümlerindeki (on-ramp) ortalama hızın hesaplandığı ilgili bölüm (ORIGSPEED( )) değiştirilmiştir. Yeni haliyle ORIGSPEED( ) fonksiyonunun (2.16) denklemleminde tanımlandığı şekliyle $v_n(k)$ değişkeninin değerini hesaplaması sağlanmıştır (bu bölüm Ek-2'de sunulmuştur). Orjinal haliyle ORIGSPEED( ) fonksiyonunda (2.16) denklemindekinden farklı olarak ağa giriş bölümlerindeki ortalama hız değeri bu nodun sahip olduğu çıkış linklerinin hızları üzerinden bir ortalama alınarak hesaplanmaktadır.

- Ağdan çıkış bölümlerindeki (off-ramp) yoğunluğun hesaplandığı ilgili bölüm (DESTDENSITY( )) değiştirilmiştir. Yeni haliyle DESTDENSITY( ) fonksiyonunun (2.18) denklemleminde tanımlandığı şekliyle $\rho_n(k)$ değişkeninin değerini hesaplaması sağlanmıştır (bu bölüm Ek-3'de sunulmuştur). Orjinal haliyle DESTDENSITY( ) fonksiyonunda (2.18) denklemindekinden farklı olarak ağdan çıkış bölümlerindeki yoğunluk değeri bu nodun sahip olduğu giriş linklerinin yoğunluk değerleri üzerinden bir ortalama alınarak hesaplanmaktadır.

- Kullanılan ağ ile ilgili bilgilerin yer aldığı giriş dosyasına eklenen bilgilerin program tarafından algılanmasını sağlayacak değişiklikler yapılmıştır. Orjinal programdaki giriş dosyasına ek olarak tanımı yapılan yeni bilgiler aşağıda verilmiştir:

  - Her bir n noduna ait (eğer varsa) ağa giriş linki için v_n^M değeri,
  - Her bir n noduna ait (eğer varsa) ağdan çıkış linki için v_n^o değeri,
  - Her bir m linki - l varış nodu çifti için α_m^l değeri,
  - Her bir n nodu - l varış nodu çifti için β_n^l değeri,
  - Her bir n nodu için tanımlanmış olan S_n^l kümesinin elemanlarına bağlı kalınarak, n nodundan ayrılan iki linkten l noduna ulaşmak için tercih edilenin n nodunun kaçıncı çıkış linki olduğunu belirleyen değerler.

Yapılan bu değişikliklerin ardından elde edilen benzetim programının işletimi için bir sonraki alt-bölümde detaylı olarak anlatılacak olan giriş dosyalarına ihtiyaç vardır. Bahsedilen giriş dosyalarından biri olan benzetim kontrol dosyasında belirlenen benzetim süresi içinde çalışan programda, her bir örnekleme periyodunda her bir linkin her bir bölümü için trafik değişkenleri olan yoğunluk, akış oranı ve ortalama hız değerleri hesaplanarak bu değerler ilerde açıklanacak olan benzetim çıkış dosyasına yazılır. Benzetim işlemleri giriş dosyasında girilen sürenin sona ermesine veya yine giriş dosyasında girilen araç giriş oranlarına ait bilgilerin sona ermesine kadar devam eder. Program sona erdikten sonra, istenildiği takdirde, bu programa bağlı olarak geliştirilmiş olan bir grafik dosyası düzenleme programı çalıştırılarak, istenilen değişkenlere ait bilgiler zamana bağlı grafikler çizdirilmek üzere hazırlanabilmektedir. Grafikler ise, düzenlenen bu dosyalar kullanılarak MATLAB paket programında hazırlanmış olan bir program çalıştırılarak çizdirilebilmektedir.

4.1. Veri Giriş Dosyaları:

4.1.1. Veri giriş dosyalarının genel özellikleri:

Tüm giriş (ve çıkış) dosyaları text (ASCII) formatındadır. Genellikle giriş dosyalarının her bir satırının ilk kolonunda o satırdaki bilgilerin tipini belirleyen özel bir karakter vardır. Bu karakterler ve anlamları aşağıda verilmiştir:

'C': Açıklama satırı. Kullanıcıyı bilgilendirmek için kullanılan bu satır program tarafından göz ardı edilir.

'H': Başlık satırı. Dosyada açıklama satırları dışında ilk satırı oluşturmalıdır.

' | ': Veri satırı. Her hangi bir veri satırını ifade eder.

'E': Liste sonu. Dosya içinde bir liste bulunuyorsa listenin sona erdiğini belirleyen satırdır.

Bu karakterler tüm giriş dosyaları için geçerlidir. Ayrıca, bu karakterlerin dışında sadece trafik veri dosyalarında kullanılan birkaç karakter daha vardır:

'F': Trafik verisi tanımlama satırı.

'T': Zaman aralığı tanımlama satırı.

'N': Trafik verilerinin ait olduğu yerleri (nod, link) tanımlama satırı.

Eğer ilk kolonda yukarıdaki karakterlerden hiç biri bulunmazsa (boşluk karakteri olursa) bu satır program tarafından bir önceki satırın devamı olarak algılanır.

Bundan sonraki 4.1.2-4.1.6 altbölümlerinde her bir giriş dosyası için açıklayıcı bilgiye ve yazım formatına yer verilmiştir. Ayrıca Bölüm 5.2'de verilen örneklere ait veri giriş dosyaları da eklerde sunulmuştur.

4.1.2. Benzetim kontrol dosyası: (*dosyaadı.CTR*) Benzetimin başlangıç ve bitiş zamanlarına, örnekleme periyoduna, benzetim sonuçlarının alınacağı (çıkış

dosyaadı.CTR

H *Başlık*

| *sim_başlangıç_zamanı* *sim_bitiş_zamanı* *sim_zaman_aralığı*

C Benzetim sonuçlarının yazılacağı

| *format_tipi* *başlangıç_zamanı* *bitiş_zamanı* *zaman_aralığı*

C Çıkış dosyasına yazılması istenen trafik değişkenlerinin link ve bölümlerinin listesi (Bu liste verilmediği durumda, tüm link ve bölümlerdeki değerler yazılır.)

| *link_adı* *link_bölümü*

| ...

| ...

| ...

E

(dosya sonu)

Şekil 4.1. *dosyaadı.CTR* dosyasının formatı

dosyasına yazdırılacağı) zaman aralığına, periyoduna ve Bölüm 4.2.2'de açıklanacak olan format seçimine (a, A, b, B, c veya C) ait bilgilerin girildiği giriş dosyasıdır. Bu dosyanın formatı Şekil 4.1'de gösterilmiştir.

4.1.3. Ağ tanıtım dosyası: (*dosyaadı.NWD*) Benzetimi yapılacak olan trafik ağının topolojisini tanımlayan bilgilerin yer aldığı giriş dosyasıdır. Nodlara, linklere, ağa giriş ve çıkış bölümlerine ait listelerin yanı sıra ağın matematiksel modelinin tanımlanmasında kullanılan ve ikinci bölümde bahsedilen parametrelerin (κ, ν, τ vb.) değerleri ve önerilen kontrolün uygulanabilirliği için gerekli olan α_m^l, β_n^l değerlerinin ve S_n^l kümesinin elemanlarının öncelik sırasının listesi de bu giriş dosyasında yer alır. Bu dosyanın formatı Şekil 4.2'de gösterilmiştir.

4.1.4. Başlangıç değerleri dosyası: (*dosyaadı.INI*) Linklerdeki yoğunluk değişkenlerine ve her bir linkteki akış oranının varış nodlarına göre dağılım oranını ifade eden değişkenlere (composition rate) ait başlangıç değerlerini içeren veri giriş dosyasıdır. Bu dosyanın adı *dosyaadı.NWD* dosyasının adı ile aynı olmalıdır. Bu

dosyaadı.NWD

H *Başlık*

C Global ağ parametreleri

| τ κ ν v_{min} ρ_{max} δ ϕ

E

C Ağa giriş linklerinin listesi

| *link_adı* *şerit_sayısı* v_f v^M $\hat{r}_{max}$

| ...

| ...

| ...

E

C Ağdaki linklerinin listesi

| *link_adı* *şerit_sayısı* *kapasite_değeri* v_f ρ_{cr} ...

| ... *link_uzunluğu* *bölüm_sayısı*

| ...

| ...

| ...

E

C Ağdan ayrılan linklerinin listesi

| *link_adı* *şerit_sayısı* v_f v^O

| ...

| ...

| ...

E

Şekil 4.2. *dosyaadı.NWD* dosyasının formatı

C Ağdaki nodların ve linklerinin bağlantıları

C (Her bir nod için 3 satır ayrılarak, ilk satıra nod adı, ikinci satıra noda giren

C linklerin adı, üçüncü satıra ise noddan ayrılan linklerin adı yazılır.)

| *nod_adı*

| *giriş_linki_1* *giriş_linki_2*

| *çıkış_linki_1* *çıkış_linki_2*

| ...

| ...

| ...

E

C α_m^l değerleri

| *link_adı* *varış_nodu_1* *varış_nodu_2* ...

| ...

| ...

| ...

E

C β_n^l değerleri

| *nod_adı* *varış_nodu_1* *varış_nodu_2* ...

| ...

| ...

| ...

E

C S_n^l kümesi elemanlarının öncelik sırası

| *nod_adı* *varış_nodu_1* *varış_nodu_2* ...

| ...

| ...

| ...

E

(dosya sonu)

Şekil 4.2. (Devam) *dosyaadı.NWD* dosyasının formatı

dosyaadı.INI

H *Başlık*

C Linklerdeki yoğunluk değişkenine ait başlangıç değerleri

link_adı	*ρ (ilk bölüm)*	*ρ (son bölüm)*
...		
...		
...		

E

C Linklerdeki akış oranının varış nodlarına göre dağılımını veren oranlar

link_adı	*varış_nodu_1*	*varış_nodu_2*	*varış_nodu_3*	...
	dağ_1_(ilk_böl.)	*dağ_2_(son_böl.)*	*dağ_3_(ilk_böl.)*	...
	dağ_1_(son_böl.)	*dağ_2_(son_böl.)*	*dağ_3_(son_böl.)*	...
...				
...				
...				

E

(dosya sonu)

Şekil 4.3. *dosyaadı.INI* dosyasının formatı

giriş dosyasının formatı Şekil 4.3'de gösterilmiştir.

4.1.5. Araç girişi veri dosyası: (*dosyaadı.MSD*) Giriş bölümlerinden ağa katılmak isteyen araç miktarlarının tanımlandığı giriş dosyasıdır. Bu verilerin girişi istenilen periyod ile yapılabilir; benzetim periyodu ile eşit olmak zorunda değildir. Veri girişinin benzetim için örnekleme periyodu olan T'den daha büyük bir periyodla (örneğin p) yapıldığı kabul edilirse; dosyadan (k) ile $(k + (p/T))$ benzetim anlarına ait araç giriş miktarlarının okunmasının ardından, (k) ile $(k+(p/T))$ anları arasında geçen her bir benzetim anı $(k + 1,\ k + 2,...,\ k + (p/T) - 1)$ için araç giriş miktarı, (k) anı ile $(k+(p/T))$ anı arasında araç giriş miktarının (k) anı için girilen

dosyaadı.MSD

H *Başlık*

F *veri_formatı*

C

T *başlangıç_zamanı* *zaman_aralığı*

C Araç girişinin yapıldığı noktalar

N *link_adı* ...

C Yukarıda verilen noktalara gelen araç miktarları

| ...

| ...

| ...

(dosya sonu)

Şekil 4.4. *dosyaadı.MSD* dosyasının formatı

değer ile $(k + (p/T))$ anı için girilen değer arasında doğrusal olarak değiştiği kabul edilerek hesaplanır. Bu dosyanın formatı Şekil 4.4'de gösterilmiştir.

4.1.6. Giriş-çıkış dağılım oranları dosyası: (*dosyaadı.ODM*) Ağa giriş bölümlerinden katılmak isteyen araçların varış nodlarına göre dağılım oranlarının $(\gamma_n^l(k))$ zamana bağlı olarak tanımlandığı giriş dosyasıdır. Bu dosyanın adı *dosyaadı.MSD* dosyasının adı ile aynı olmalıdır. Bu giriş dosyasının formatı Şekil 4.5'de gösterilmiştir.

dosyaadı.ODM

H *Başlık*

F *veri_formatı*

C

T *başlangıç_zamanı* *zaman_aralığı*

C

C Giriş-çıkış dağılım oranları değerlerinin yazılım yapısı

N *giriş_linki_1* *varış_nodu_1.1* *varış_nodu_1.2* ...

giriş_linki_2 *varış_nodu_2.1* *varış_nodu_2.2* ...

...

C Giriş-çıkış dağılım oranları değerleri (yukarıda verilen yapıya uygun olarak)

| ...

| ...

| ...

(dosya sonu)

Şekil 4.5. *dosyaadı.ODM* dosyasının formatı

4.2. Çıkış Dosyaları:

4.2.1. Veri girişi test dosyası: (*dosyaadı.CHK*) Kullanıcının girilen verileri bir kez daha gözden geçirebilmesi imkanınını sağlamak amacıyla, programın işletimi sırasında *dosyaadı.NWD* ve *dosyaadı.INI* dosyalarının okunması sırasında oluşturulan bir çıkış dosyasıdır. *dosyaadı.NWD* ve *dosyaadı.INI* dosyalarından okunan ağ topolojisi ve benzetim kontrolu ile ilgili tüm verilerin düzenlenmesiyle oluşturulan bu dosyada bu bilgilerin yanısıra ayrıca program tarafından o sırada hesaplanan bazı bilgiler de yer alır:

- Denklem (2.13)'deki denklem kullanılarak her bir link için hesaplanan a_m değişkeninin değerleri.
- Her bir link için o link üzerinden ulaşılabilecek varış nodları kümesinin elemanları.

4.2.2. Benzetim sonuç dosyası: (*dosyaadı.SMD*) *dosyaadı.CTR* giriş dosyasında tanımlanan zaman aralığı içinde ve periyodda yine aynı dosyada yapılan format seçimine göre istenilen linkler ve bölümlerdeki trafik değişkenlerinin değerlerinin yazıldığı çıkış dosyasıdır. Seçilen formata göre benzetim sonuç dosyasına:

'a' veya 'A' formatı seçildiğinde: Liste ile verilmiş olan link ve bölümlerdeki trafik değişkenlerinin değerleri,

'b' veya 'B' formatı seçildiğinde: Çıktı periyodu üzerinden ortalama alınarak liste ile verilmiş olan link ve bölümlerdeki trafik değişkenlerinin değerleri,

'c' veya 'C' formatı seçildiğinde: Ağdaki tüm link ve bölümlerdeki trafik değişkenlerinin değerleri,

'd' veya 'D' formatı seçildiğinde: Çıktı periyodu üzerinden ortalama alınarak ağdaki tüm link ve bölümlerdeki trafik değişkenlerinin değerleri

yazılmaktadır. Format seçiminin büyük harf ile yazılması linklerdeki trafik değişkenlerinin değerlerinin yanı sıra her bir linkteki araçların varış nodlarına göre dağılım oranlarının da yazılmasını sağlamaktadır.

Ayrıca benzetim sonuçlarına bağlı olarak hesaplanan toplam seyahat süresi, toplam kuyrukta bekleme süresi, benzetim süresince ağa katılan araç sayısı, ağdan ayrılan araç sayısı, seyahat edilen toplam uzunluk, nodlardaki maksimum kuyruk uzunlukları ve harcanan yakıt miktarı gibi belirli performans ölçütlerinin değerleri de bu çıkış dosyasına yazılmaktadır. Bahsedilen bu performans ölçütlerinin değerleri [17]'de önerildiği gibi hesaplanmaktadır:

- *Toplam seyahat süresi (TSS):* Benzetim süresince ağda seyahat eden araçların ağ içinde harcadıkları toplam süre (*saat*).

$$TSS = \sum_k \left(\sum_m \left(\sum_i \rho_{m,i}(k) \times \lambda_m \times L_m \times T \right) \right)$$

- *Toplam kuyrukta bekleme süresi (TBS):* Benzetim süresince ağa katılmak üzere giriş bölümlerine gelen araçların kuyrukta beklerken harcadıkları toplam süre (*saat*).

$$TBS = \sum_k \left(\sum_n \left(\sum_l \zeta_n^l(k) \times T \right) \right)$$

- *Ağa katılan araç sayısı (AKAS):* Benzetim süresince ağa katılmak üzere giriş bölümlerine gelen araçlardan uygulanan kontrol gereğince ağa katılmasına izin verilenlerin toplam sayısı (*araç*).

$$AKAS = \sum_k \left(\sum_n \left(\sum_l \hat{r}_n^l(k) \times T \right) \right)$$

- *Ağdan ayrılan araç sayısı (AAAS):* Benzetim süresince ağa katılmış olan araçlardan varış yerlerine ulaşarak ağdan ayrılanların sayısı (*araç*). (Aşağıdaki denklemde D, ağdan ayrılan çıkış linklerinin kümesini ifade etmektedir. Ayrıca, $N(.)$, $(.)$ linkinin çıkış linki olduğu nodu temsil etmektedir.)

$$AAAS = \sum_k \left(\sum_{m \in D} q_{m,0}^{N(m)}(k) \times T \right)$$

- *Toplam seyahat uzunluğu (TSU):* Benzetim süresince ağa katılan araçların ağ içinde aldıkları toplam yol uzunluğu (*km*).

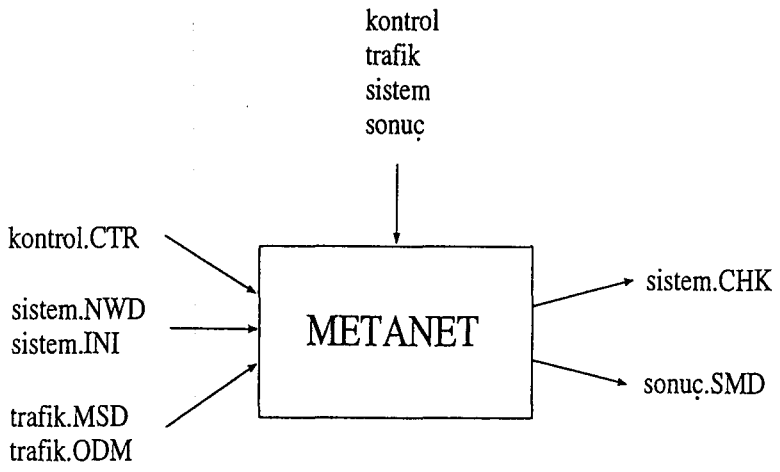
$$TSU = \sum_k \left(\sum_m \left(\sum_i \left(\sum_l q_{m,i}^l(k) \times L_m \times T \right) \right) \right)$$

- *Maksimum kuyruk uzunlukları (MKU(n)):* Benzetim süresince her bir nod için oluşan kuyruk uzunluğunun maksimum değeri (*araç*).

$$MKU(n) = \max_k \left(\sum_l \left(\zeta_n^l(k) \right) \right), \qquad \forall n.$$

- *Harcanan yakıt (HY):* Harcanan yakıt miktarı (*litre*). [17]'de yine [17]'de kullanılan ve 2. Bölüm'de tanıtılmış olan trafik ağı modeline dayalı olarak benzetim süresince ağda seyahat eden araçlar tarafından harcanan yakıt miktarının aşağıdaki denklem yardımıyla hesaplanabileceği kabul edilmiştir.

$$HY = \begin{cases} \frac{T}{100} \sum_k \left(\sum_m \left(\sum_i L_m \left(4.49\, q_{m,i}(k) + 122\, \lambda_m\, \rho_{m,i}(k) \right.\right.\right. \\ \qquad \left.\left.\left. +0.0016\, q_{m,i}(k)\, \left(v_{m,i}(k) - 60\right)^2 \right) \right) \right), & v_{m,i}(k) > 60 \\ \frac{T}{100} \sum_k \left(\sum_m \left(\sum_i L_m \left(4.49\, q_{m,i}(k) + 122\, \lambda_m\, \rho_{m,i}(k) \right) \right) \right), & v_{m,i}(k) \leq 60 \end{cases}$$



Şekil 4.6. METANET'in genel yapısı

4.3. Programın Çalışması:

C programlama dilinde hazırlanmış olan benzetim programı üç ayrı programdan oluşmaktadır:

metanet.c: Ana program,

mn_subs.c: Benzetim ile ilgili modüllerin yer aldığı program,

mn_utils.c: Genel giriş/çıkış fonksiyonlarının yer aldığı program.

Programın UNIX ortamında,

cc metanet.c -lm -o metanet

şeklinde derlenmesinin ardından *metanet* işletilebilir program dosyası çalıştırılabilir. Program öncelikle kullanıcıdan giriş ve çıkış dosyalarının isimlerini isteyecek (Şekil 4.6), ardından da aşağıda özetlenen program algoritmasını işletime sokacaktır:

- Trafik ağının topolojisini tanımlayan giriş dosyası (*dosyaadı.NWD*) okunur.
- Benzetimin başlangıç ve bitiş zamanlarının, örnekleme süresinin ve benzetim sonuçlarının alınacağı periyod gibi benzetimin kontrolu ile ilgili bilgilerin yer

aldığı giriş dosyası (*dosyaadı.CTR*) okunur.

- *dosyaadı.INI* giriş dosyasından her bir linkin başlangıcındaki ve sonundaki yoğunluk değişkenlerine ait başlangıç değerleri okunur. Okunan bu değerler yardımıyla (2.12) ve (2.8) denklemleri kullanılarak her bir link için hız ve akış oranlarına ait başlangıç değerleri hesaplanır. Ardından *dosyaadı.MSD* giriş dosyasından ilk benzetim periyodu için nodlardan ağa katılmak isteyen araç miktarları okunur. Ağa katılmak üzere nodlara gelen bu araçların varış nodlarına göre dağılım oranlarının *dosyaadı.ODM* dosyasından okunmasının da ardından, benzetimin yatışkın durum değerlerinden başlaması için trafik değişkenleri uygun değerlere ulaşıncaya kadar benzetim algoritması işletilir (benzetim başlamadan önce ağa belli miktarda araç katılması sağlanır). Bu süre içinde nodlardan ağa katılmak isteyen araç miktarları her bir periyod için sabit (*dosyaadı.MSD* dosyasından okunan, benzetimin ilk periyodu için girilmiş olan değerler) olarak alınır.

- Ana benzetim döngüsü: Kontrol dosyasından okunan benzetim süresi veya giriş dosyasından okunan araç girişi bilgileri sona erinceye kadar her bir k örnekleme periyodunda sırasıyla aşağıdaki işlemler yapılır (her bir adım için o adımda kullanılan başlıca kontrol modülü fonksiyonu da belirtilmiştir):

  * Ağa giriş bölümlerindeki ortalama hız ve ağdan çıkış bölümlerindeki yoğunluk değerleri (2.16) ve (2.18) denklemlerinde ifade edildiği şekilde hesaplanır (ORIGSPEED( ) ve DESTDENSITY( )).

  * Trafik akış dinamiği denklemlerinin sınır koşullarından olan her bir linkin başlangıcındaki ortalama hız ve her bir linkin sonundaki yoğunluk değerleri (2.15) ve (2.17) denklemleriyle hesaplanır (BEGINSPEED( ) ve ENDDENSITY( )).

  * Nodlardan ağa katılabilen araç miktarı bir önceki bölümde sunulan algoritmanın akış kontrolu parçasını oluşturan 1 - 7 adımları ile belirlenir (IN_MODEL( )). Ayrıca, ağa katılan her bir aracın varış nodu, ağa

katıldığı nodda bekleyen araçların varış nodlarına göre dağılımını veren oranlarla orantılı olacak şekilde rastgele atanır (FIND_DEST( )).

* Akış yukarı linklerden ve/veya ağ dışından nodlara katılan araçların akış-aşağı linklere dağılımı yapılır (her bir linkin başlangıcındaki akış oranları belirlenir). Bu işlem için (2.14) denklemindeki ifade kullanılır ve $\phi^l_{n,m}(k)$ dağılım oranlarının belirlenmesinde önerilen algoritmanın (ikinci bölümde sunulmuş olan) yönlendirme kontrolu parçasını oluşturan 8 - 14 adımları kullanılır (NODEMODEL( )).

* k örnekleme anına ait ilgili değişken değerleri, benzetim süresince ağa katılan araçların toplam sayısının ve ağdan ayrılan araçların toplam sayısının belirlenebilmesini sağlayacak olan işlemler için kullanılır. Ayrıca, benzetim süresi başladığında ($k = 0$ için) ağda bulunan araç sayısı hesaplanır (BALANCE( )).

* k örnekleme periyoduna ait trafik değişkenlerinin değerleri belirli performans ölçütlerinin (toplam seyahat süresi, toplam kuyrukta bekleme süresi, nodlardaki maksimum kuyruk uzunlukları, seyahat edilen toplam uzunluk, benzetim süresince ağa katılan araç sayısı, ağdan ayrılan araç sayısı ve harcanan yakıt miktarı gibi) değerlerinin hesaplanmasında kullanılır (CRITERIA( )).

* k örnekleme periyoduna ait trafik değişkenlerinin değerleri çıkış dosyasına yazdırılır (WRITE_OUTPUT( )).

* Benzetim zamanı bir periyod artırılır ($k = k + 1$).

* Her bir linkteki her bir bölüm için trafik değişkenlerinin (yoğunluk, akış oranı, ortalama hız) değerleri (2.8) - (2.13) denklemleri kullanılarak hesaplanır (TRAFFICMODEL( )).

* (3.3) denklemiyle nodlardaki kuyruk uzunlukları hesaplanır. Hesaplanan bu değerler ve (3.2) denklemiyle hesaplanan değerler yardımıyla, (3.1) denkleminde ifade edildiği şekilde tıkanıklık ölçeği değerleri belirlenir (CALCULATE_P( )).

* $k+1$ örnekleme periyodu için nodlardan ağa katılmak isteyen araç miktarları belirlenir (NEW_DEMAND( )).

- Benzetim süresi sona erdiğinde ağda bulunan araçların sayısı hesaplanır (BALANCE( )).

- Hesaplanan son trafik değişkenlerinin değerleri çıkış dosyasına yazılır (WRITE_OUTPUT( )).

- Yukarıda bahsedilen ve benzetim döngüsü boyunca hesaplanan performans ölçütlerinin değerleri çıkış dosyasına yazılır (WRITE_CRITERIA( )).

- Benzetim süresince açılan giriş-çıkış dosyaları kapatılır (CLOSE_FILES( )).

5. BENZETİM ÇALIŞMALARI

Önerilen kontrol algoritmasının gerçek trafik ağlarında göstereceği performansı önceden saptayabilmek için benzetim çalışmaları da yapılmıştır. Bu amaçla, çeşitli trafik ağı topolojileri ele alınmış ve bu ağlar üzerinde önerilen kontrolun benzetimi yapılmıştır. Ayrıca, önerilen kontrolun performansını karşılaştırabilmek amacıyla, aynı ağ topolojilerine [17]'de önerilen ve METANET'te kullanılan kontrol yaklaşımı da uygulanmıştır. Her bir ağa aynı trafik durumları için uygulanan bu yaklaşım bir sonraki alt bölümde tanıtılacaktır.

5.1. Alternatif Kontrol Yaklaşımı:

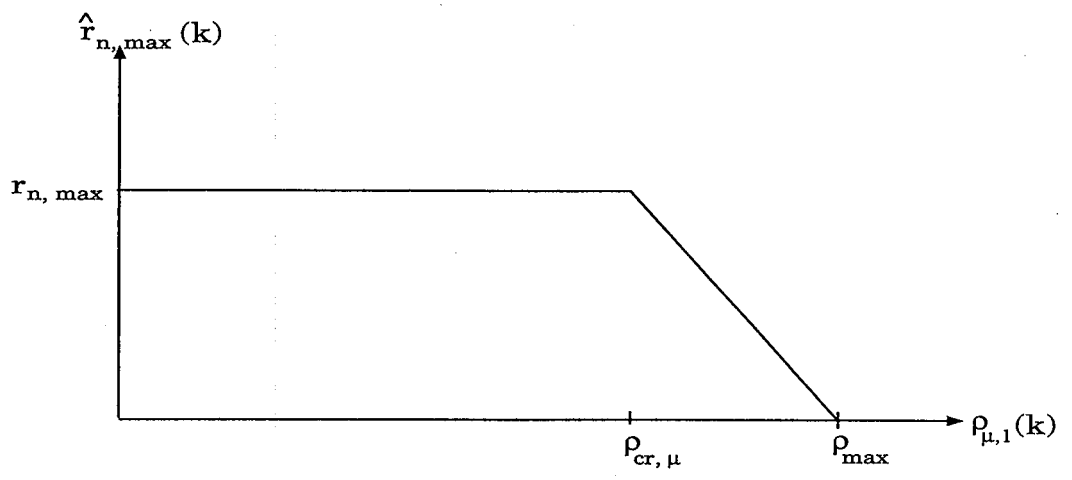
[17]'de önerilen ve METANET'te kullanılan kontrol yaklaşımında akış kontrolu olarak aşağıdaki yaklaşım ele alınmıştır.

$$\hat{r}_n^l(k) = \gamma_n^l(k)\,\hat{r}_n(k) \tag{5.1}$$

$$\hat{r}_n(k) = \min\left(r_n(k) + \frac{1}{T}\sum_{\nu\in J_n'}\zeta_n^\nu(k)\ ,\ \hat{r}_{n,\max}(k)\right) \tag{5.2}$$

$$\hat{r}_{n,\max}(k) = \begin{cases} r_{n,\max}\,, & \rho_{\mu,1}(k) < \rho_{cr,\mu} \\ r_{n,\max}\left(1 - \frac{\rho_{\mu,1}(k)-\rho_{cr,\mu}}{\rho_{max}-\rho_{cr,\mu}}\right), & \rho_{max} \geq \rho_{\mu,1}(k) \geq \rho_{cr,\mu} \\ 0, & \rho_{\mu,1}(k) \geq \rho_{max} \end{cases} \tag{5.3}$$

$(\mu \in O_n)$



Şekil 5.1. $\hat{r}_{n,\max}(k)$ fonksiyonu

Bu yaklaşımda, nodlardan ağa katılacak araç miktarları her bir örnekleme anında ve her bir nod için değişen bir maksimum giriş oranı ile sınırlandırılmıştır (Şekil 5.1). n nodu için hesaplanan maksimum değer n nodunun akış aşağısındaki linkin ilk bölümündeki yoğunluk değeri o link için tanımlanmış olan kritik yoğunluk değerine ulaştığı veya üzerine çıktığı zaman n nodundan izin verilen maksimum giriş miktarının belli oranda düşürülmesi ile hesaplanan değer olarak, aksi durumda ise izin verilen maksimum giriş miktarı olarak belirlenir.

Bölüm 4.1.6'da da belirtildiği gibi $\gamma_n^l(k)$ değerleri ağa giriş bölümünden katılmak isteyen araçların varış nodlarına göre dağılım oranlarını ifade eden değerlerdir. Bu değerler giriş dosyasından okutulmaktadır.

METANET'te kullanılan kontrol yaklaşımında, yönlendirme kontrolu olarak özel bir yaklaşım ele alınmamıştır. Birden fazla çıkış noduna sahip nodlar için araçların akış aşağı nodlara paylaşımı zamana bağlı olarak dışarıdan girilen oranlarda yapılmaktadır.

5.2. Örnek Trafik Ağları İçin Benzetim Çalışmaları:

5.2.1 Birinci örnek trafik ağı için benzetim çalışması:

Bu bölümde, benzetim çalışması amacıyla yapılan çalışmalardan bir örnek

sunulacaktır. Yapılan çalışmada, [17]'de kullanılan ve Şekil 5.2'de şematik olarak gösterilen ağ ele alınmıştır. Giriş ve çıkış linkleri hariç 21 link ve 19 noda sahip olan ağdaki her bir link farklı sayıda şerite ve bölüme (linkin uzunluğuna bağlı olarak) sahiptir. Ağa, 5 kaynak nodundan (U1,...,U5) 5 varış noduna (Z1,...,Z5) ulaşmak üzere araç girişi yapılabilmektedir.

Bu trafik ağına saat 04:00 ile 10:00 arasında olmak üzere 6 saat boyunca araç girişinin yapıldığının kabul edildiği bir durum için benzetim yapılmıştır. Örnekleme periyodunun 10 s. olarak alındığı bu çalışmada, ağ topolojisine ait bilgilerin bulunduğu veri giriş dosyaları [18]'dekine bağlı kalınarak hazırlanmıştır. *ag1.INI* giriş dosyasında değişiklik yapılmamıştır. *ag1.NWD* dosyasına Bölüm 4'de anlatılmış olan ek bilgiler eklenmiştir. *ag1.MSD* dosyasında girilen dışarıdan ağa katılmak isteyen araç miktarlarında değişiklikler yapılmıştır. Bu dosyada tanımlanan benzetim süresince ağdaki 5 giriş bölümünden (U1,...,U5) ağa katılmak isteyen araç miktarlarının zamana bağlı olarak grafikleri sırasıyla Şekil 5.3-5.7'de verilmiştir. *ag1.MSD* dosyasında girilen değerlerin varış nodlarına göre dağılımını veren oranların bulunduğu *ag1.ODM* dosyasında da değişiklikler yapılmıştır. Burada sunulan örnekte bu oranların benzetim süresi boyunca değişmediği kabul edilerek aşağıdaki gibi seçilmiştir:

$$\gamma_{U1}^{Z1} = \gamma_{U1}^{Z2} = \gamma_{U1}^{Z3} = \gamma_{U1}^{Z4} = \gamma_{U1}^{Z5} = 0.2$$

$$\gamma_{U2}^{Z1} = \gamma_{U2}^{Z2} = \gamma_{U2}^{Z3} = \gamma_{U2}^{Z4} = \gamma_{U2}^{Z5} = 0.2$$

$$\gamma_{U3}^{Z1} = \gamma_{U3}^{Z2} = 0.3; \quad \gamma_{U3}^{Z5} = 0.4$$

$$\gamma_{U4}^{Z1} = 0.4; \quad \gamma_{U4}^{Z2} = \gamma_{U4}^{Z5} = 0.3$$

$$\gamma_{U5}^{Z1} = \gamma_{U5}^{Z5} = 0.3; \quad \gamma_{U5}^{Z2} = 0.4$$

Bu benzetim çalışmasına ait tüm veri giriş dosyaları Ek-4'de sunulmuştur.

Karşılaştırma amacıyla, aynı ağa, aynı koşullar altında, METANET'te kullanılan ve Bölüm 5.1'de tanıtılan alternatif yaklaşım da uygulanmıştır. Bu yaklaşım için seçilmiş olan $\phi_{n,m}^{l}(k)$ dağılım oranlarının değerlerinin zamana bağlı olarak

grafikleri Şekil 5.8-5.13'de verilmiştir. Her iki benzetim için elde edilen sonuçlar benzetim süresince hesaplanmış olan performans ölçütlerinin değerleri kullanılarak karşılaştırılmıştır. Elde edilen performans ölçütlerinin (Bölüm 4.2.2'de tanımlanan) değerleri Tablo 5.1'de görülmektedir. Bu değerlerden görüldüğü gibi, önerilen kontrolde benzetim süresince ağa katılmasına izin verilen araçların sayısı, alternatif kontroldekine oranla %4 daha fazladır. Benzetim süresince daha fazla araç girmiş olmasına rağmen önerilen kontrolde seyahat için harcanan zaman alternatif kontrole oranla %24 daha azdır. Ağa katılan araç başına seyahat süresi hesaplandığında, önerilen kontrolde 6.9 *dak.* alternatif kontrolde ise 9.4 *dak.* olduğu görülmektedir. Bu da önerilen kontrolün alternatif kontrole oranla %27'lik bir bir zaman kazancı sağladığını göstermektedir. Ayrıca, kuyrukta beklerken harcanan toplam sürenin de önerilen kontrolde alternatif kontrole oranla %14 daha az olduğu görülmektedir. Bunların yanısıra, önerilen kontrolde seyahat edilen toplam uzunluk alternatif kontrole göre %9 daha fazla olduğu (trafik sıkışıklığını önlemek amacıyla bazı araçların daha uzun - ancak harcanan zaman açısından daha çabuk - yollardan yönlendirilmesinden dolayı) halde harcanan toplam yakıt miktarı %5 daha azdır. Seyahat edilen her 100 *km* başına önerilen kontrolde 8.02 *lt*, alternatif yaklaşımda ise 9.19 *lt* yakıt harcanmış, dolayısıyla önerilen yaklaşımda %13'lük bir yakıt tasarrufu sağlanmıştır.

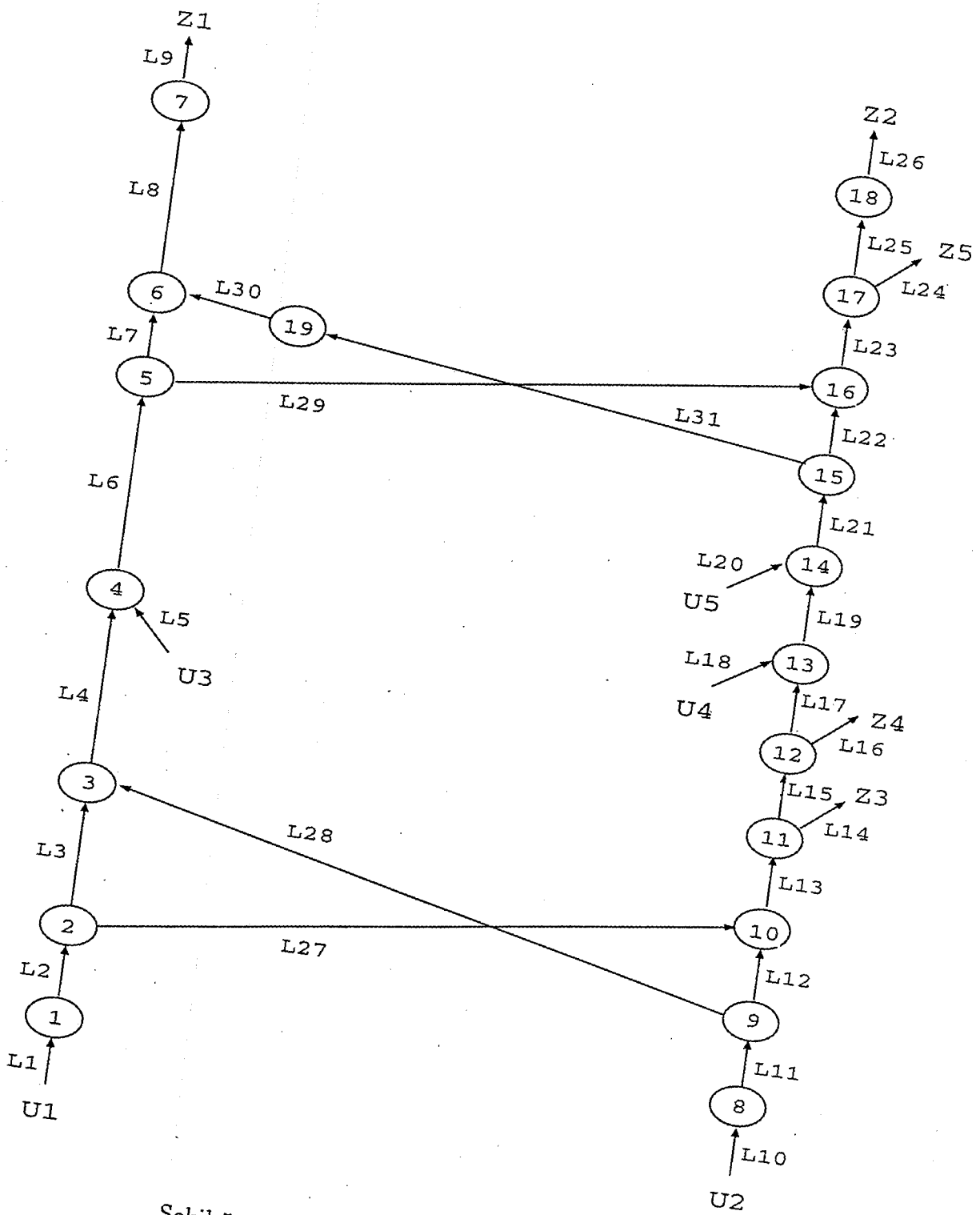
Her iki benzetim için, ağa araç girişinin gerçekleştiği nodlardaki araç kuyruk uzunluklarının toplamının (*araç*) zamana (*günün saati*) bağlı grafiklerine Şekil 5.14'de yer verilmiştir. Bu ve bundan sonra sunulacak olan benzetim sonuçlarına ait tüm şekillerde karşılaştırma amacıyla grafikler aynı şekil üzerinde verilmektedir. Önerilen kontrole ait grafikler sürekli çizgi ile, alternatif kontrole ait grafikler ise kesikli çizgi ile ifade edilmektedir. Bu grafiklerden görülebileceği gibi önerilen kontrolün benzetimi sonunda oluşan toplam kuyruk uzunluğunun maksimum değeri alternatif kontrolünkinin %80'i kadardır. Ayrıca ağdaki her bir giriş bölümünde oluşan kuyrukların zamana bağlı olarak grafikleri de Şekil5.15-5.19'da sunulmuştur.

Ayrıca, ağ üzerinde çeşitli linklerin hemen hemen orta bölümleri için trafik

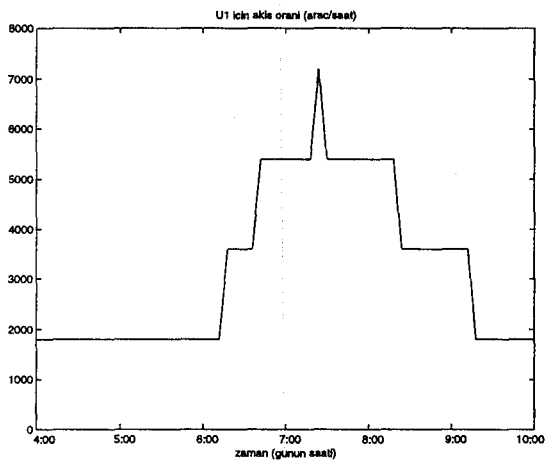
	Önerilen Yaklaşım	Alternatif Yaklaşım
TSS (*saat*)	7401.01	9677.49
TBS (*saat*)	35679.38	41554.59
AKAS (*araç*)	64305	62021
AAAS (*araç*)	62935	60221
TSU (*km*)	337046.70	308304.50
HY (*litre*)	27019.80	28331.70
MKU(U1) (*araç*)	64	5188
MKU(U2) (*araç*)	0	2769
MKU(U3) (*araç*)	2773	1007
MKU(U4) (*araç*)	7199	6117
MKU(U5) (*araç*)	4080	2434

Tablo 5.1. Performans ölçütlerinin değerleri

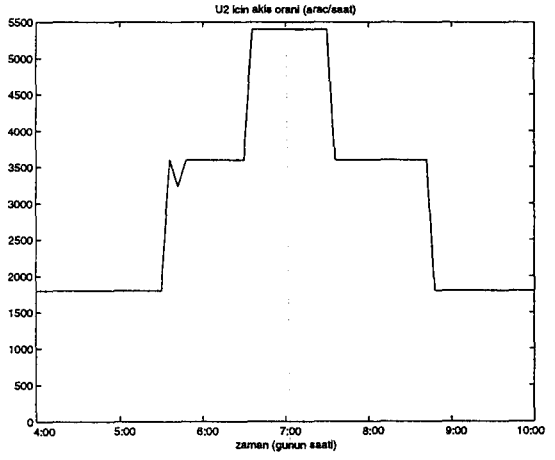
değişkenlerinin zamana bağlı olarak grafikleri incelenmiş ve Şekil 5.20-5.61'de sunulmuştur. Bu grafikler incelendiğinde de görülebileceği gibi alternatif kontrolde benzetim süresince L2, L11, L17, L27 ve L29 linklerinde yaklaşık olarak saat 6:00-6:30 civarında sıkışıklık oluşmaya başlamış ve bundan sonra linklerdeki yoğunluk değerlerinin alternatif kontrolde önerilen kontrole oranla çok daha büyük değerlere ulaşmış ve akış oranları ile ortalama hız değerleri düşmüştür. Bu sonuçlara dayanılarak, bu örnekte önerilen kontrol yaklaşımının alternatif yaklaşıma göre trafik sıkışıklığını önemli ölçüde önlediği söylenebilir.



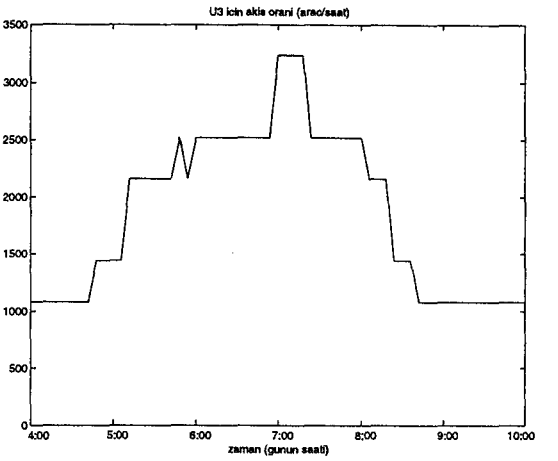
Şekil 5.2. Birinci benzetim için trafik ağı



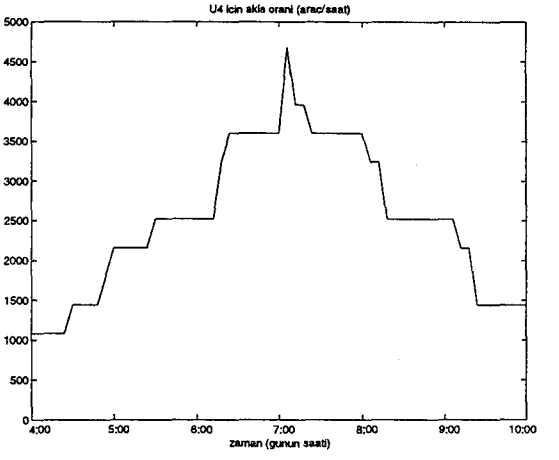
Şekil 5.3. U1 için akış oranı



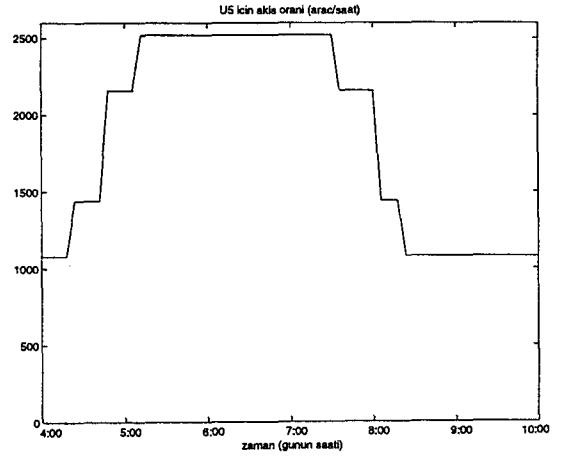
Şekil 5.4. U2 için akış oranı



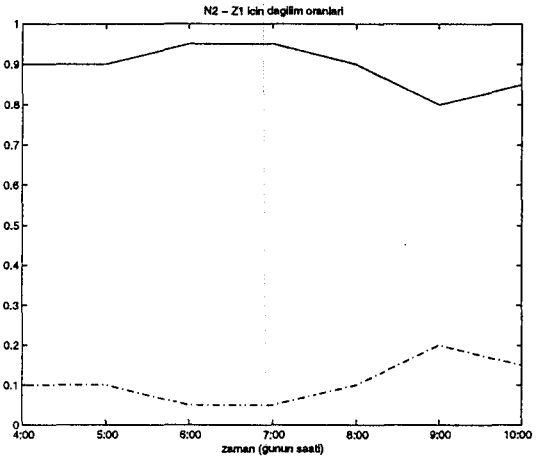
Şekil 5.5. U3 için akış oranı



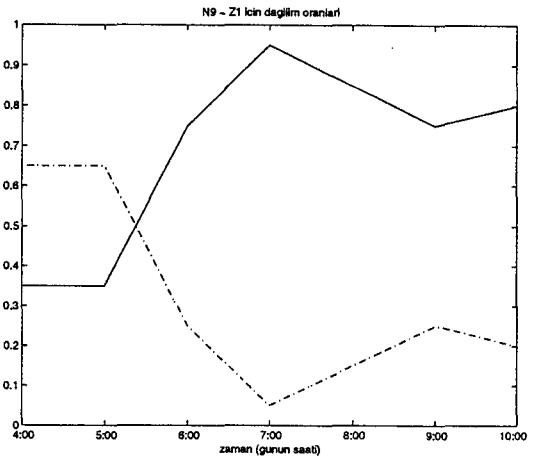
Şekil 5.6. U4 için akış oranı



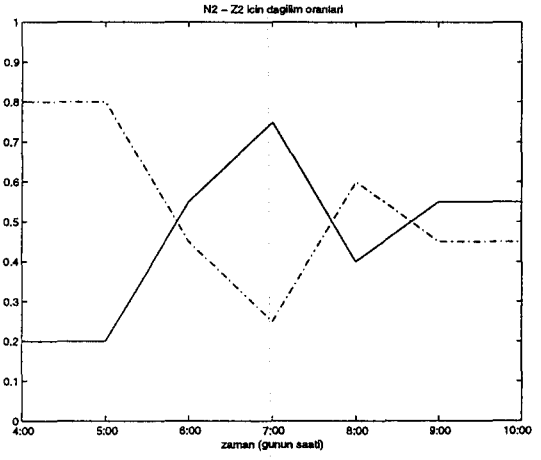
Şekil 5.7. U5 için akış oranı



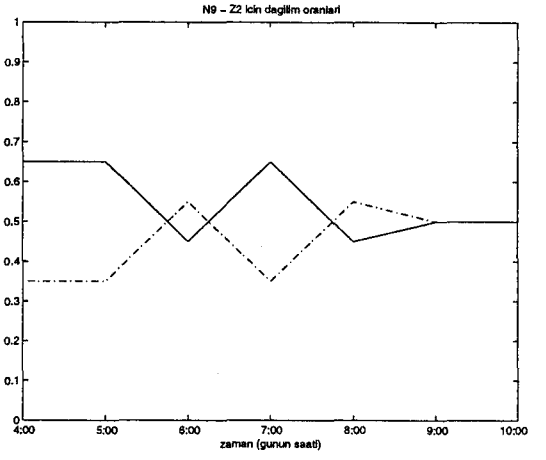
Şekil 5.8. $\phi^{Z1}_{N2,L3}$ (–), $\phi^{Z1}_{N2,L27}$ (-.)



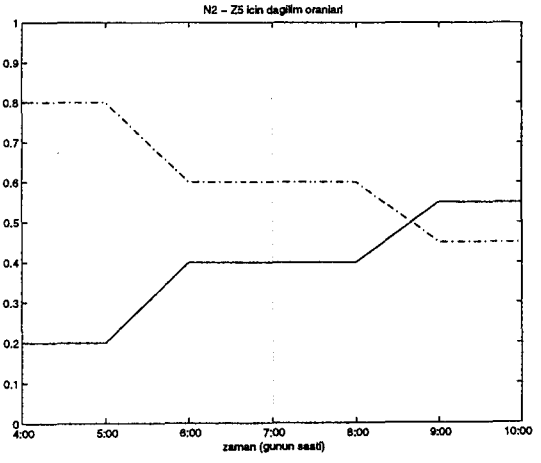
Şekil 5.11. $\phi^{Z1}_{N9,L28}$ (–), $\phi^{Z1}_{N9,L12}$ (-.)



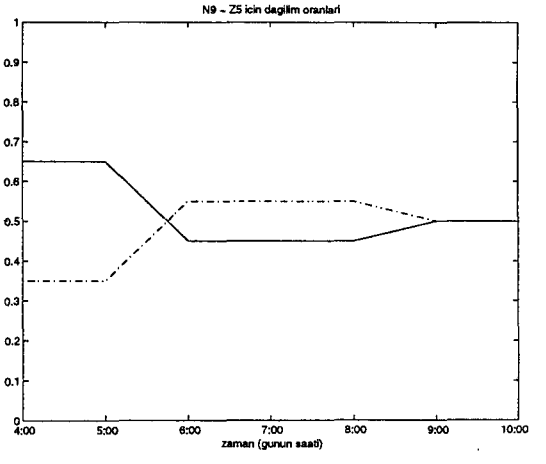
Şekil 5.9. $\phi^{Z2}_{N2,L3}$ (–), $\phi^{Z2}_{N2,L27}$ (-.)



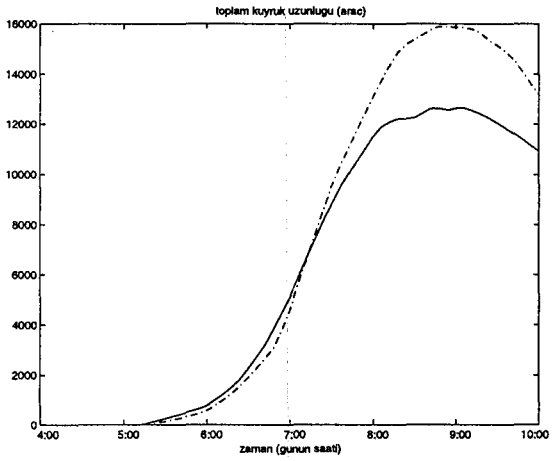
Şekil 5.12. $\phi^{Z2}_{N9,L28}$ (–), $\phi^{Z2}_{N9,L12}$ (-.)



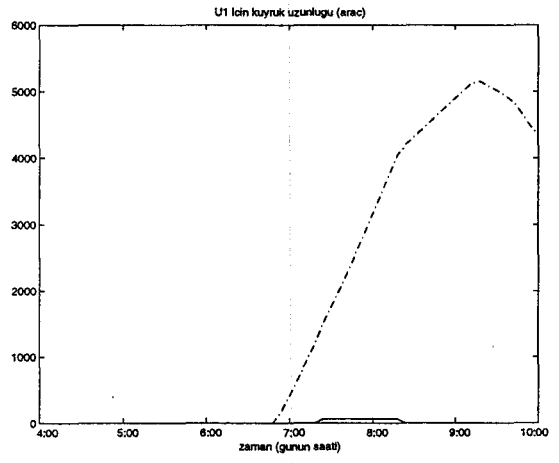
Şekil 5.10. $\phi^{Z5}_{N2,L3}$ (–), $\phi^{Z5}_{N2,L27}$ (-.)



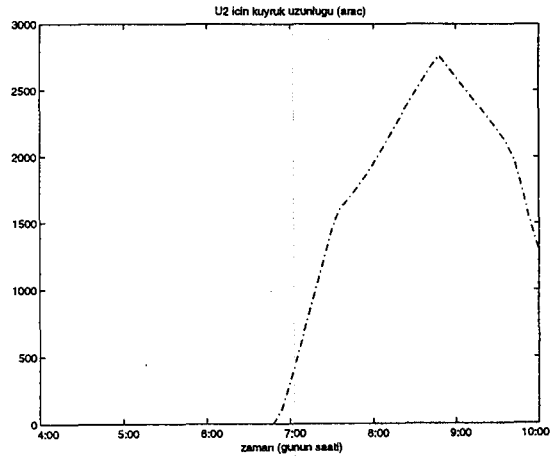
Şekil 5.13. $\phi^{Z5}_{N9,L28}$ (–), $\phi^{Z5}_{N9,L12}$ (-.)



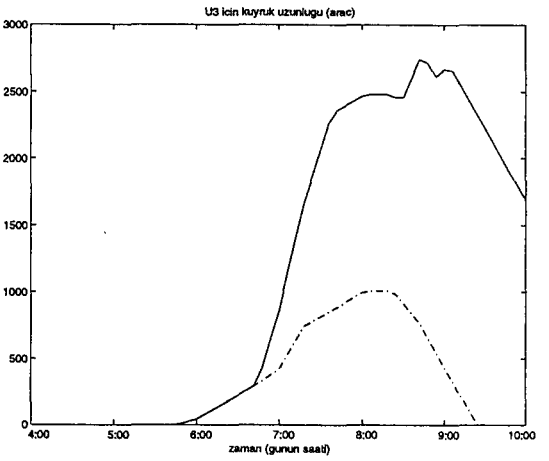
Şekil 5.14. Toplam kuyruk uzunluğu



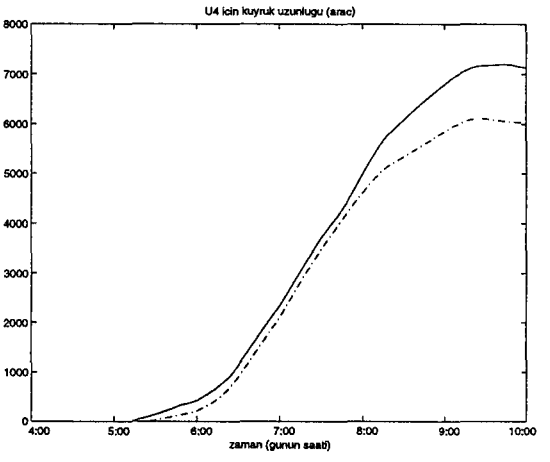
Şekil 5.15. U1 için kuyruk uzunluğu



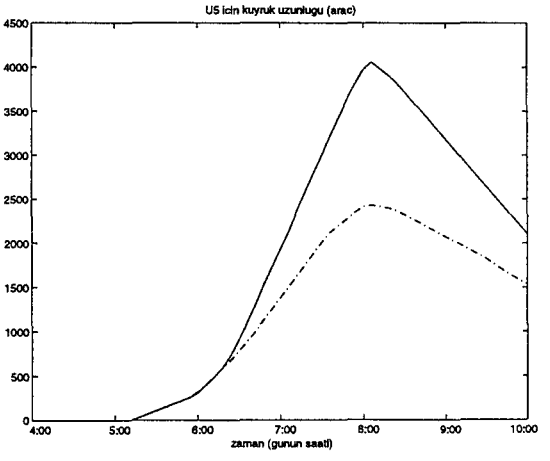
Şekil 5.16. U2 için kuyruk uzunluğu



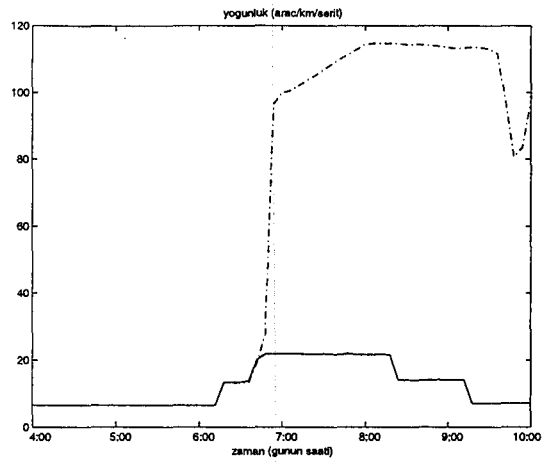
Şekil 5.17. U3 için kuyruk uzunluğu



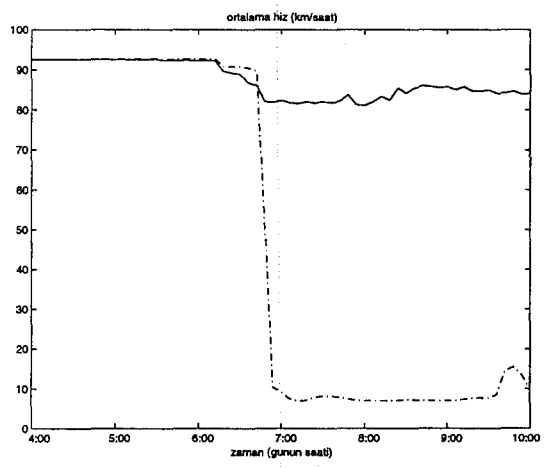
Şekil 5.18. U4 için kuyruk uzunluğu



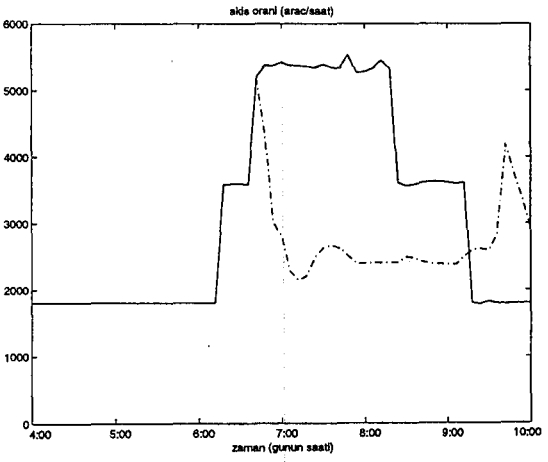
Şekil 5.19. U5 için kuyruk uzunluğu



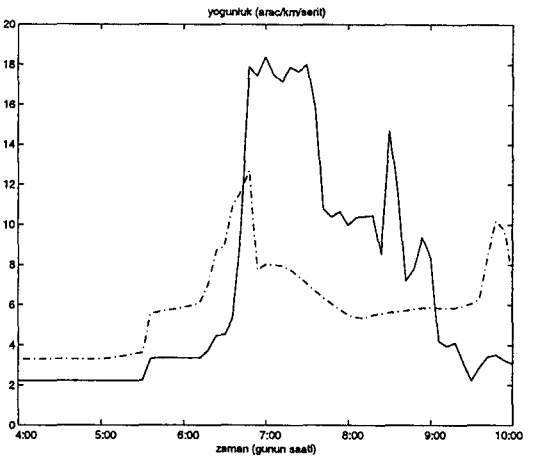
Şekil 5.20. L2'deki yoğunluk



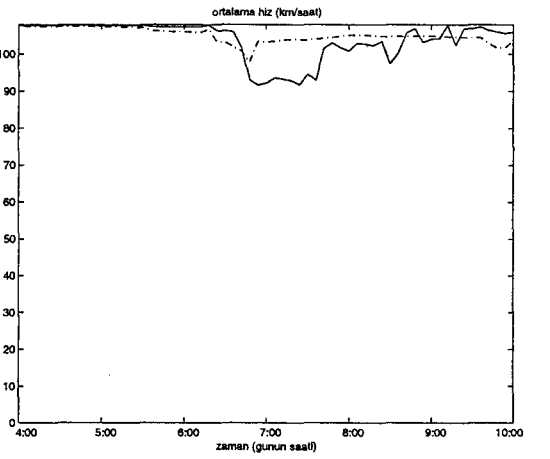
Şekil 5.21. L2'deki ortalama hız



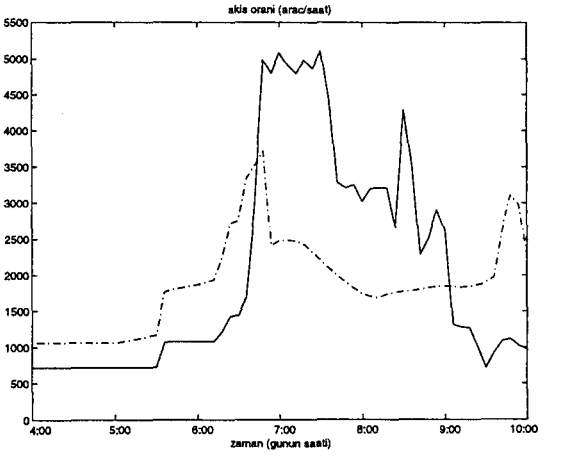
Şekil 5.22. L2'deki akış oranı



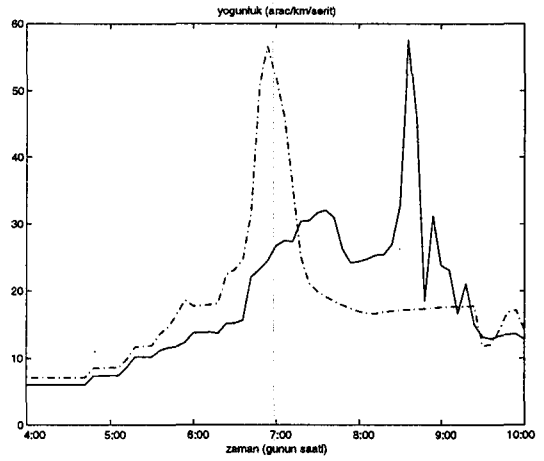
Şekil 5.23. L4'deki yoğunluk



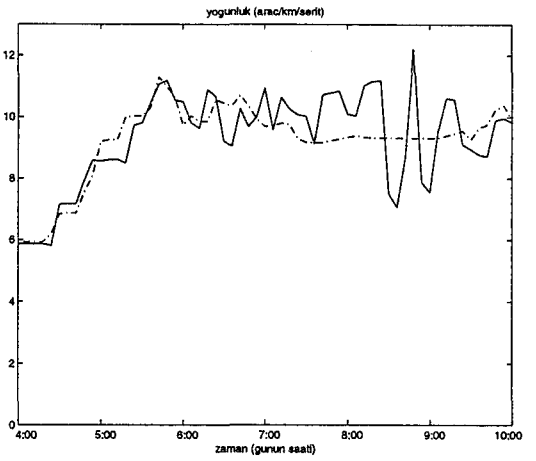
Şekil 5.24. L4'deki ortalama hız



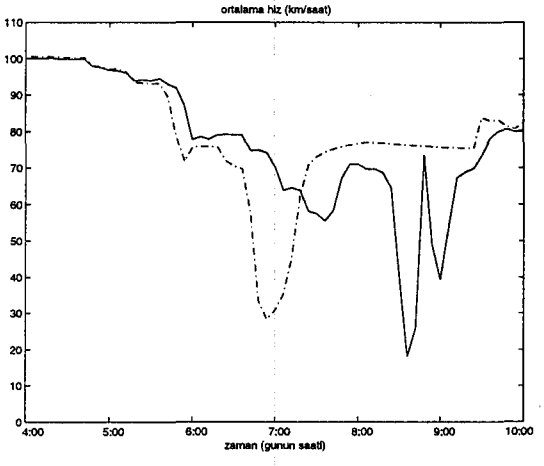
Şekil 5.25. L4'deki akış oranı



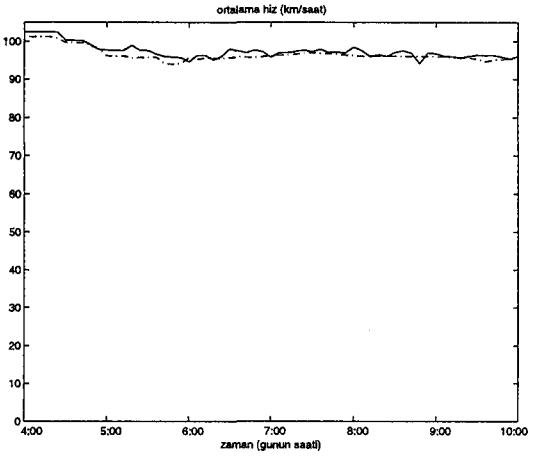
Şekil 5.26. L6'daki yoğunluk



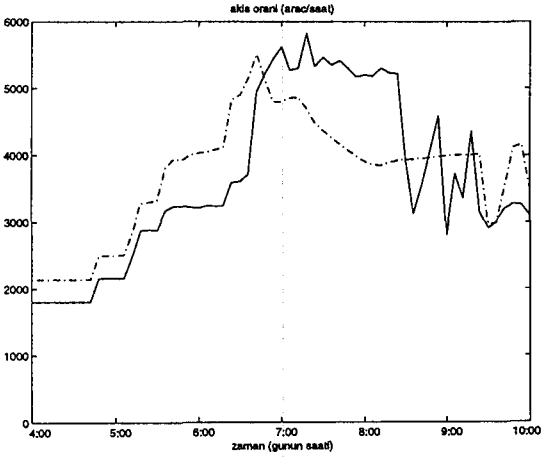
Şekil 5.29. L8'deki yoğunluk



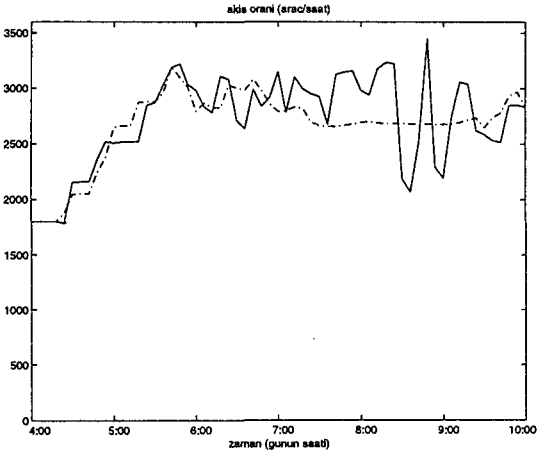
Şekil 5.27. L6'daki ortalama hız



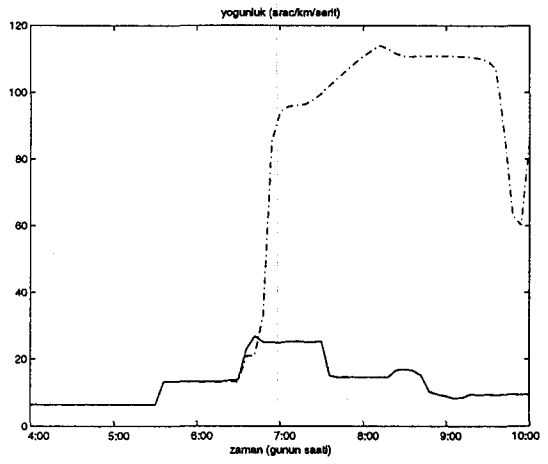
Şekil 5.30. L8'deki ortalama hız



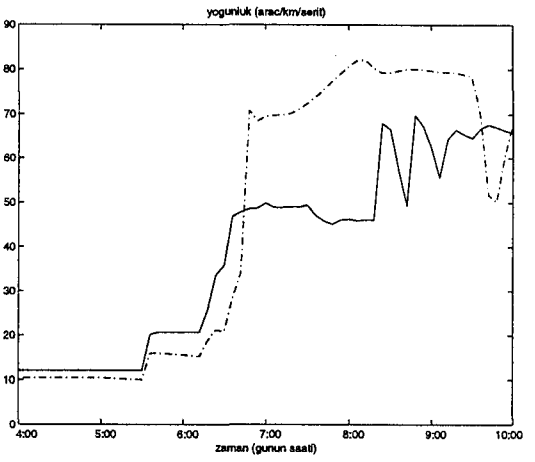
Şekil 5.28. L6'daki akış oranı



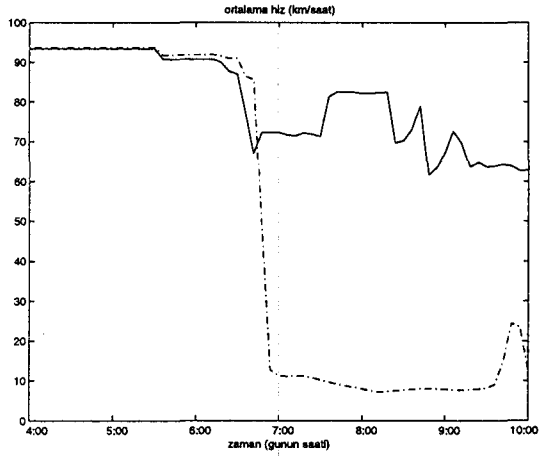
Şekil 5.31. L8'deki akış oranı



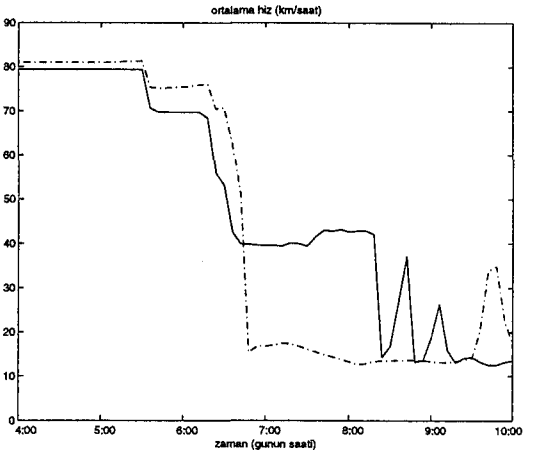
Şekil 5.32. L11'deki yoğunluk



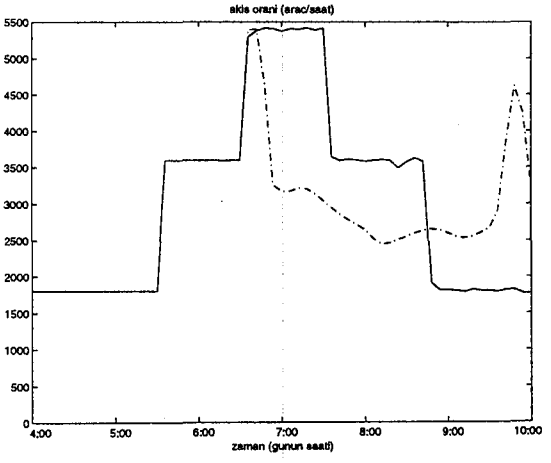
Şekil 5.35. L13'deki yoğunluk



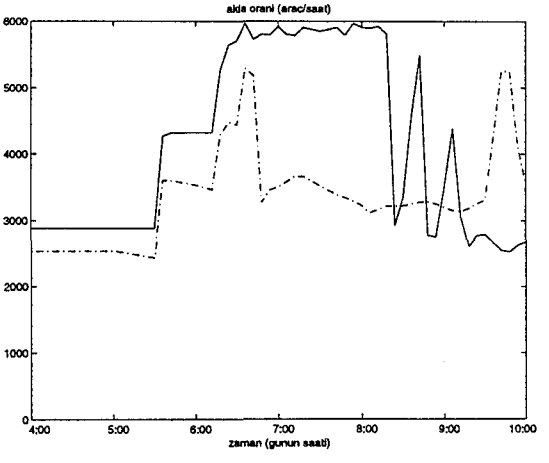
Şekil 5.33. L11'deki ortalama hız



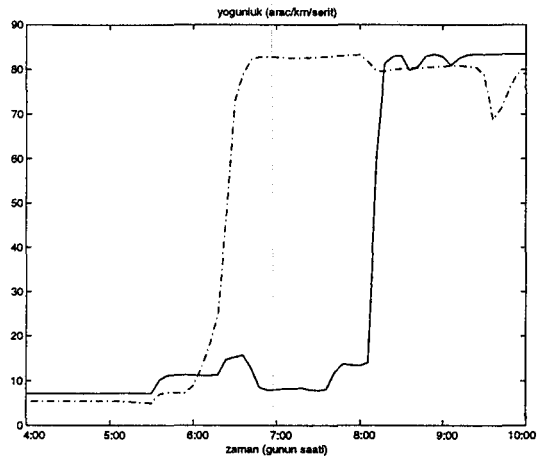
Şekil 5.36. L13'deki ortalama hız



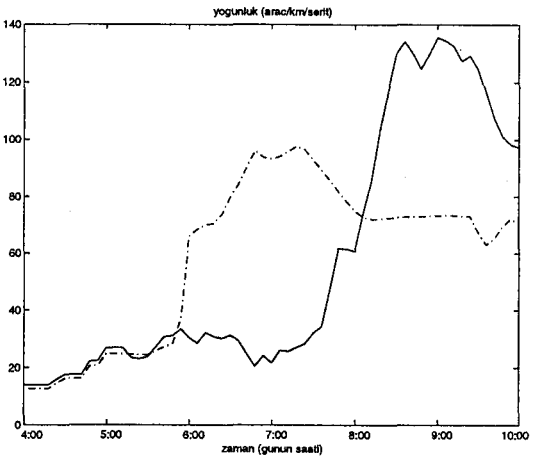
Şekil 5.34. L11'deki akış oranı



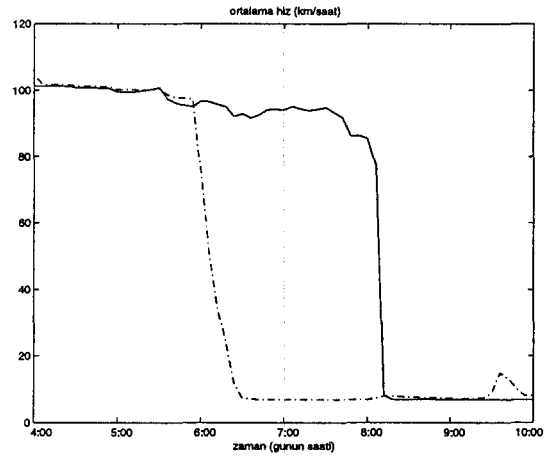
Şekil 5.37. L13'deki akış oranı



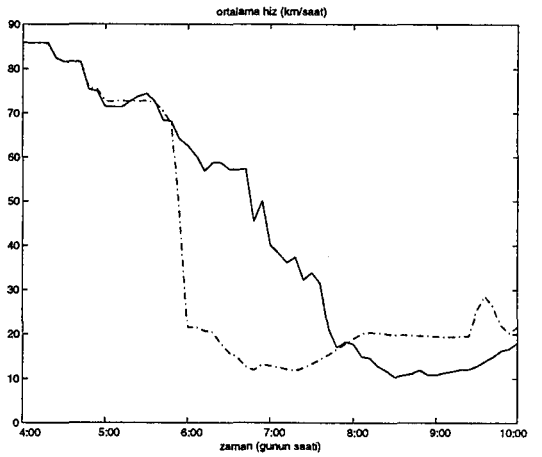
Şekil 3.38. L17'deki yoğunluk



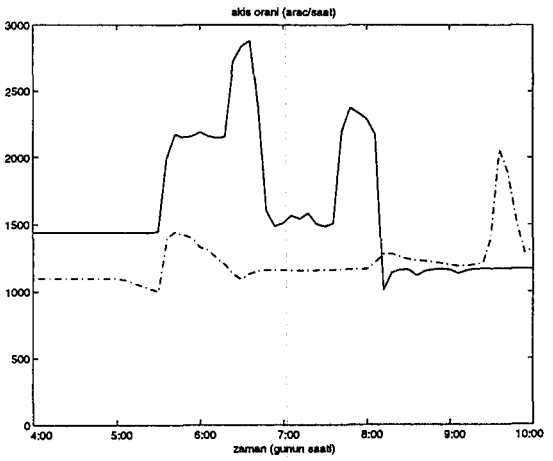
Şekil 5.41. L21'deki yoğunluk



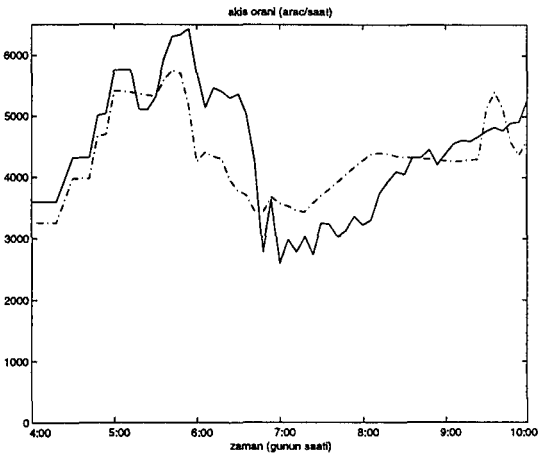
Şekil 5.39. L17'deki ortalama hız



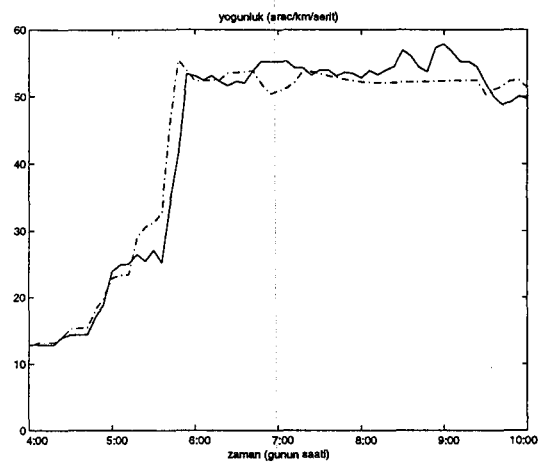
Şekil 5.42. L21'deki ortalama hız



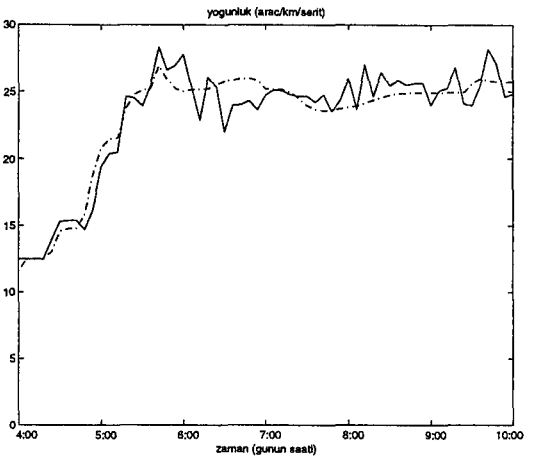
Şekil 5.40. L17'deki akış oranı



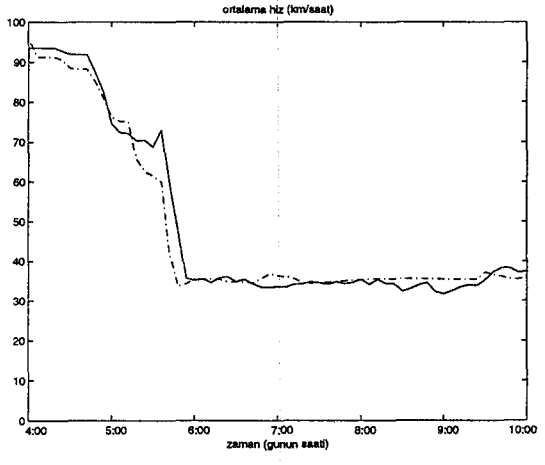
Şekil 5.43. L21'deki akış oranı



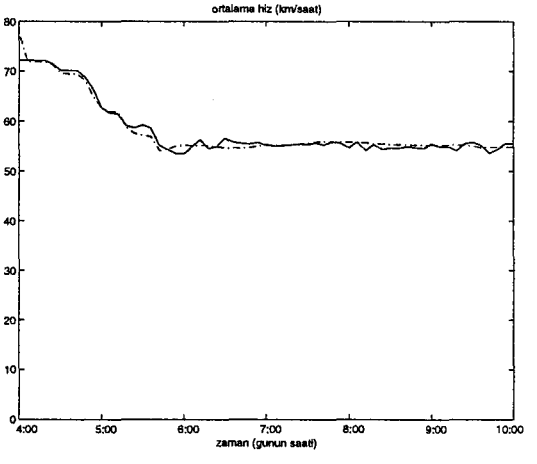
Şekil 5.44. L23'deki yoğunluk



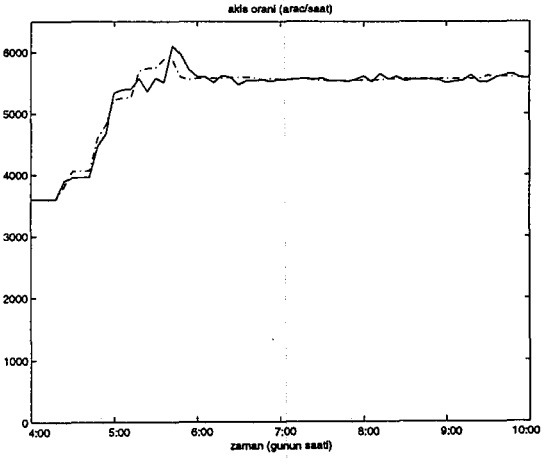
Şekil 5.47. L25'deki yoğunluk



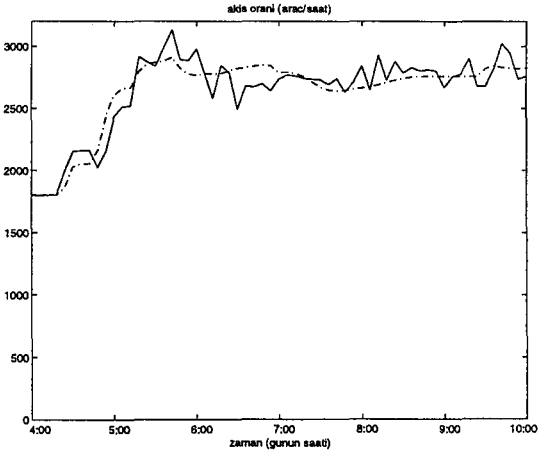
Şekil 5.45. L23'deki ortalama hız



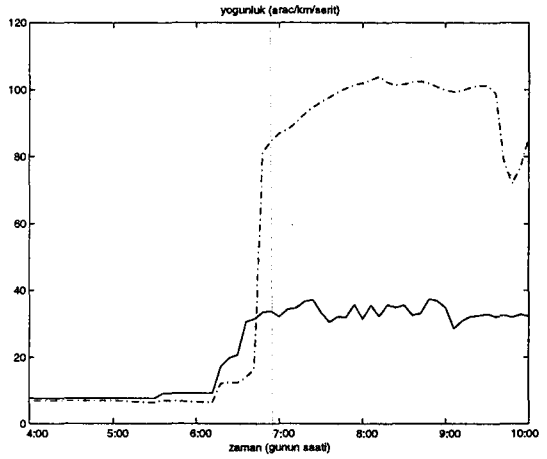
Şekil 5.48. L25'deki ortalama hız



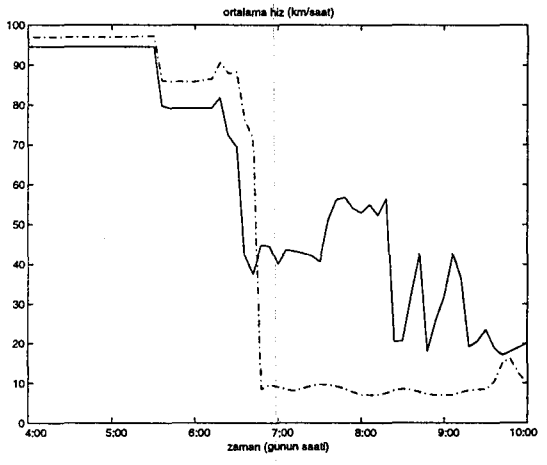
Şekil 5.46. L23'deki akış oranı



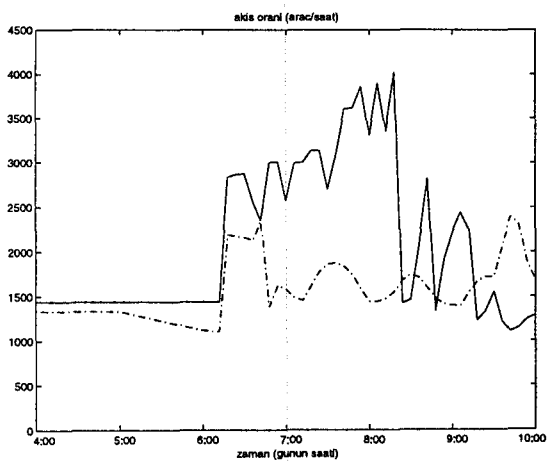
Şekil 5.49. L25'deki akış oranı



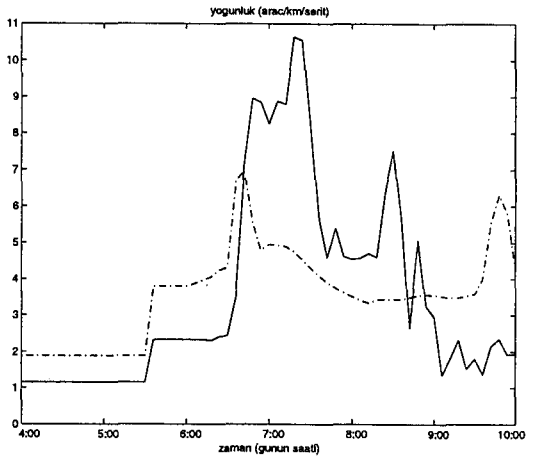
Şekil 5.50. L27'deki yoğunluk



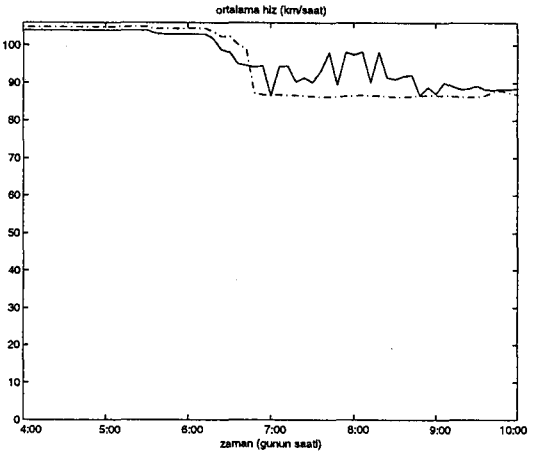
Şekil 5.51. L27'deki ortalama hız



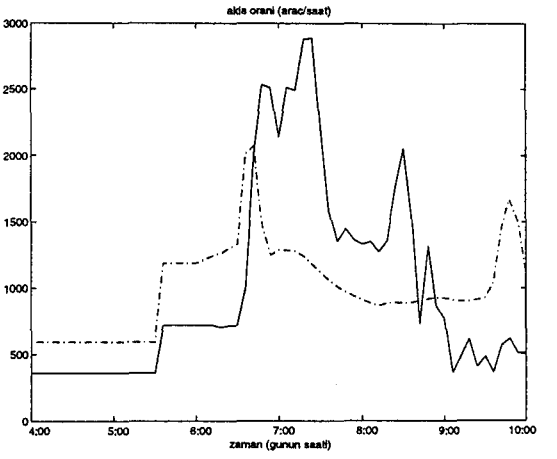
Şekil 5.52. L27'deki akış oranı



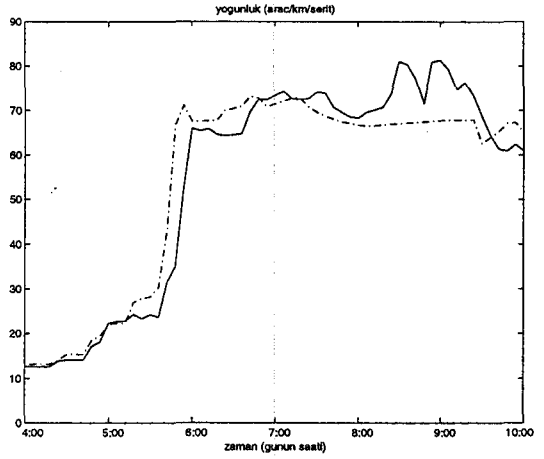
Şekil 5.53. L28'deki yoğunluk



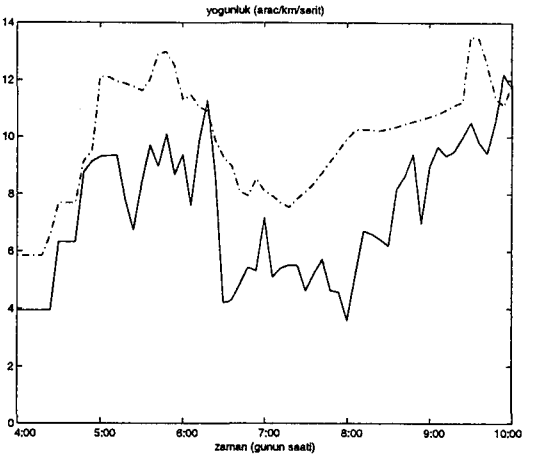
Şekil 5.54. L28'deki ortalama hız



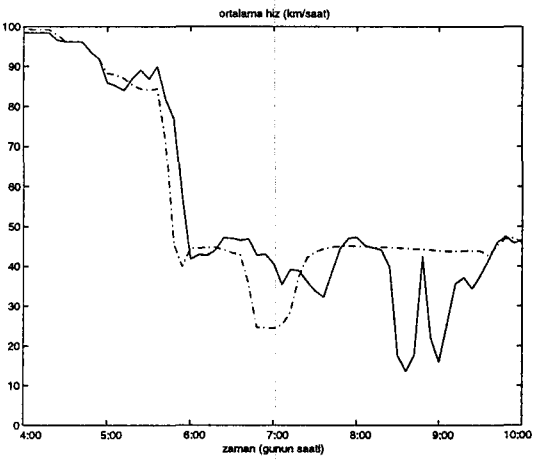
Şekil 5.55. L28'deki akış oranı



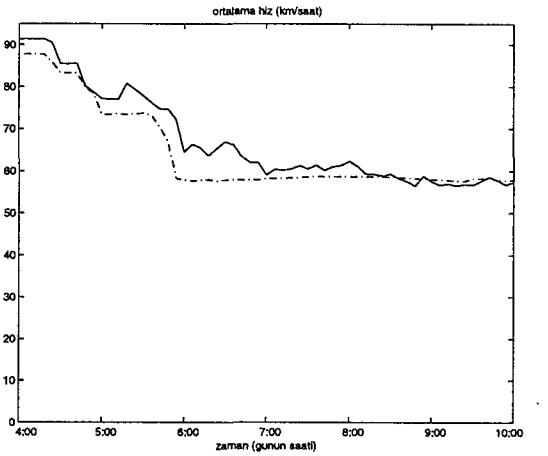
Şekil 5.56. L29'daki yoğunluk



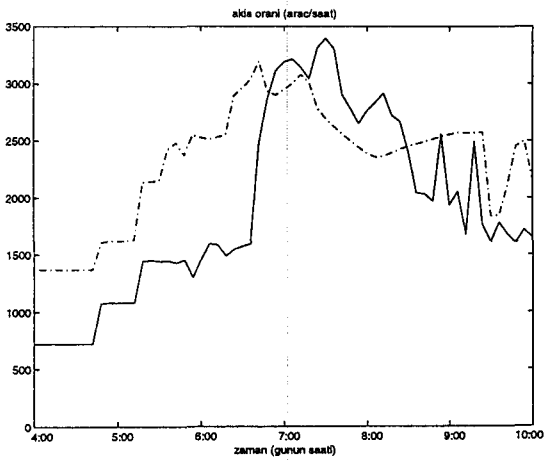
Şekil 5.59. L31'deki yoğunluk



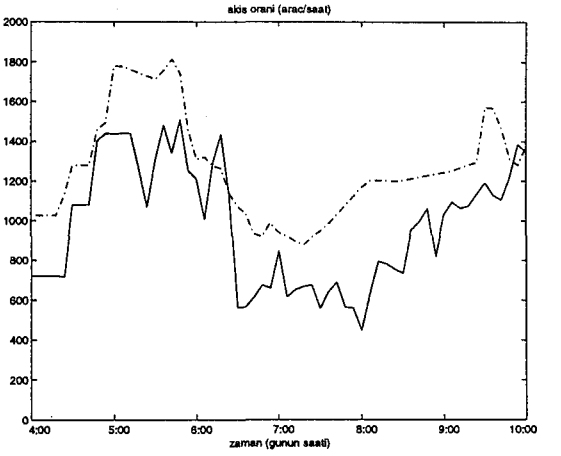
Şekil 5.57. L29'daki ortalama hız



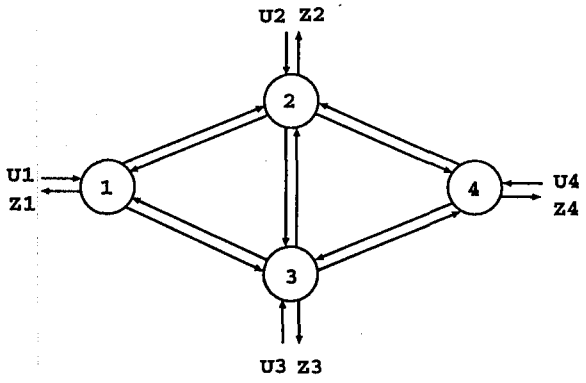
Şekil 5.60. L31'deki ortalama hız



Şekil 5.58. L29'daki akış oranı



Şekil 5.61. L31'deki akış oranı



Şekil 5.62. İkinci örnek trafik ağı

5.2.2 İkinci örnek trafik ağı için benzetim çalışması:

Farklı bir benzetim çalışması için Şekil 5.62'de görülen ağ ele alınmıştır. Bu ağda, bir öncekinden farklı olarak, her bir giriş noktasından her bir varış noktasına ulaşılabilinmektedir. Ayrıca, nodlar arası linklerin bağlantıları da araçların ağa katılmasından sonra varış noduna ulaşıncaya kadar döngüye girme ihtimalini doğuracak şekilde seçilmiştir. Böylece önerilen kontrolün araçların döngüye girmesini ne oranda etkilediği gözlemlenmek istenmiştir. Hazırlanan benzetim programının bu ağa hizmet edebilmesini sağlamak amacıyla sanal nodlar eklenerek Şekil 5.63'de sunulan ağ modeli elde edilmiştir. Elde edilen bu ağ modelinde 16 nod ve 4 giriş (U1,...,U4), 4 çıkış (Z1,...,Z4) linki de dahil olmak üzere toplam 31 link bulunmaktadır.

Bu trafik ağına 04:00 ile 10:00 saatleri arasında 6 saat boyunca araç girişinin yapıldığının kabul edildiği bir durum için benzetim yapılmıştır. Bu benzetim çalışmasına ait tüm veri giriş dosyaları (ag2.*) Ek-5'de sunulmuştur. Örnekleme periyodunun 10 s. olarak alındığı bu çalışmada, benzetim süresince ağdaki 4 giriş bölümünden (U1,...,U4) ağa katılmak isteyen araç miktarlarının zamana bağlı olarak grafikleri sırasıyla Şekil 5.64-5.67'de verilmiştir. *ag2.MSD* dosyasında girilen bu değerlerin varış nodlarına göre dağılımını veren grafikler de Şekil 5.68-5.79'da

görülmektedir.

Aynı ağa, aynı koşullar altında, Bölüm 5.1'de tanıtılan alternatif yaklaşım da uygulanmıştır. Bu yaklaşım için seçilmiş olan $\phi^{l}_{n,m}(k)$ dağılım oranlarının değerleri aşağıda verilmiştir (sadece sıfır'dan farklı olanlar verilmiştir):

$$\phi^{Z1}_{N2,L5} = \phi^{Z4}_{N2,L5} = 0.5; \quad \phi^{Z2}_{N2,L5} = 1.0;$$

$$\phi^{Z1}_{N2,L14} = \phi^{Z4}_{N2,L14} = 0.5; \quad \phi^{Z3}_{N2,L14} = 1.0;$$

$$\phi^{Z1}_{N5,L8} = \phi^{Z2}_{N5,L8} = \phi^{Z4}_{N5,L8} = 1.0; \quad \phi^{Z3}_{N5,L24} = 1.0;$$

$$\phi^{Z1}_{N6,L9} = \phi^{Z2}_{N6,L9} = \phi^{Z3}_{N6,L9} = 1.0; \quad \phi^{Z4}_{N6,L25} = 1.0;$$

$$\phi^{Z2}_{N8,L12} = \phi^{Z3}_{N8,L12} = \phi^{Z4}_{N8,L12} = 1.0; \quad \phi^{Z1}_{N8,L13} = 1.0;$$

$$\phi^{Z1}_{N10,L17} = \phi^{Z3}_{N10,L17} = \phi^{Z4}_{N10,L17} = 1.0; \quad \phi^{Z2}_{N10,L23} = 1.0;$$

$$\phi^{Z2}_{N11,L18} = \phi^{Z3}_{N11,L18} = \phi^{Z4}_{N11,L18} = 1.0; \quad \phi^{Z1}_{N11,L22} = 1.0;$$

$$\phi^{Z1}_{N13,L21} = \phi^{Z2}_{N13,L21} = \phi^{Z3}_{N13,L21} = 1.0; \quad \phi^{Z4}_{N13,L26} = 1.0;$$

$$\phi^{Z1}_{N15,L27} = \phi^{Z4}_{N15,L27} = 0.5; \quad \phi^{Z2}_{N15,L27} = 1.0;$$

$$\phi^{Z1}_{N15,L28} = \phi^{Z4}_{N15,L28} = 0.5; \quad \phi^{Z3}_{N15,L28} = 1.0;$$

Yukarıda verilen dağılım oranları incelendiğinde görülebileceği gibi, bu değerler ağda araçların döngüye girmesini önleyecek şekilde seçilmiştir.

Her iki benzetim sonucunda elde edilen performans ölçütlerinin değerleri Tablo 5.2'de görülmektedir. Bu değerlerden görüldüğü gibi, önerilen kontrolde benzetim süresince ağa katılmasına izin verilen araçların sayısı, alternatif kontroldekine oranla %8 daha fazladır. Ağa daha fazla araç alınmış olmasına rağmen önerilen kontrolde toplam seyahat süresi %63 oranında daha düşüktür. Ayrıca, önerilen kontrolde benzetim süresince ağa katılan araçların %99'u varış noduna ulaşarak ağdan ayrılırken, alternatif kontrolde ise araçların %94'ü ağdan ayrılmış olup %6'sı benzetim süresi sona erdiğinde hala ağda bulunmaktadır. Bu da önerilen kontrolde araçların nodlardan akış aşağı linklere paylaşımı yapılırken linklerdeki sıkışıklık durumunun göz önüne alınıyor olmasının getirdiği avantajı göstermektedir. Alternatif kontrolde araçlar nodlara geldiğinde varış noduna ulaşmak üzere en kısa yoldan

gönderildiği halde linklerde belli zamanlarda yoğunluklar arttığı için trafik akışının hızı düşmekte; bunun sonucu olarak toplam seyahat süresi beklenenin üzerinde olmaktadır.

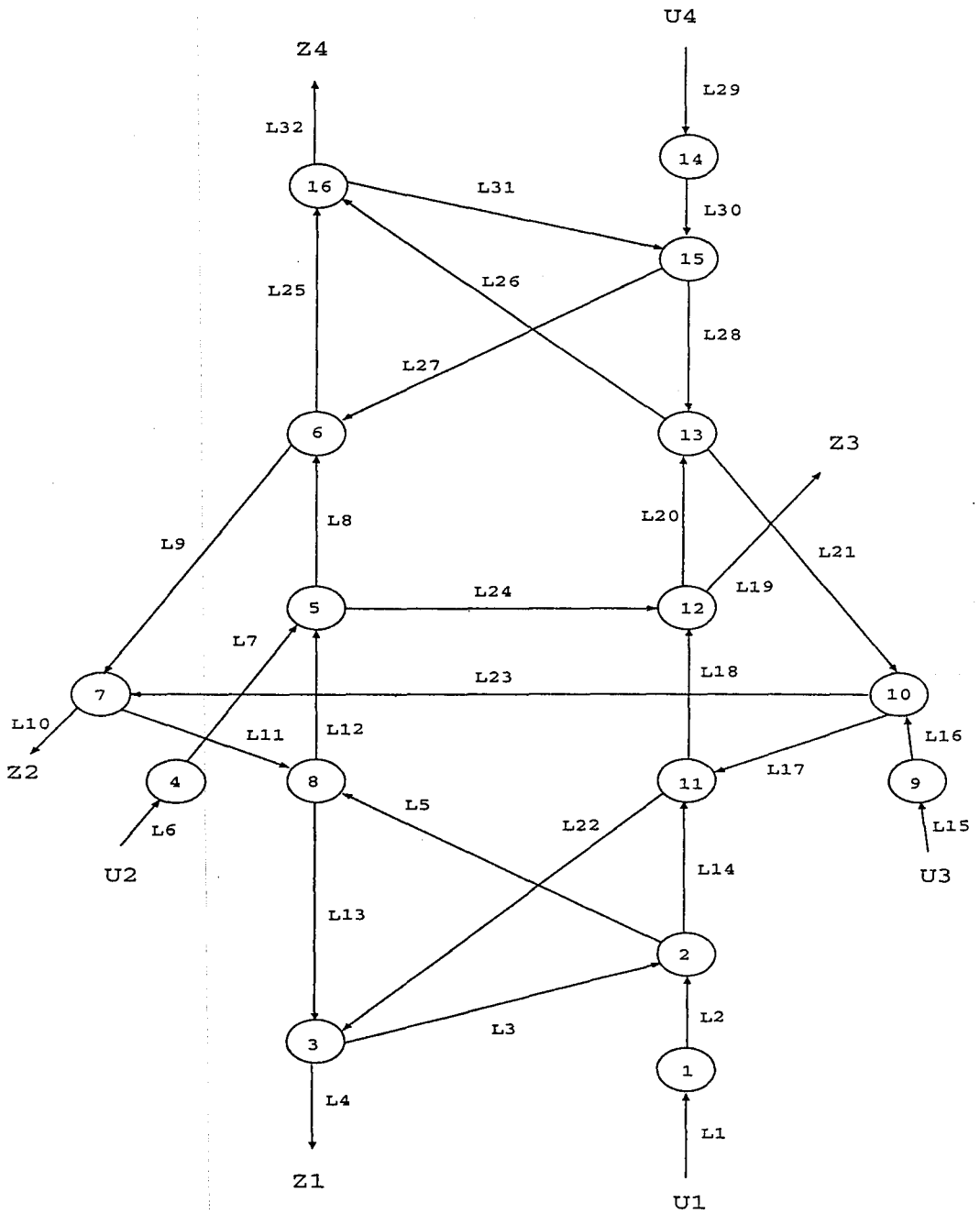
Ağa katılan araç başına seyahat süresi hesaplandığında, önerilen kontrolde 4.8 *dak.*, alternatif kontrolde ise 14.2 *dak.* olduğu görülmektedir. Bu da önerilen kontrolün alternatif kontrole oranla %66'lık bir zaman kazancı sağladığını göstermektedir. Ayrıca, önerilen kontrolde seyahat edilen toplam uzunluk alternatif kontrole oranla %45 daha fazladır. Seyahat edilen uzunluk için hesaplanmış olan bu orana rağmen, önerilen kontrolde harcanan toplam yakıt miktarı ancak %1 oranında fazladır. Seyahat edilen her 100 *km* başına önerilen kontrolde 8.4 *lt*, alternatif yaklaşımda ise 12.1 *lt* yakıt düşmekte; bu da önerilen yaklaşımın %31'lik bir yakıt tasarrufu sağladığını göstermektedir.

Her iki benzetim sonucunda elde edilmiş olan toplam kuyruk uzunluklarının grafikleri Şekil 5.80'de görülmektedir. Bu grafiklerden görülebileceği gibi önerilen kontrolün benzetimi sonunda oluşan toplam kuyruk uzunluğunun maksimum değeri alternatif kontrolünkinin %77'si kadardır. Ağdaki her bir giriş bölümünde oluşan kuyrukların zamana bağlı olarak grafikleri de Şekil 5.81-5.84'de sunulmuştur.

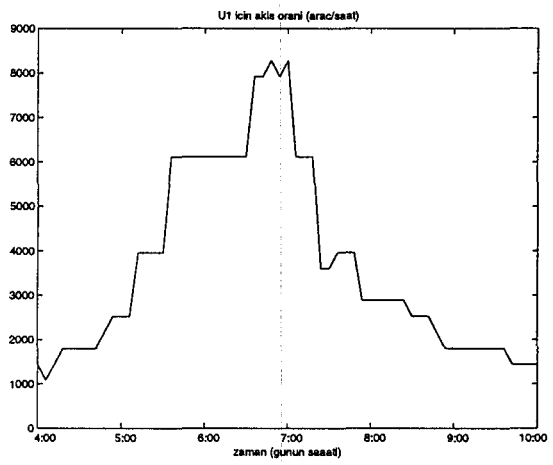
Ayrıca, ağdaki çeşitli linklerin hemen hemen orta bölümleri için trafik değişkenlerinin zamana bağlı olarak grafikleri Şekil 5.85-5.120'de verilmiştir. Alternatif kontrol için benzetim süresi içinde L5, L12, L14, L18, L27 ve L28 linklerinde yoğunluk değerlerinde büyük oranda artış olduğu gözlenmektedir. Bu da hızın düşmesine ve dolayısıyla linklerdeki araç akışının çok yavaş ilerlemesine sebep olmaktadır. Gerek bu sonuçlardan gerekse Tablo 5.2'de verilen performans değerlerinden de görülebileceği gibi, bu örnekte de, önerilen kontrol yaklaşımı alternatif yaklaşıma göre trafik sıkışıklığının önlenmesinde çok daha başarılı olmuştur.

	Önerilen Yaklaşım	Alternatif Yaklaşım
TSS (*saat*)	6392.43	17572.95
TBS (*saat*)	40725.29	55840.00
AKAS (*araç*)	79882	74041
AAAS (*araç*)	79364	69590
TSU (*km*)	521329.10	357344.00
HY (*litre*)	43746.40	43249.80
MKU(U1) (*araç*)	8856	4078
MKU(U2) (*araç*)	663	5561
MKU(U3) (*araç*)	3335	8279
MKU(U4) (*araç*)	3336	2261

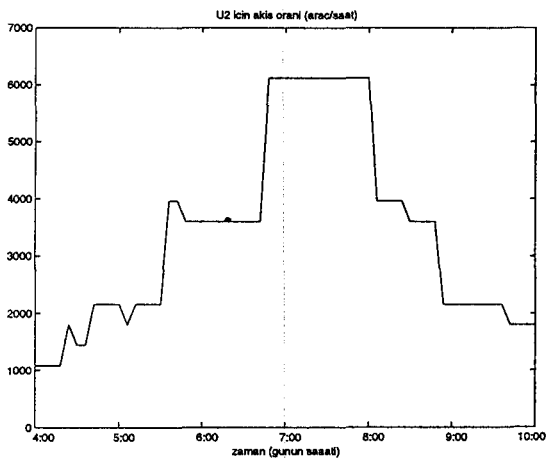
Tablo 5.2. Performans ölçütlerinin değerleri



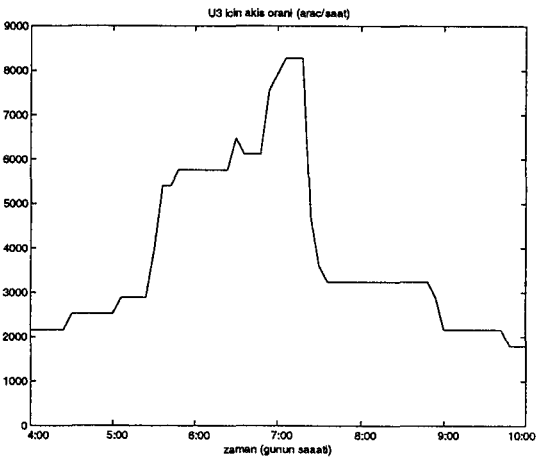
Şekil 5.63. İkinci benzetim için trafik ağı



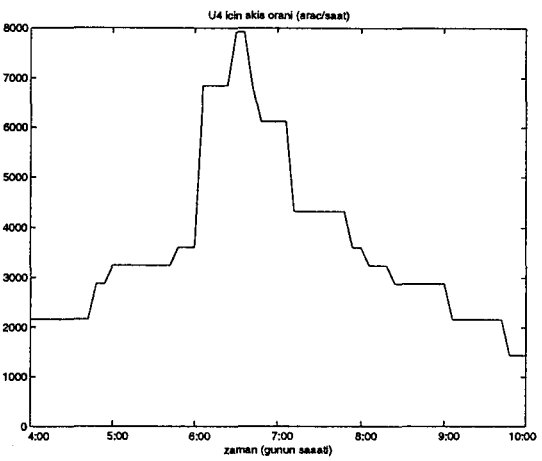
Şekil 5.64. U1 için akış oranı



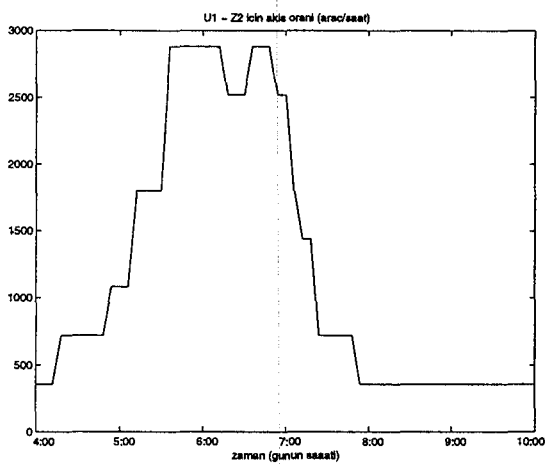
Şekil 5.65. U2 için akış oranı



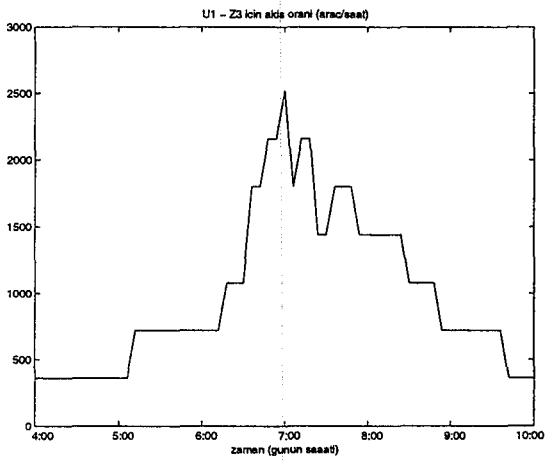
Şekil 5.66. U3 için akış oranı



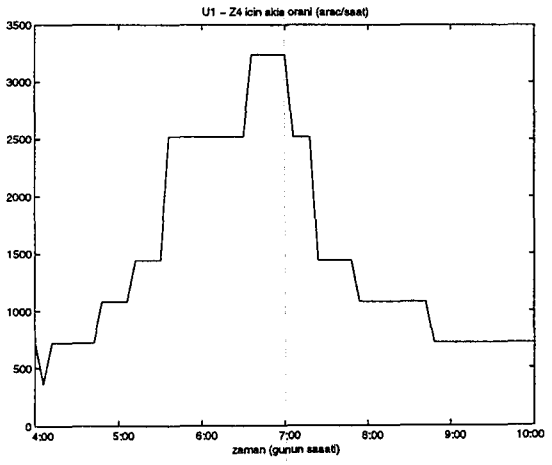
Şekil 5.67. U4 için akış oranı



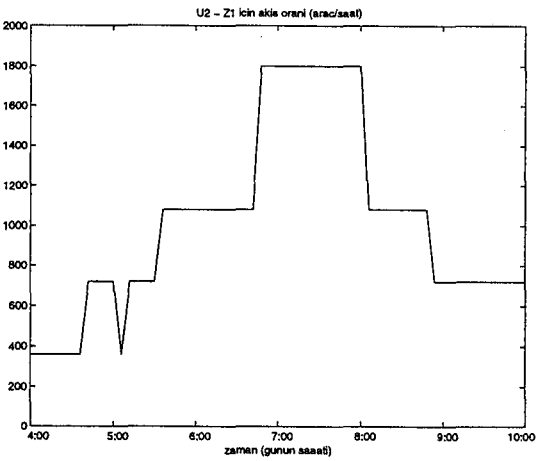
Şekil 5.68. U1 - Z2 için akış oranı



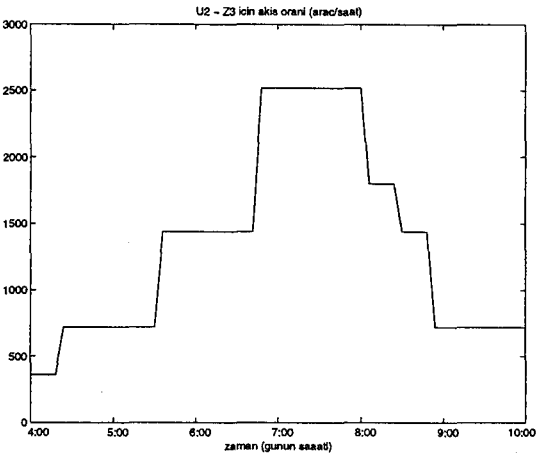
Şekil 5.69. U1 - Z3 için akış oranı



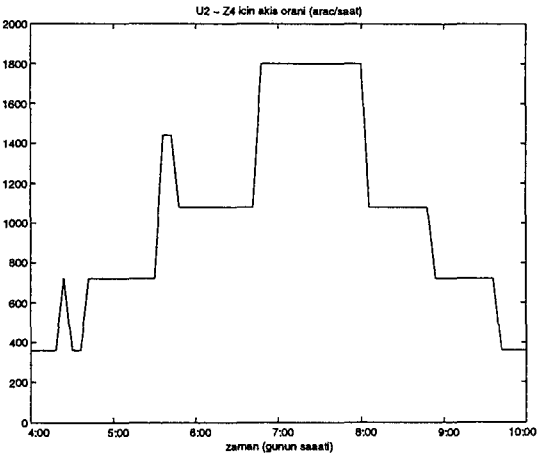
Şekil 5.70. U1 - Z4 için akış oranı



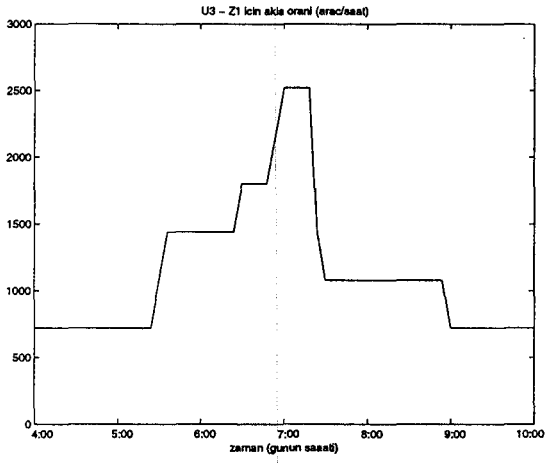
Şekil 5.71. U2 - Z1 için akış oranı



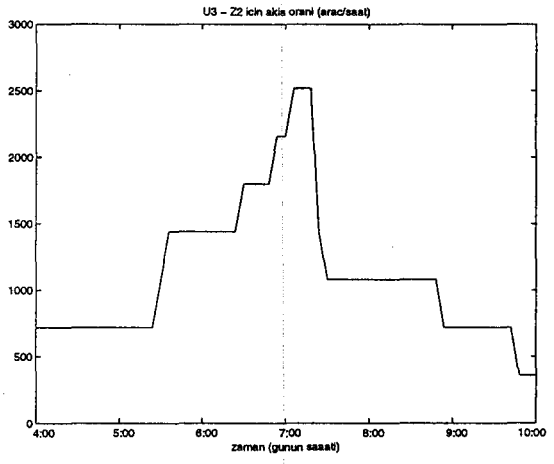
Şekil 5.72. U2 - Z3 için akış oranı



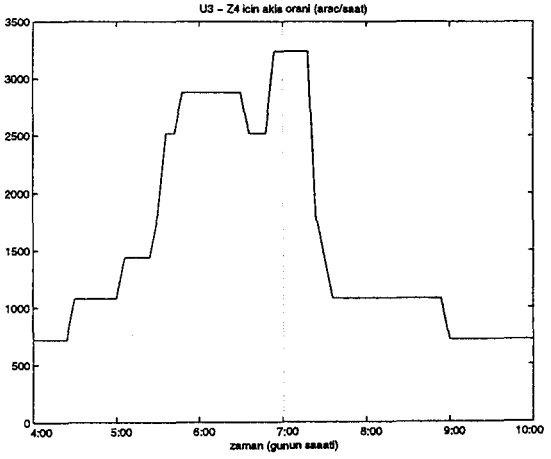
Şekil 5.73. U2 - Z4 için akış oranı



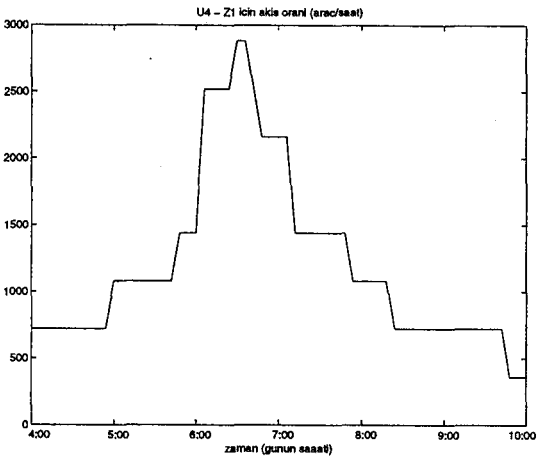
Şekil 5.74. U3 - Z1 için akış oranı



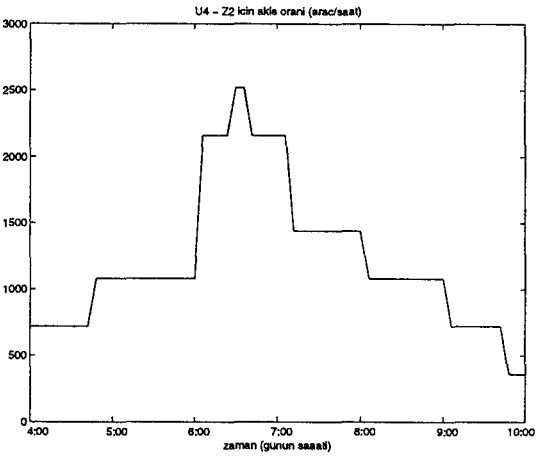
Şekil 5.75. U3 - Z2 için akış oranı



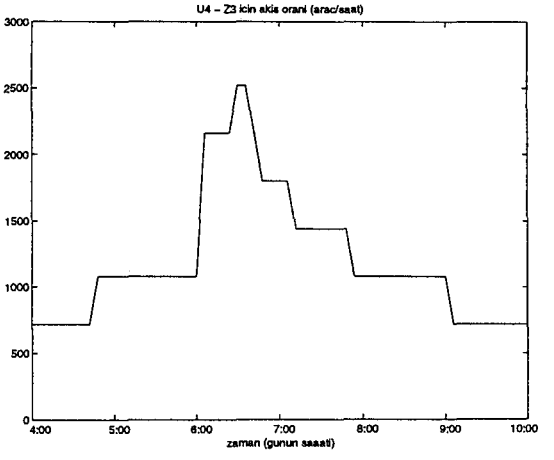
Şekil 5.76. U3 - Z4 için akış oranı



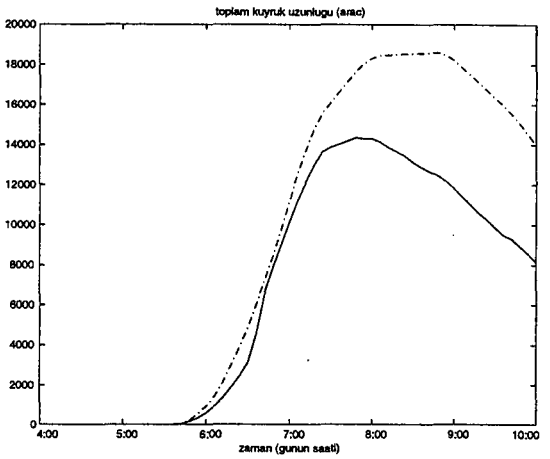
Şekil 5.77. U4 - Z1 için akış oranı



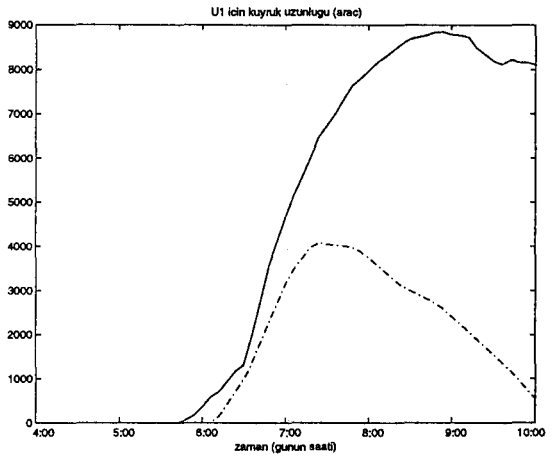
Şekil 5.78. U4 - Z2 için akış oranı



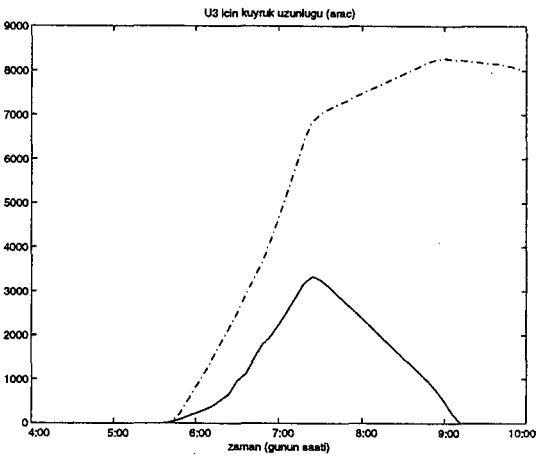
Şekil 5.79. U4 - Z3 için akış oranı



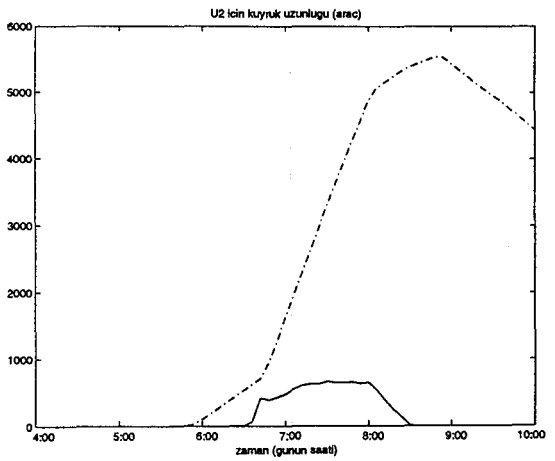
Şekil 5.80. Toplam kuyruk uzunluğu



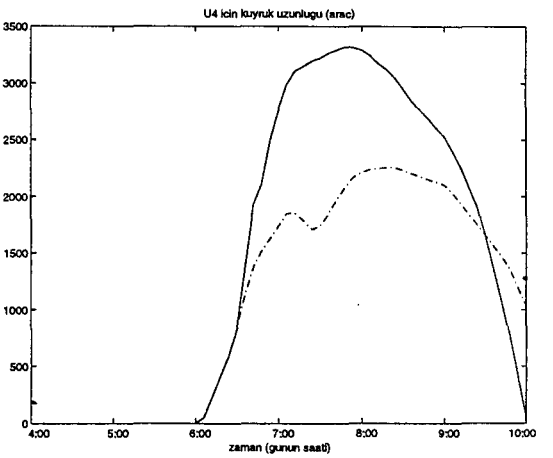
Şekil 5.81. U1 için kuyruk uzunluğu



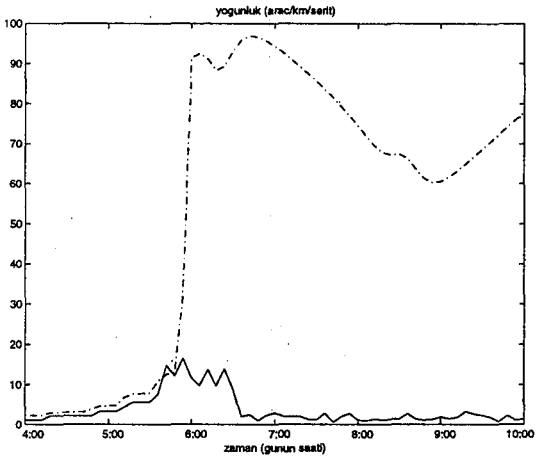
Şekil 5.83. U3 için kuyruk uzunluğu



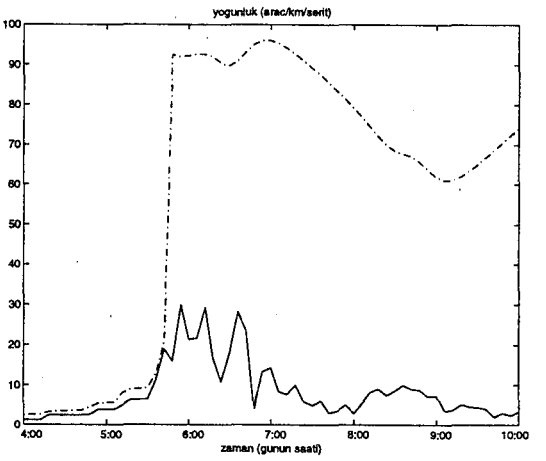
Şekil 5.82. U2 için kuyruk uzunluğu



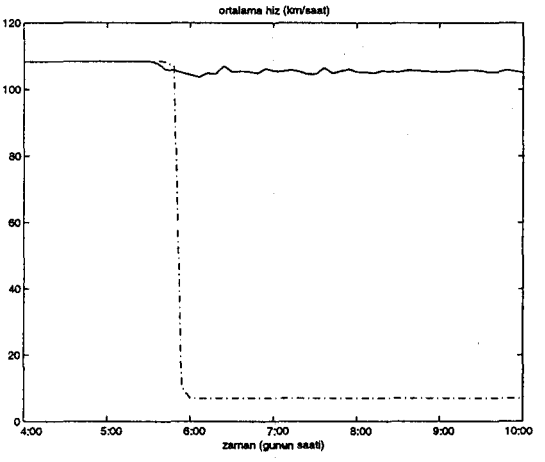
Şekil 5.84. U4 için kuyruk uzunluğu



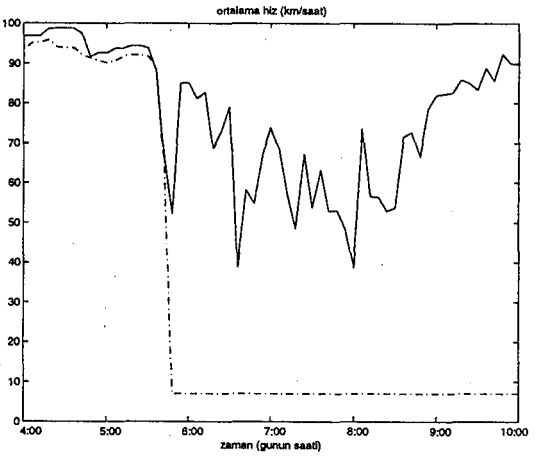
Şekil 5.85. L5'deki yoğunluk



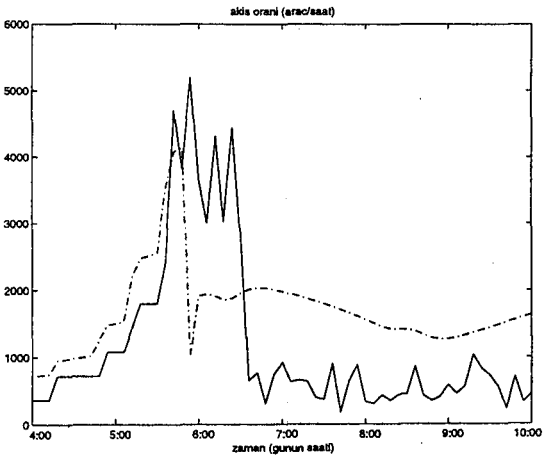
Şekil 5.88. L12'deki yoğunluk



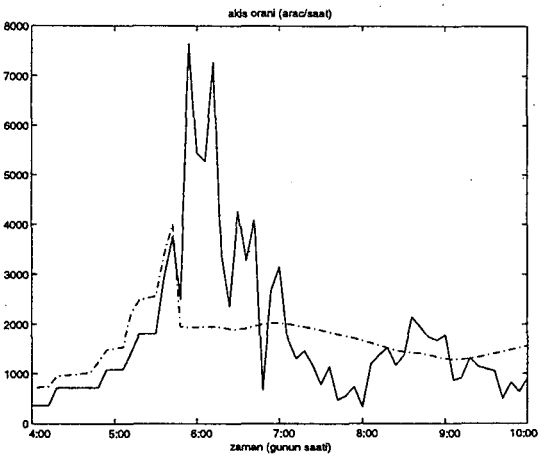
Şekil 5.86. L5'deki ortalama hız



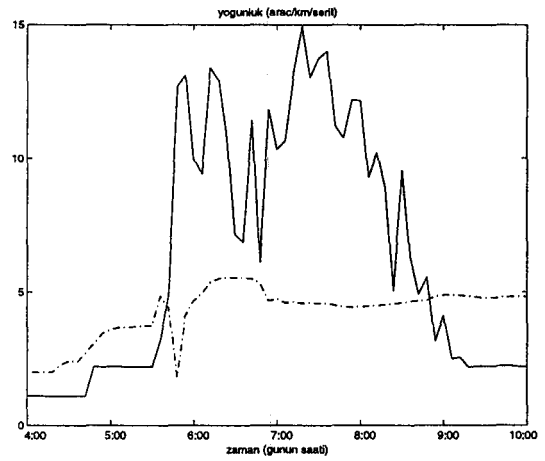
Şekil 5.89. L12'deki ortalama hız



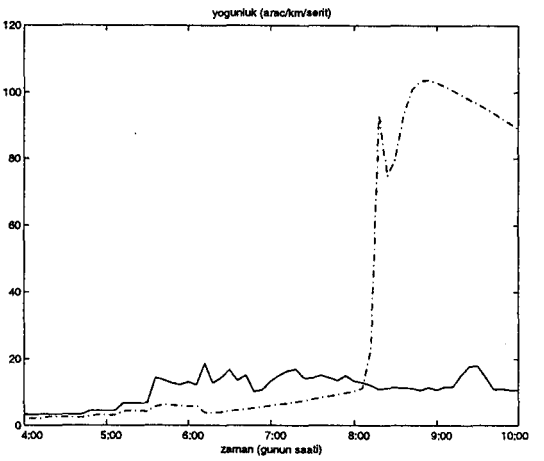
Şekil 5.87. L5'deki akış oranı



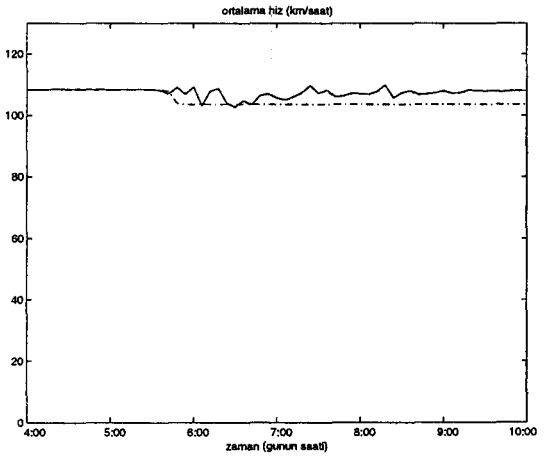
Şekil 5.90. L12'deki akış oranı



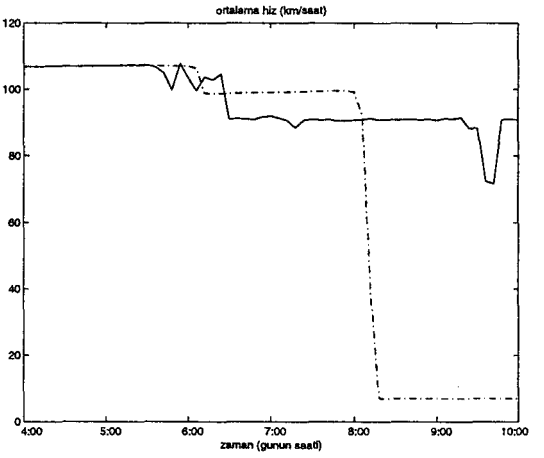
Şekil 5.91. L13'deki yoğunluk



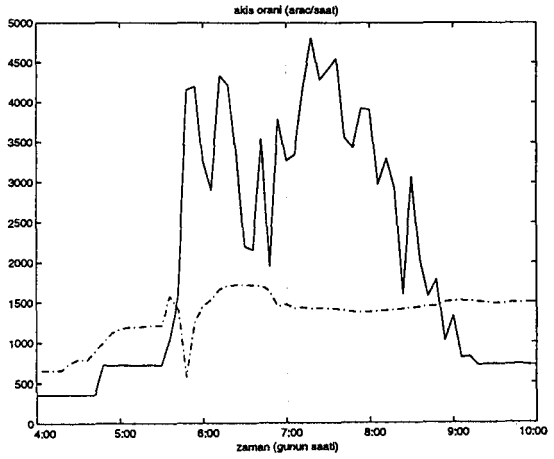
Şekil 5.94. L14'deki yoğunluk



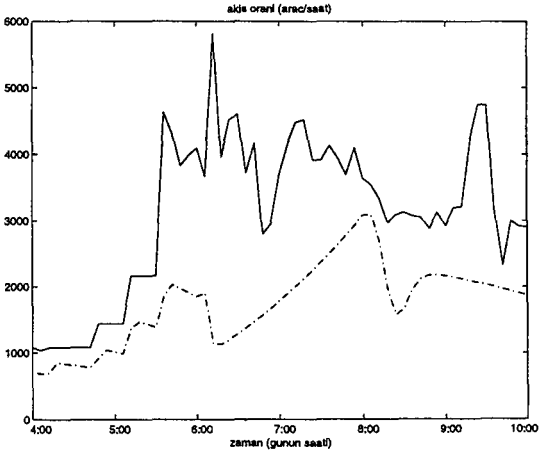
Şekil 5.92. L13'deki ortalama hız



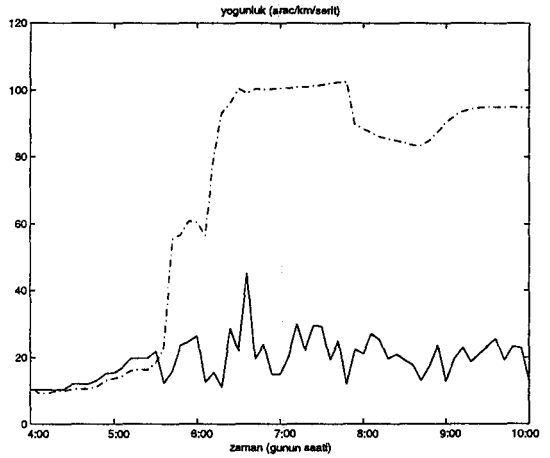
Şekil 5.95. L14'deki ortalama hız



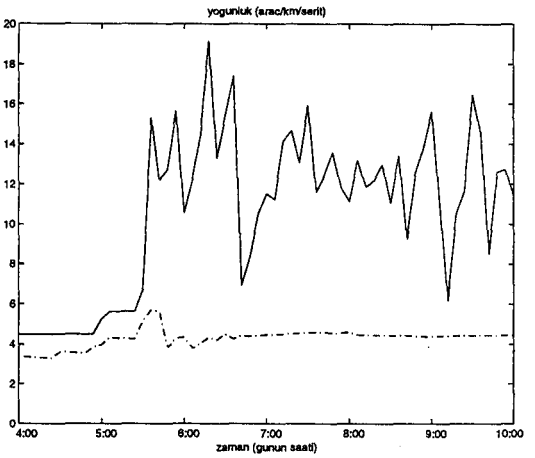
Şekil 5.93. L13'deki akış oranı



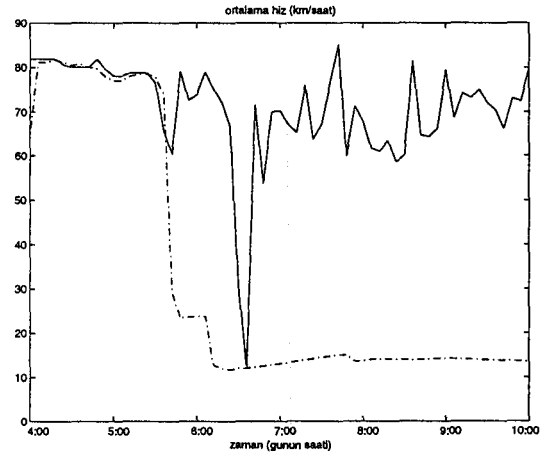
Şekil 5.96. L14'deki akış oranı



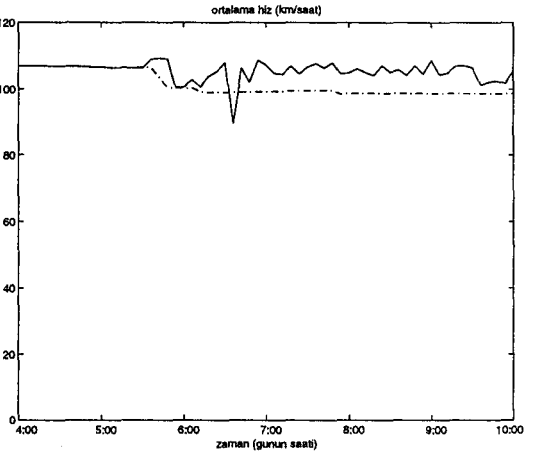
Şekil 5.97. L18'deki yoğunluk



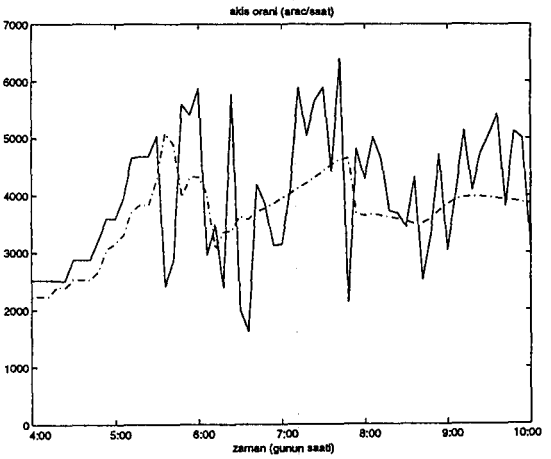
Şekil 5.100. L22'deki yoğunluk



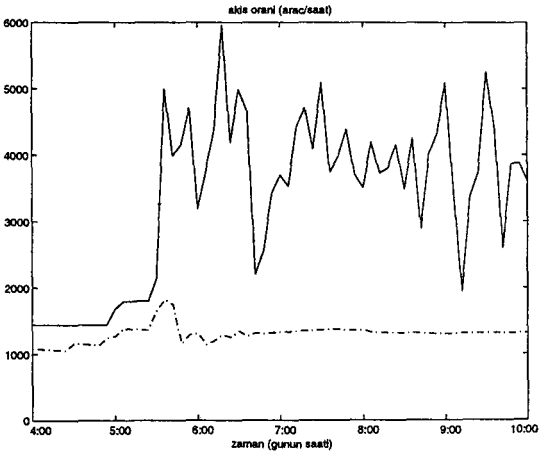
Şekil 5.98. L18'deki ortalama hız



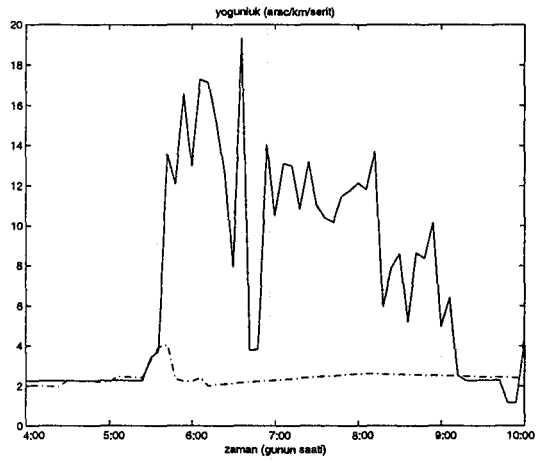
Şekil 5.101. L22'deki ortalama hız



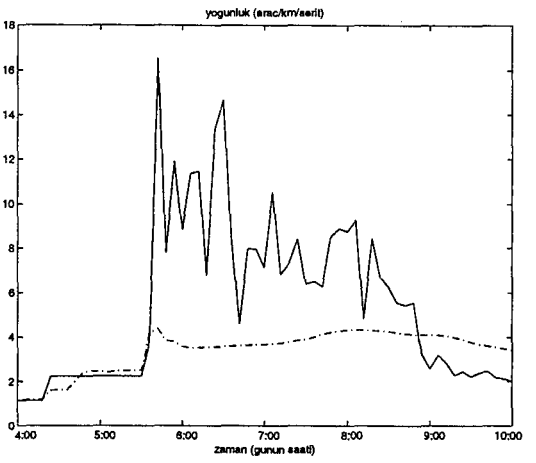
Şekil 5.99. L18'deki akış oranı



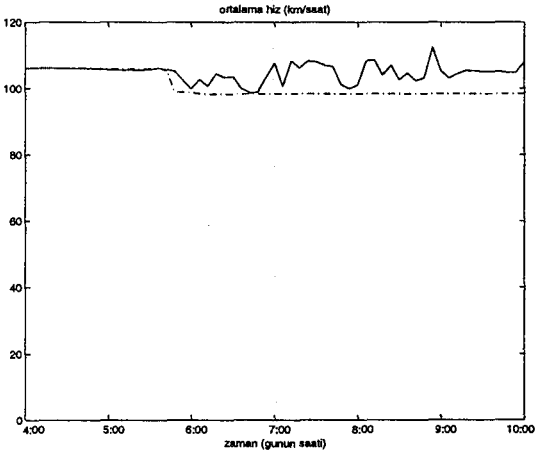
Şekil 5.102. L22'deki akış oranı



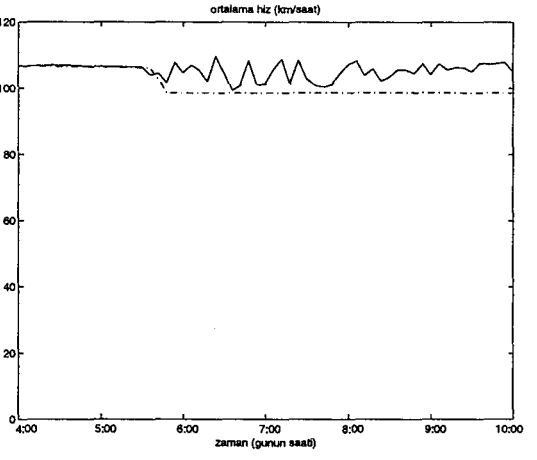
Şekil 5.103. L23'deki yoğunluk



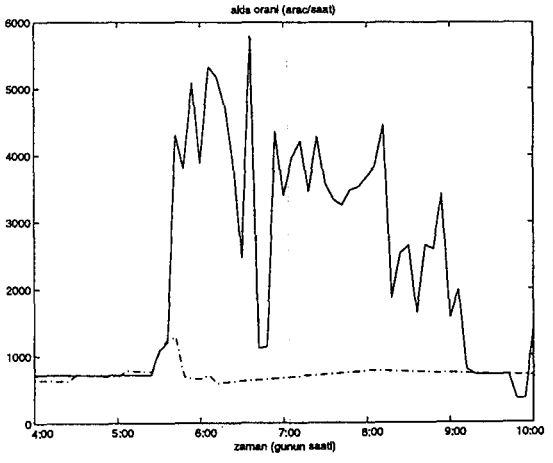
Şekil 5.106. L24'deki yoğunluk



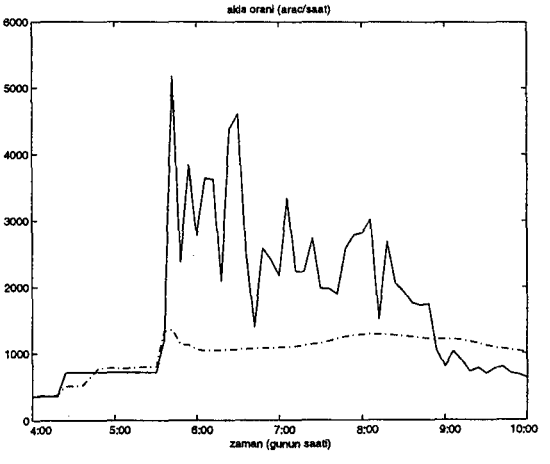
Şekil 5.104. L23'deki ortalama hız



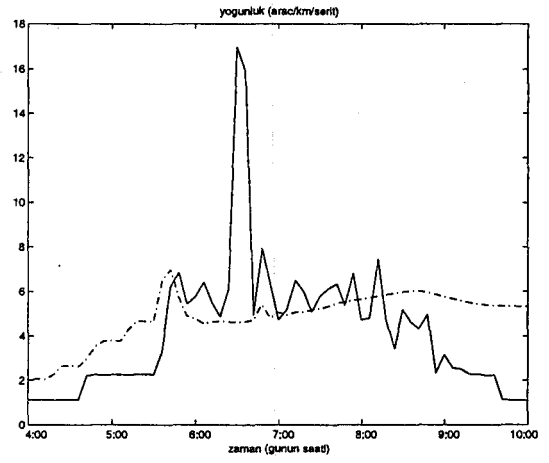
Şekil 5.107. L24'deki ortalama hız



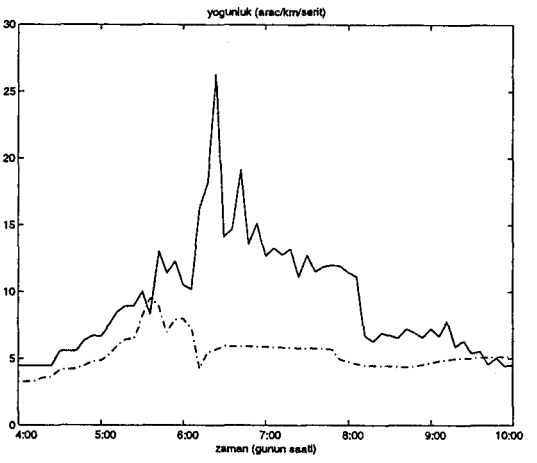
Şekil 5.105. L23'deki akış oranı



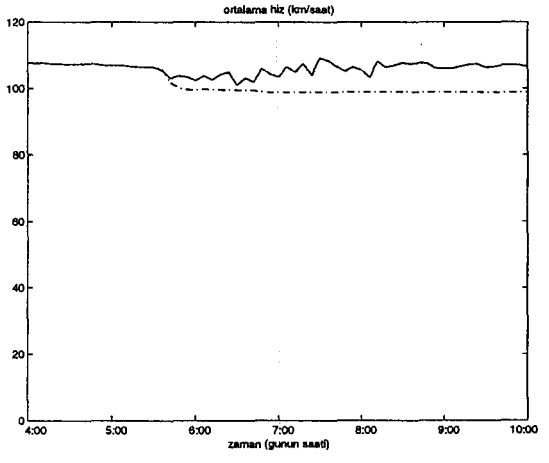
Şekil 5.108. L24'deki akış oranı



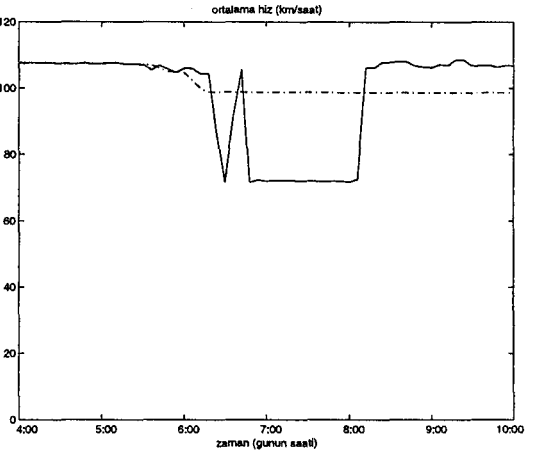
Şekil 5.109. L25'deki yoğunluk



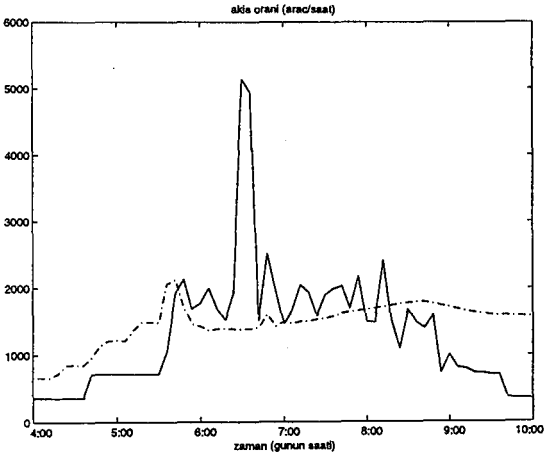
Şekil 5.112. L26'daki yoğunluk



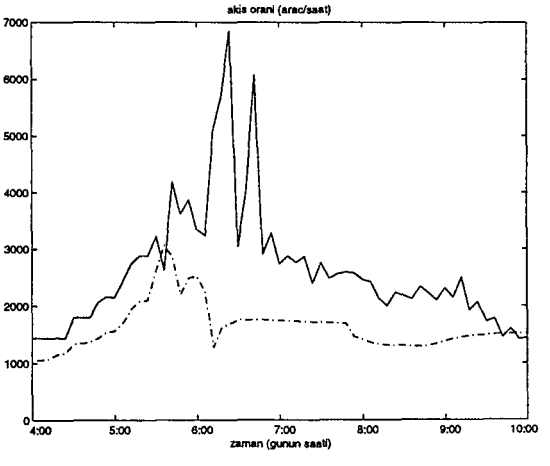
Şekil 5.110. L25'deki ortalama hız



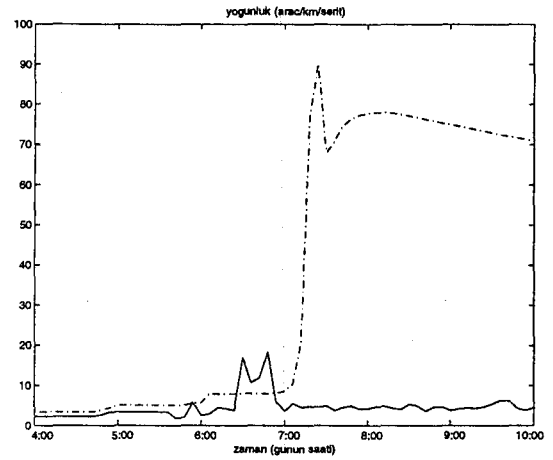
Şekil 5.113. L26'daki ortalama hız



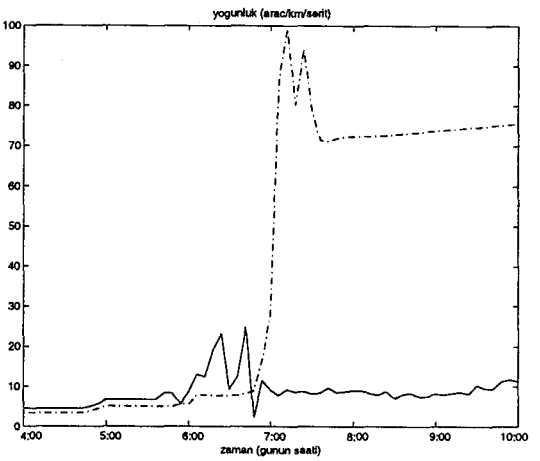
Şekil 5.111. L25'deki akış oranı



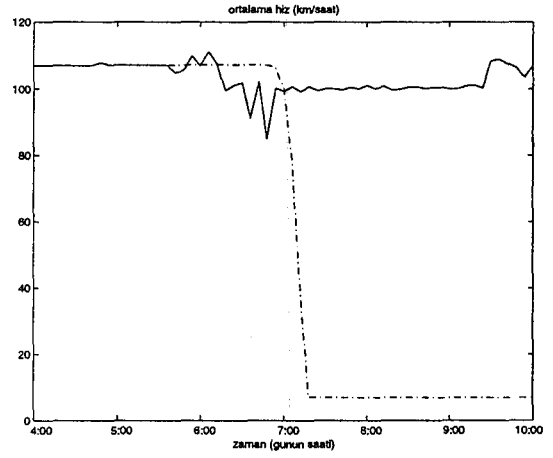
Şekil 5.114. L26'daki akış oranı



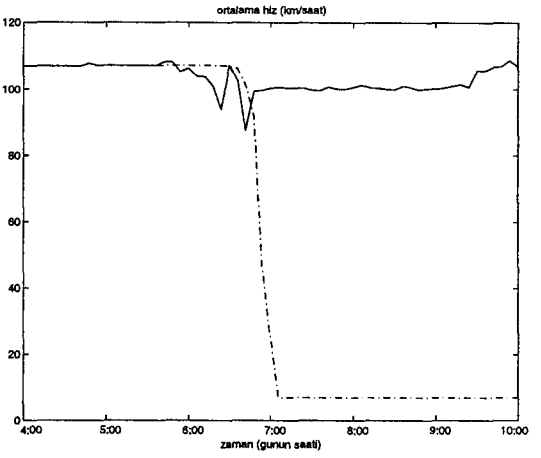
Şekil 5.115. L27'deki yoğunluk



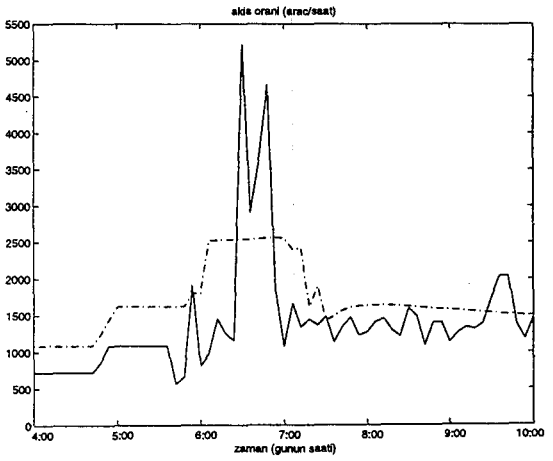
Şekil 5.118. L28'deki yoğunluk



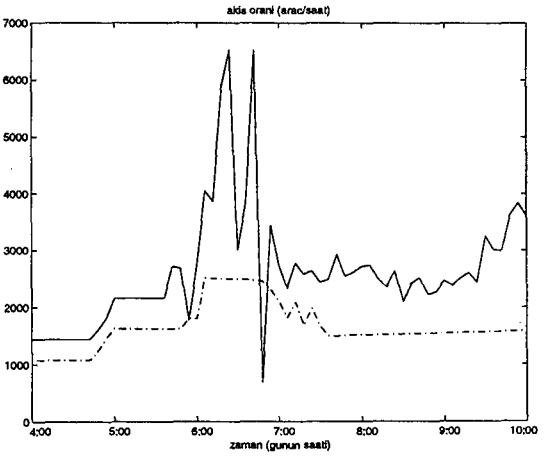
Şekil 5.116. L27'deki ortalama hız



Şekil 5.119. L28'deki ortalama hız



Şekil 5.117. L27'deki akış oranı



Şekil 5.120. L28'deki akış oranı

6. SONUÇ

Bu çalışmada, ulaşım ağlarında ortaya çıkan tıkanıklığı önlemek amacıyla bir dışmerkezli kontrol yaklaşımı önerilmiştir. Bu amaçla öncelikle [17]'de kullanılan model temel alınarak trafik ağları için bir trafik akış dinamiği modeli geliştirilmiştir. Ardından, trafik ağları için akış kontrolu ve yönlendirme kontrolunun birlikte ele alındığı bir kontrol algoritması önerilmiştir. Önerilen kontrol yaklaşımı ilk kez İftar [7] tarafından önerilmiş olan tıkanıklık ölçeğine dayalı olarak geliştirilmiştir. Önerilen algoritma her bir nodda yerel olarak uygulanabilmektedir. Bu uygulama sırasında ağdaki diğer nodlarla ilgili olarak gereken tek bilgi akış-aşağı nodlara ait tıkanıklık ölçeği miktarlarıdır. Böylece, önerilen algoritma nodlar arası çok az bilgi alış-verişi gerektirmesi anlamında dışmerkezlidir. Ayrıca önerilen algoritma nodlar arası senkronizasyon gerektirmemektedir. Yapılan bu çalışmada geliştirilen kontrol yaklaşımı hem yönlendirme kontrolu hem de akış kontrolu yaklaşımlarını birlikte ele aldığı için literatüre olan katkısı önemlidir.

Önerilen kontrol yaklaşımının gerçek trafik ağlarında göstereceği performansı önceden saptayabilmek için benzetim çalışmaları da yapılmıştır. Bu amaçla UNIX ortamında C programlama dilinde bir benzetim programı hazırlanmış ve bu program yardımıyla farklı topolojilere sahip iki trafik ağı için önerilen kontrolun benzetimi yapılmıştır. Aynı trafik ağlarına aynı trafik koşulları altında bir alternatif kontrol yaklaşımı da uygulanmış ve elde edilen sonuçlar karşılaştırılmıştır. Gerek belirli performans kriterleri üzerinden yapılan karşılaştırmalar gerekse ağ üzerinde belirli yerlerdeki trafik değişkenlerine ait grafikler üzerinden yapılan karşılaştırmalar önerilen kontrolun daha iyi sonuçlar verdiğini göstermiştir. Önerilen kontrola ait benzetimde ağa katılmak üzere nodlara gelen araçların kuyrukta daha kısa süre bekledikleri ve belirli bir benzetim süresi boyunca ağdan daha çok aracın yararlanmasının mümkün olduğu benzetim sonuçlarından açıkça görülebilmiştir. Ayrıca

ağa katılan araçlar alternatif kontrole oranla daha kısa sürede varış noktalarına ulaşabilmektedir. Ancak, önerilen kontrolde araçların nodlardan akış aşağı linklere paylaşımı yapılırken linklerdeki sıkışıklık durumunun göz önüne alınıyor olmasının getirdiği bu avantaj, bazı durumlarda seyahat edilen toplam uzunluğun önerilen kontrolde alternatif kontrole oranla daha büyük olmasına sebep olmaktadır. Bunun da araçların yıpranma payını belli açılardan artırabileceği düşünülebilir. Ancak, araçlar daha az tıkanık olan veya hiç tıkanık olmayan linklerden yönlendirildiği için trafik akışı optimum değere yakın hız değerlerinde seyredebilmekte ve böylece önerilen kontrolun benzetimi süresince zaman tasarrufunun yanısıra araçların harcadıkları yakıt miktarında da belli oranlarda tasarruf sağlanabilmektedir.

Ülkemizde özellikle büyük şehirlerimizde ortaya çıkan trafik sıkışıklığı hergün trafiğe çıkan araçlar için hem zaman kaybına ve dolayısıyla da iş gücü kaybına hem de fazla yakıt sarfiyatına sebep olmaktadır. Bu kayıpların ülke ekonomisine verdiği zararın yanısıra, trafikte yaşanan bu zorluklar sürücüleri de psikolojik yönden etkilemektedir. Yeni yolların yapılması ise ülke ekonomisi için çok büyük külfet olmaktadır. Bu sebeple, var olan yollardan etkin bir şekilde yararlanmak için kontrol yaklaşımları üretmek büyük yararlar sağlayacaktır. Bu çalışmada önerilen kontrol yaklaşımı bu açıdan da önemlidir. Ancak, ağa katılan araçların varış nodlarının bilinmesinin gerekliliği geliştirilen kontrol algoritmasının günümüz şartlarında uygulanabilirliğini güçleştirmektedir. Gelecekte, araçların içlerine yerleştirilen iletişim araçları ile varış nodu bilgisi sağlandğı takdirde, yollara yerleştirilebilecek algılayıcılar yardımıyla elde edilecek olan yoğunluk ve kuyruk uzunluğu bilgileri de kullanılarak kontrolörler tarafından yönlendirme komutları oluşturulabilecektir. Her bir kavşağa konan kontrolörler o kavşağa ait bilgilere ek olarak sadece akış-aşağı kavşaklarda bulunan kontrolörlerdeki bilgilere ihtiyaç duyacağından önerilen algoritmanın uygulanabilirliği kolay olacaktır. Belirlenen kontrol komutları yol kenarlarına konulabilecek ışıklı göstergeler veya araç içlerine yerleştirilen iletişim araçları sayesinde sürücülere ulaştırılabilecektir.

KAYNAKLAR

[1] P. E. Sarachik ve Ü. Özgüner, "On decentralized dynamic routing for congested traffic networks," *IEEE Transactions on Automatic Control*, c. AC–27, s. 1233–1238, 1982.

[2] H. T. Sheu, "A coordinated decentralized flow and routing control algorithm for an automated highway system," Ph.D. Dissertation, Dept. of Electrical Eng., The Ohio State University, Columbus, Ohio, 1987.

[3] M. Papageorgiou, "Dynamic modeling, assignment, and route guidance in traffic networks," *Transportation Research, Part B*, c. 24, s. 471–495, 1990.

[4] A. Messmer ve M. Papageorgiou, "Automatic control methods applied to freeway network traffic," *Automatica*, c. 30, s. 691–702, 1994.

[5] S. M. Easa, "Shortest route algorithm with movement prohibitions," *Transportation Research, Part B*, c. 19, s. 197–208, 1985.

[6] M. Papageorgiou, "A new approach to time-of-day control based on a dynamic traffic model," *Transportation Research, Part B*, c. 14B, s. 349–360, 1980.

[7] A. İftar, "A decentralized routing control strategy for semi-congested highways," *Preprints of the 13th IFAC World Congress*, c. P, s. 319–324, San Francisco, CA, USA, Temmuz 1996.

[8] A. İftar, "An intelligent control approach to decentralized routing and flow control in highways," *Proceedings of the 12th IEEE International Symposium on Intelligent Control*, İstanbul, Temmuz 1997.

[9] L. Isaksen ve H. J. Payne, "Suboptimal control of linear systems by augmentation with application to freeway traffic regulation," *IEEE Transactions on Automatic Control*, c. AC–18, s. 210–219, 1973.

[10] N. B. Goldstein ve K. S. P. Kumar, "A decentralized control strategy for freeway regulation," *Transportation Research, Part B*, c. 16, s. 279–290, 1982.

[11] M. Papageorgiou, J. Blosseville ve H. Hadj-Salem, "Modeling and real-time control of traffic flow on the southern part of Boulevard Peripherique in Paris: Part II: Coordinated on-ramp metering," *Transportation Research, Part A*, c. 24, s. 361–370, 1990.

[12] M. Papageorgiou, J. Blosseville ve H. Hadj-Salem, "Modeling and real-time control of traffic flow on the southern part of Boulevard Peripherique in Paris: Part I: Modelling," *Transportation Research, Part A*, c. 24, s. 345–359, 1990.

[13] M. Papageorgiou, B. Posch ve G. Schmidt, "Comparison of macroscopic models for control of freeway traffic," *Transportation Research, Part B*, c. 17, s. 107–116, 1983.

[14] K. K. Sanwal, K. Petty, J. Walrand ve Y. Fawaz, "An extended macroscopic model for traffic flow," *Transportation Research, Part B*, c. 30, s. 1–9, 1996.

[15] M. Cremer ve M. Papageorgiou, "Parameter identification for a traffic flow model," *Automatica*, c. 17, s. 837–843, 1981.

[16] M. Papageorgiou, J. Blosseville ve H. Hadj-Salem, "Macroscopic modeling of traffic flow on the Boulevard Peripherique in Paris," *Transportation Research, Part B*, c. 23, s. 29–47, 1989.

[17] A. Messmer ve M. Papageorgiou, "Metanet: A macroscopic simulation program for motorway networks," *Traffic Engng. and Control*, c. 31, s. 466–470, 1990.

[18] M. Papageorgiou ve A. M. Messmer, "Metanet: A simulation program for motorway networks," *Program Documentation*, 1990.

EK-1

Kontrol Modülü

```
/*********************************************************************/
void IN_MODEL()
{
  /* determination of the number of vehicles to be admitted to the
     network */
/*********************************************************************/
  long i, n, nd, d, ds, inr1, inr2, flag, cnt, dt, ff, lgec;
  nodespec *WITH;
  linkspec *WITH1, *WITH2, *WITH3;

  for (n = 0; n < nrofnodes; n++)
  {
   rhattop[n] = 0;
   for (d = 0; d < nrofdests; d++)  rhat[n][d] = 0;
  }
  for (n = 0; n < nrofnodes; n++)
  {
   WITH = &node[n];
   tot_rat = 0;
   switch (WITH->n_iocfg)
   {
    case 1:   /* one in-, one out-link */
     if (WITH->n_in1_lx <= nroforigs)
     {
      for (dt = 0; dt < nrofdests; dt++)
      {
       rat[dt] = t * (zeta[n][dt] + od[WITH->n_in1_lx - 1][dt]);
       tot_rat += rat[dt];
       b[dt] = 0;
      }

      if (tot_rat != 0)
      {
       WITH1 = &link[WITH->n_in1_lx - 1];
       cnt = 0;
       flag = 0;
       WITH2 = &link[WITH->n_out1_lx - 1];
       do
       {
        ff = 0;
        ds = FIND_DEST();
        if ((REACHABLE(ds+1,WITH->n_out1_lx)) &&  (p[n][ds] >=
           p[WITH2->l_nx_next - 1][ds])) ff = 1;
        if (ff == 1)
        {
         rhat[n][ds] = rhat[n][ds] + (1 / t);
         flag = 1;
         cnt ++;
         if (cnt >= WITH1->l_rnmax || cnt == tot_rat) flag = 0;
        }
        else flag = 0;
       } while (flag == 1);
      }
      for (d=0; d < nrofdests; d++) rhattop[n] += rhat[n][d];
      q[WITH->n_in1_lx - 1][0] = rhattop[n];
      if (rhattop[n] != 0.0)
```

```
   {
    for (d=0; d<nrofdests; d++)
    comps[WITH->n_in1_lx - 1][d] = ((float)rhat[n][d]) /
         ((float)rhattop[n]);
   }
  }
break;

case 2:   /* two in-, one or two out-link */
case 4:
 if (WITH->n_in1_lx <= nroforigs) inr1 = 1; else inr1 = 0;
 if (WITH->n_in2_lx <= nroforigs) inr2 = 1; else inr2 = 0;
 if ((inr1 == 1) && (inr2 == 0))
 {
  for (dt=0; dt < nrofdests; dt++)
  {
   rat[dt] = t * (zeta[n][dt] + od[WITH->n_in1_lx - 1][dt]);
   tot_rat += rat[dt];
   b[dt] = 0;
  }
  if (tot_rat != 0)
  {
   WITH1 = &link[WITH->n_in1_lx - 1];
   flag = 0;
   cnt = 0;
   WITH2 = &link[WITH->n_out1_lx - 1];
   if (WITH->n_iocfg == 4) WITH3 = &link[WITH->n_out2_lx - 1];
   do
   {
    ff = 0;
    ds = FIND_DEST();
    if ((REACHABLE(ds+1,WITH->n_out1_lx)) &&  (p[n][ds] >=
       p[WITH2->l_nx_next - 1][ds])) ff = 1;
    else {
         if ((WITH->n_iocfg == 4) &&
            (REACHABLE(ds+1,WITH->n_out2_lx)) &&
            (p[n][ds] >= p[WITH3->l_nx_next - 1][ds])) ff = 1;
         }
    if (ff == 1)
    {
     rhat[n][ds] = rhat[n][ds] + (1 / t);
     flag = 1;
     cnt++;
     if (cnt >= WITH1->l_rnmax || cnt == tot_rat) flag = 0;
    }
    else flag = 0;
    } while (flag == 1);
  }
  for (d=0; d<nrofdests; d++) rhattop[n] += rhat[n][d];
  q[WITH->n_in1_lx - 1][0] = rhattop[n];
  if (rhattop[n] != 0)
  {
  for (d=0; d<nrofdests; d++)
  comps[WITH->n_in1_lx - 1][d] = ((float)rhat[n][d]) /
       ((float)rhattop[n]);
  }
```

```
  }
  if ((inr1 == 0) && (inr2 == 1))
  {
   for (dt=0; dt < nrofdests; dt++)
   {
    rat[dt] = t * (zeta[n][dt] + od[WITH->n_in2_lx - 1][dt]);
    tot_rat += rat[dt];
    b[dt] = 0;
   }
   if (tot_rat != 0)
   {
    WITH1 = &link[WITH->n_in2_lx - 1];
    flag = 0;
    cnt = 0;
    WITH2 = &link[WITH->n_out1_lx - 1];
    if (WITH->n_iocfg == 4) WITH3 = &link[WITH->n_out2_lx - 1];
    do
    {
     ff = 0;
     ds = FIND_DEST();
     if ((REACHABLE(ds+1,WITH->n_out1_lx)) &&  (p[n][ds] >=
        p[WITH2->l_nx_next - 1][ds])) ff = 1;
     else {
          if ((WITH->n_iocfg == 4) &&
             (REACHABLE(ds+1,WITH->n_out2_lx)) &&
             (p[n][ds] >= p[WITH3->l_nx_next - 1][ds])) ff = 1;
          }
     if (ff == 1)
     {
      rhat[n][ds] = rhat[n][ds] + (1 / t);
      flag = 1;
      cnt++;
      if (cnt >= WITH1->l_rnmax || cnt == tot_rat) flag = 0;
     }
     else flag = 0;
    } while (flag == 1);
   }
   for ( d=0; d<nrofdests; d++) rhattop[n] += rhat[n][d];
   q[WITH->n_in2_lx - 1][0] = rhattop[n];
   if (rhattop[n] != 0)
   {
    for (d=0; d<nrofdests; d++)
    comps[WITH->n_in2_lx - 1][d] = ((float)rhat[n][d]) /
         ((float)rhattop[n]);
   }
  }
break;

case 3:   /* one in-, two out-links */
 if (WITH->n_in1_lx <= nroforigs)
 {
  for (dt=0; dt < nrofdests; dt++)
  {
   rat[dt] = t * (zeta[n][dt] + od[WITH->n_in1_lx - 1][dt]);
   tot_rat += rat[dt];
   b[dt] = 0;
```

```
        }
        if (tot_rat != 0)
        {
         WITH1 = &link[WITH->n_in1_lx - 1];
         flag = 0;
         cnt = 0;
         WITH2 = &link[WITH->n_out1_lx - 1];
         WITH3 = &link[WITH->n_out2_lx - 1];
         do
         {
          ds = FIND_DEST();
          if (p[n][ds] >= p[WITH2->l_nx_next - 1][ds])
          {
           rhat[n][ds] = rhat[n][ds] + (1 / t);
           rhattop[n] ++;
           flag = 1;
           cnt++;
           if (cnt >= WITH1->l_rnmax || cnt == tot_rat) flag = 0;
          }
          if (p[n][ds] >= p[WITH3->l_nx_next - 1][ds])
          {
           rhat[n][ds] = rhat[n][ds] + (1 / t);
           flag = 1;
           cnt++;
           if (cnt >= WITH1->l_rnmax || cnt == tot_rat) flag = 0;
          }
          if ((p[n][ds] < p[WITH2->l_nx_next - 1][ds]) && (p[n][ds] <
             p[WITH3->l_nx_next - 1][ds])) flag = 0;
         } while (flag == 1);
        }
        for ( d=0; d<nrofdests; d++)  rhattop[n] += rhat[n][d];
        q[WITH->n_in1_lx - 1][0] = rhattop[n];
        if (rhattop[n] != 0)
        {
         for (d=0; d<nrofdests; d++)
         comps[WITH->n_in1_lx - 1][d] = ((float)rhat[n][d]) /
              ((float)rhattop[n]);
        }
       }
      break;
     }
    }
}


/*****************************************************************/
randomize()
{

/*****************************************************************/
  time_t s;
  srand((unsigned) time(&s));
}

/*****************************************************************/
```

```
/*********************************************************************/
int FIND_DEST()
{
  /* production of destination nodes for vehicles waiting in the
     queue */
/*********************************************************************/
  long int rd, rdd, i, int_const, c[4], dt;
  int_const = 327679;

  for (dt=0; dt < nrofdests; dt++)
  {
   if (b[dt] == rat[dt]) rt[dt] = 0;
   else rt[dt] = rat[dt];
  }
  tot_rt = rt[0]+rt[1]+rt[2]+rt[3];
  for (dt = 0; dt <= 2; dt++)
  c[dt] = floor(int_const * ((float)rt[dt] / (float)tot_rt));
  randomize();
  rdd=rand();
  rd = rdd*10;
  if ((rd <= c[0]) && (c[0] != 0)) {b[0]++;  return 0;}
  if ((rd <= (c[0]+c[1]+1)) && ((c[0]+c[1]+1) != (c[0]+1)))
     {b[1]++;  return 1;}
  if ((rd <= (c[0]+c[1]+c[2]+2)) &&
     ((c[0]+c[1]+c[2]+2) != (c[0]+c[1]+2)))
     {b[2]++;  return 2;}
  if (rd <= int_const) {b[3]++; return 3;}
}


/*********************************************************************/
void NODEMODEL()
{
  /* merging and distribution at nodes                               */
/*********************************************************************/
  long i, j, n, nd, d, ff, nd1, nd2;
  fvar qmultbeta;
  nodespec *WITH;
  linkspec *WITH1, *WITH2;

  for (n = 0; n < nrofnodes; n++)
  {
   WITH = &node[n];
   switch (WITH->n_iocfg)
   {
    case 1:   /* one in-, one out-link */
     WITH1 = &link[WITH->n_in1_lx - 1];
     q[WITH->n_out1_lx - 1][0] = q[WITH->n_in1_lx - 1]
                                          [WITH1->l_nr_segm];
     nd = WITH1->l_nr_dests;
     for (i = 0; i < nd; i++)
     comps[WITH->n_out1_lx - 1][WITH1->l_dests[i] - 1] = comps[WITH->
          n_in1_lx - 1][WITH1->l_nr_segm * nrofdests
          + WITH1->l_dests[i] - 1];
    break;
```

```
    case 2:   /* two in-, one out-link */
     q[WITH->n_out1_lx - 1][0] = q[WITH->n_in1_lx - 1]
      [link[WITH->n_in1_lx - 1].l_nr_segm] + q[WITH->n_in2_lx - 1]
      [link[WITH->n_in2_lx - 1].l_nr_segm];
     for (i = 0; i < nrofdests; i++)  qn[i] = 0.0;
     nd = link[WITH->n_in1_lx - 1].l_nr_dests;
     for (i = 0; i < nd; i++)
     {
      WITH1 = &link[WITH->n_in1_lx - 1];
      qn[WITH1->l_dests[i] - 1] =
      q[WITH->n_in1_lx - 1][WITH1->l_nr_segm] * comps[WITH->n_in1_lx -
              1][WITH1->l_nr_segm * nrofdests + WITH1->l_dests[i] -
1];
     }
     nd = link[WITH->n_in2_lx - 1].l_nr_dests;
     for (i = 0; i < nd; i++)
     {
      WITH1 = &link[WITH->n_in2_lx - 1];
      qn[WITH1->l_dests[i] - 1] += q[WITH->n_in2_lx - 1]
          [WITH1->l_nr_segm] * comps[WITH->n_in2_lx - 1]
          [WITH1->l_nr_segm * nrofdests + WITH1->l_dests[i] - 1];
     }
     if (q[WITH->n_out1_lx - 1][0] != 0.0)
     {
      nd = link[WITH->n_out1_lx - 1].l_nr_dests;
      for (i = 0; i < nd; i++)
      {
       WITH1 = &link[WITH->n_out1_lx - 1];
       comps[WITH->n_out1_lx - 1][WITH1->l_dests[i] - 1] =
           qn[WITH1->l_dests[i] - 1] / q[WITH->n_out1_lx - 1][0];

      }
     }
    break;

    case 3:   /* one in-, two out-links */
     nd = link[WITH->n_in1_lx - 1].l_nr_dests;
     for (i = 0; i < nd; i++)
     {
      WITH1 = &link[WITH->n_in1_lx - 1];
      qn[WITH1->l_dests[i] - 1] = q[WITH->n_in1_lx - 1]
          [WITH1->l_nr_segm] * comps[WITH->n_in1_lx - 1]
          [WITH1->l_nr_segm * nrofdests + WITH1->l_dests[i] - 1];
     }
     q[WITH->n_out1_lx - 1][0] = 0.0;   /* init out-link flow */
     nd = link[WITH->n_out1_lx - 1].l_nr_dests;
     for (i = 0; i < nd; i++)
     {
      WITH1 = &link[WITH->n_out1_lx - 1];
      if (P_inset((int)WITH1->l_dests[i], WITH->n_dests_of_choice))
      {
       d= WITH1->l_dests[i] - 1;
       qmultbeta = 0.0;
       if (crt[n][d] == 1)
       {
        if ((p[n][d] >= p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d])
```

```
      || ((p[n][d] == p[link[WITH->n_out1_lx - 1].l_nx_next -
      1][d]) && (p[n][d] == p[link[WITH->n_out2_lx - 1].l_nx_next
      - 1][d])))
     qmultbeta = qn[WITH1->l_dests[i] - 1];
    if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
     && p[n][d] >= p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
     qmultbeta = 0.0;

    if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
     && p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
    {
     qmultbeta = p[link[WITH->n_out2_lx - 1].l_nx_next
       - 1][d] / (p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) *
       qn[WITH1->l_dests[i] - 1];
    }
   }
   if (crt[n][d] == 2)
   {
    if (p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]
     && p[n][d] >= p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d])
     qmultbeta = qn[WITH1->l_dests[i] - 1];
    if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
     && p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
    {
     if ((p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) != 0.0)
     qmultbeta = p[link[WITH->n_out2_lx - 1].l_nx_next
       - 1][d] / (p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) *
       qn[WITH1->l_dests[i] - 1];
    }
   }
  }
  else  qmultbeta = qn[WITH1->l_dests[i] - 1];
  q[WITH->n_out1_lx - 1][0] += qmultbeta;
  gammatmp[WITH1->l_dests[i] - 1] = qmultbeta;   /* preliminary */
}
if (q[WITH->n_out1_lx - 1][0] != 0.0)
{
 nd = link[WITH->n_out1_lx - 1].l_nr_dests;
 for (i = 0; i < nd; i++)
 {
  WITH1 = &link[WITH->n_out1_lx - 1];
  comps[WITH->n_out1_lx - 1][WITH1->l_dests[i] - 1] =
   gammatmp[WITH1->l_dests[i] - 1] / q[WITH->n_out1_lx - 1][0];
 }
}
q[WITH->n_out2_lx - 1][0] = 0.0;   /* init out-link flow */
nd = link[WITH->n_out2_lx - 1].l_nr_dests;
for (i = 0; i < nd; i++)
{
 WITH1 = &link[WITH->n_out2_lx - 1];
 if (P_inset((int)WITH1->l_dests[i], WITH->n_dests_of_choice))
 {
  d = WITH1->l_dests[i] - 1;
```

```
    qmultbeta = 0.0;
    if (crt[n][d] == 1)
    {
     if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
        && p[n][d] >= p[link[WITH->n_out2_lx - 1].l_nx_next
        - 1][d]) qmultbeta = qn[WITH1->l_dests[i] - 1];
     if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
        && p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
     {
      qmultbeta = p[link[WITH->n_out1_lx - 1].l_nx_next
        - 1][d] / (p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
        + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) *
        qn[WITH1->l_dests[i] - 1];
     }
    }
    if (crt[n][d] == 2)
    {
     if ((p[n][d] >= p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
        || ((p[n][d] == p[link[WITH->n_out1_lx - 1].l_nx_next -
        1][d]) && (p[n][d] == p[link[WITH->n_out2_lx - 1].l_nx_next
        - 1][d]))) qmultbeta = qn[WITH1->l_dests[i] - 1];
     if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
      && p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
     {
      qmultbeta = p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d] /
       (p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) *
       qn[WITH1->l_dests[i] - 1];
     }
    }
   }
   else qmultbeta = qn[WITH1->l_dests[i] - 1];
   q[WITH->n_out2_lx - 1][0] += qmultbeta;
   gammatmp[WITH1->l_dests[i] - 1] = qmultbeta;    /* preliminary */
  }
  if (q[WITH->n_out2_lx - 1][0] != 0.0)
   {
    nd = link[WITH->n_out2_lx - 1].l_nr_dests;
    for (i = 0; i < nd; i++)
    {
     WITH1 = &link[WITH->n_out2_lx - 1];
     comps[WITH->n_out2_lx - 1][WITH1->l_dests[i] - 1] =
       gammatmp[WITH1->l_dests[i] - 1] / q[WITH->n_out2_lx - 1][0];
    }
   }
 break;

 case 4:   /* two in-, two out-links */
  nd1 = link[WITH->n_in1_lx - 1].l_nr_dests;
  nd2 = link[WITH->n_in2_lx - 1].l_nr_dests;
  WITH1 = &link[WITH->n_in1_lx - 1];
  WITH2 = &link[WITH->n_in2_lx - 1];
  if (nd1 >= nd2)
  {
  for (i = 0; i < nd1; i++)
  {
```

```
ff = 0;
for (j = 0; j < nd2; j++)
{
 if (WITH1->l_dests[i] == WITH2->l_dests[j]){
 qn[WITH1->l_dests[i] - 1] = (q[WITH->n_in1_lx - 1]
      [WITH1->l_nr_segm]* comps[WITH->n_in1_lx - 1]
      [WITH1->l_nr_segm * nrofdests +
      WITH1->l_dests[i] - 1]) + (q[WITH->n_in2_lx - 1]
      [WITH2->l_nr_segm] * comps[WITH->n_in2_lx - 1]
      [WITH2->l_nr_segm * nrofdests + WITH2->l_dests[j] - 1]);
 ff = 1; }
}
if (ff == 0)
qn[WITH1->l_dests[i] - 1] = q[WITH->n_in1_lx - 1]
     [WITH1->l_nr_segm] * comps[WITH->n_in1_lx - 1]
     [WITH1->l_nr_segm * nrofdests + WITH1->l_dests[i] - 1];
}
}

if (nd1 < nd2)
{
for (i = 0; i < nd2; i++)
{
ff = 0;
for (j = 0; j < nd1; j++)
{
 if (WITH1->l_dests[j] == WITH2->l_dests[i]){
 qn[WITH2->l_dests[i] - 1] = (q[WITH->n_in1_lx - 1]
      [WITH1->l_nr_segm] * comps[WITH->n_in1_lx - 1]
      [WITH1->l_nr_segm * nrofdests + WITH1->l_dests[j] - 1]) +
      (q[WITH->n_in2_lx - 1][WITH2->l_nr_segm] *
      comps[WITH->n_in2_lx - 1][WITH2->l_nr_segm * nrofdests +
      WITH2->l_dests[i] - 1]);
 ff = 1; }
}
if (ff == 0)
qn[WITH2->l_dests[i] - 1] = q[WITH->n_in2_lx - 1]
     [WITH2->l_nr_segm] * comps[WITH->n_in2_lx - 1]
     [WITH2->l_nr_segm * nrofdests + WITH2->l_dests[i] - 1];
}
}

q[WITH->n_out1_lx - 1][0] = 0.0;    /* init out-link flow */
nd = link[WITH->n_out1_lx - 1].l_nr_dests;
for (i = 0; i < nd; i++)
{
 WITH1 = &link[WITH->n_out1_lx - 1];
 if (P_inset((int)WITH1->l_dests[i], WITH->n_dests_of_choice))
 {
  d= WITH1->l_dests[i] - 1;
  qmultbeta = 0.0;

  if (d == (nrofdests - nroflinks + WITH->n_out1_lx - 1))
       qmultbeta = qn[WITH1->l_dests[i] - 1];
  else{
  if (crt[n][d] == 1)
```

```
   {
    if ((p[n][d] >= p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d])
       || ((p[n][d] == p[link[WITH->n_out1_lx - 1].l_nx_next -
       1][d]) && (p[n][d] == p[link[WITH->n_out2_lx - 1].l_nx_next
       - 1][d]))) qmultbeta = qn[WITH1->l_dests[i] - 1];
    if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
     && p[n][d] >= p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
     qmultbeta = 0.0;
    if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
     && p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
    {
     qmultbeta = p[link[WITH->n_out2_lx - 1].l_nx_next
       - 1][d] / (p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) *
       qn[WITH1->l_dests[i] - 1];
    }
   }
   if (crt[n][d] == 2)
   {
    if (p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]
     && p[n][d] >= p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d])
     qmultbeta = qn[WITH1->l_dests[i] - 1];

    if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
     && p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
    {
     if ((p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) != 0.0)
       qmultbeta = p[link[WITH->n_out2_lx - 1].l_nx_next
       - 1][d] / (p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) *
       qn[WITH1->l_dests[i] - 1];
    }
   }
  }
  }
  else  qmultbeta = qn[WITH1->l_dests[i] - 1];
  qprt[WITH->n_out1_lx - 1][WITH1->l_dests[i] - 1] = qmultbeta;
  q[WITH->n_out1_lx - 1][0] += qmultbeta;
  gammatmp[WITH1->l_dests[i] - 1] = qmultbeta;   /* preliminary */
}
if (q[WITH->n_out1_lx - 1][0] != 0.0)
{
 nd = link[WITH->n_out1_lx - 1].l_nr_dests;
 for (i = 0; i < nd; i++)
 {
  WITH1 = &link[WITH->n_out1_lx - 1];
  comps[WITH->n_out1_lx - 1][WITH1->l_dests[i] - 1] =
   gammatmp[WITH1->l_dests[i] - 1] / q[WITH->n_out1_lx - 1][0];
 }
}
q[WITH->n_out2_lx - 1][0] = 0.0;   /* init out-link flow */
nd = link[WITH->n_out2_lx - 1].l_nr_dests;
for (i = 0; i < nd; i++)
{
 WITH1 = &link[WITH->n_out2_lx - 1];
```

```
    if (P_inset((int)WITH1->l_dests[i], WITH->n_dests_of_choice))
    {
     d = WITH1->l_dests[i] - 1;
     qmultbeta = 0.0;
     if (d == (nrofdests - nroflinks + WITH->n_out2_lx - 1))
        qmultbeta = qn[WITH1->l_dests[i] - 1];
     else{
     if (crt[n][d] == 1)
     {
      if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       && p[n][d] >= p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
       qmultbeta = qn[WITH1->l_dests[i] - 1];

      if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       && p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
      {
       qmultbeta = p[link[WITH->n_out1_lx - 1].l_nx_next
         - 1][d] / (p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
         + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) *
         qn[WITH1->l_dests[i] - 1];
      }
     }
     if (crt[n][d] == 2)
     {
      if ((p[n][d] >= p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
         || ((p[n][d] == p[link[WITH->n_out1_lx - 1].l_nx_next -
         1][d]) && (p[n][d] == p[link[WITH->n_out2_lx - 1].l_nx_next
         - 1][d]))) qmultbeta = qn[WITH1->l_dests[i] - 1];
      if (p[n][d] < p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
       && p[n][d] < p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d])
      {
       qmultbeta = p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d] /
        (p[link[WITH->n_out1_lx - 1].l_nx_next - 1][d]
        + p[link[WITH->n_out2_lx - 1].l_nx_next - 1][d]) *
        qn[WITH1->l_dests[i] - 1];
      }
     }
    }
    }
    else qmultbeta = qn[WITH1->l_dests[i] - 1];
    qprt[WITH->n_out2_lx - 1][WITH1->l_dests[i] - 1] = qmultbeta;
    q[WITH->n_out2_lx - 1][0] += qmultbeta;
    gammatmp[WITH1->l_dests[i] - 1] = qmultbeta;    /* preliminary */
   }
   if (q[WITH->n_out2_lx - 1][0] != 0.0)
    {
     nd = link[WITH->n_out2_lx - 1].l_nr_dests;
     for (i = 0; i < nd; i++)
     {
      WITH1 = &link[WITH->n_out2_lx - 1];
      comps[WITH->n_out2_lx - 1][WITH1->l_dests[i] - 1] =
        gammatmp[WITH1->l_dests[i] - 1] / q[WITH->n_out2_lx - 1][0];
     }
    }
break;
}
```

```
  }
}


/*****************************************************************/
CALCULATE_P()
{
  /* determination of congestion measure values                   */
/*****************************************************************/
  long n, d, ds;
  fvar qmultbeta, cgec;
  nodespec *WITH;

  for (n = 0; n < nrofnodes; n++)
  {
   WITH = &node[n];
   switch (WITH->n_iocfg)
   {
    case 1:   /* one in-, one out-link */
     for (d = 0; d < nrofdests; d++)
     {
      if (ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm] >
       link[WITH->n_in1_lx - 1].l_rocr)
      {
       cgec = comps[WITH->n_in1_lx - 1][link[WITH->n_in1_lx  -
              1].l_nr_segm * nrofdests + d];
       if (cgec > 0.0)
       ropt[WITH->n_in1_lx - 1][d] = ro[WITH->n_in1_lx - 1]
           [link[WITH->n_in1_lx - 1].l_nr_segm] * cgec;
       else ropt[WITH->n_in1_lx - 1][d] = 0.0;
       sigma[WITH->n_in1_lx - 1][d] = ropt[WITH->n_in1_lx - 1][d] /
          ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm] *
          (ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm]
          - link[WITH->n_in1_lx - 1].l_rocr);
      }
      else sigma[WITH->n_in1_lx - 1][d]=0.0;
      alfa[WITH->n_in1_lx - 1][d] = 1.0;
      bta[n][d] = 0.025;
      if (WITH->n_in1_lx != 0 && WITH->n_in1_lx <= nroforigs)
       zeta[n][d] = zeta[n][d] + od[WITH->n_in1_lx - 1][d] -
                    rhat[n][d];

      p[n][d] = alfa[WITH->n_in1_lx - 1][d] * sigma[WITH->n_in1_lx -
                1][d] + bta[n][d] * t * zeta[n][d];

    }
    break;

    case 2:   /* two in-, one out-link */
     for (d = 0; d < nrofdests; d++)
     {
      if (ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm] >
       link[WITH->n_in1_lx - 1].l_rocr)
      {
       ropt[WITH->n_in1_lx - 1][d] = ro[WITH->n_in1_lx - 1]
```

```
            [link[WITH->n_in1_lx - 1].l_nr_segm] *
            comps[WITH->n_in1_lx - 1][d];
        sigma[WITH->n_in1_lx - 1][d] = ropt[WITH->n_in1_lx - 1][d] /
          ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm] *
          (ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm]
          - link[WITH->n_in1_lx - 1].l_rocr);
      }
      else sigma[WITH->n_in1_lx - 1][d]=0.0;
      if (ro[WITH->n_in2_lx - 1][link[WITH->n_in2_lx - 1].l_nr_segm] >
       link[WITH->n_in2_lx - 1].l_rocr)
      {
       ropt[WITH->n_in2_lx - 1][d] = ro[WITH->n_in2_lx - 1]
            [link[WITH->n_in2_lx - 1].l_nr_segm] *
            comps[WITH->n_in2_lx - 1][d];
        sigma[WITH->n_in2_lx - 1][d] = ropt[WITH->n_in2_lx - 1][d] /
          ro[WITH->n_in2_lx - 1][link[WITH->n_in2_lx - 1].l_nr_segm] *
          (ro[WITH->n_in2_lx - 1][link[WITH->n_in2_lx - 1].l_nr_segm]
          - link[WITH->n_in2_lx - 1].l_rocr);
      }
      else sigma[WITH->n_in2_lx - 1][d]=0.0;
      if (WITH->n_in1_lx != 0 && WITH->n_in1_lx <= nroforigs)
       zeta[n][d] = zeta[n][d] + od[WITH->n_in1_lx - 1][d] -
rhat[n][d];
      if (WITH->n_in2_lx != 0 && WITH->n_in2_lx <= nroforigs)
       zeta[n][d] = zeta[n][d] + od[WITH->n_in2_lx - 1][d] -
                    rhat[n][d];
      p[n][d] = alfa[WITH->n_in1_lx - 1][d] * sigma[WITH->n_in1_lx -
                1][d] + alfa[WITH->n_in2_lx - 1][d] *
                sigma[WITH->n_in2_lx - 1][d] + bta[n][d] * t *
                zeta[n][d];
     }
    break;

    case 3:   /* one in-, two out-links */
     for (d = 0; d < nrofdests; d++)
     {
      if (ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm] >
       link[WITH->n_in1_lx - 1].l_rocr)
      {
       ropt[WITH->n_in1_lx - 1][d] = ro[WITH->n_in1_lx - 1]
            [link[WITH->n_in1_lx - 1].l_nr_segm] *
            comps[WITH->n_in1_lx - 1][d];
       sigma[WITH->n_in1_lx - 1][d] = ropt[WITH->n_in1_lx - 1][d] /
         ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm] *
         (ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm]
         - link[WITH->n_in1_lx - 1].l_rocr);
      }
      else sigma[WITH->n_in1_lx - 1][d]=0.0;
      if (WITH->n_in1_lx != 0 && WITH->n_in1_lx <= nroforigs)
       zeta[n][d] = zeta[n][d] + od[WITH->n_in1_lx - 1][d] -
                    rhat[n][d];
      p[n][d] = alfa[WITH->n_in1_lx - 1][d] * sigma[WITH->n_in1_lx -
                1][d] + bta[n][d] * t * zeta[n][d];

     }
    break;
```

```
    case 4:   /* two in-, two out-link */
      for (d = 0; d < nrofdests; d++)
      {
       if (ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm] >
        link[WITH->n_in1_lx - 1].l_rocr)
       {
        cgec = comps[WITH->n_in1_lx - 1][link[WITH->n_in1_lx  -
               1].l_nr_segm * nrofdests + d];
        if (cgec > 0.0)
        ropt[WITH->n_in1_lx - 1][d] = ro[WITH->n_in1_lx - 1]
              [link[WITH->n_in1_lx - 1].l_nr_segm] * cgec;
        else ropt[WITH->n_in1_lx - 1][d] = 0.0;
        sigma[WITH->n_in1_lx - 1][d] = ropt[WITH->n_in1_lx - 1][d] /
          ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm] *
          (ro[WITH->n_in1_lx - 1][link[WITH->n_in1_lx - 1].l_nr_segm]
          - link[WITH->n_in1_lx - 1].l_rocr);
       }
       else sigma[WITH->n_in1_lx - 1][d]=0.0;

       if (ro[WITH->n_in2_lx - 1][link[WITH->n_in2_lx - 1].l_nr_segm] >
        link[WITH->n_in2_lx - 1].l_rocr)
       {
        cgec = comps[WITH->n_in2_lx - 1][link[WITH->n_in2_lx -
               1].l_nr_segm * nrofdests + d];
        if (cgec > 0.0) ropt[WITH->n_in2_lx - 1][d] = ro[WITH->n_in2_lx
             - 1][link[WITH->n_in2_lx - 1].l_nr_segm] * cgec;
        else ropt[WITH->n_in2_lx - 1][d] = 0.0;
        sigma[WITH->n_in2_lx - 1][d] = ropt[WITH->n_in2_lx - 1][d] /
          ro[WITH->n_in2_lx - 1][link[WITH->n_in2_lx - 1].l_nr_segm] *
          (ro[WITH->n_in2_lx - 1][link[WITH->n_in2_lx - 1].l_nr_segm]
          - link[WITH->n_in2_lx - 1].l_rocr);
       }
       else sigma[WITH->n_in2_lx - 1][d]=0.0;
       if (WITH->n_in1_lx != 0 && WITH->n_in1_lx <= nroforigs)
        zeta[n][d] = zeta[n][d] + od[WITH->n_in1_lx - 1][d] -
                     rhat[n][d];
       if (WITH->n_in2_lx != 0 && WITH->n_in2_lx <= nroforigs)
        zeta[n][d] = zeta[n][d] + od[WITH->n_in2_lx - 1][d] -
                     rhat[n][d];

       p[n][d] = alfa[WITH->n_in1_lx - 1][d] * sigma[WITH->n_in1_lx -
        1][d] + alfa[WITH->n_in2_lx - 1][d] * sigma[WITH->n_in2_lx -
        1][d] + bta[n][d] * t * zeta[n][d];
      }
     break;
    }
   }
}
```

EK-2

Ağa Giriş Bölümlerindeki Ortalama Hız Değerlerinin Hesaplanması

```
/*********************************************************************/
void ORIGSPEED()
{
  /* calc. speeds at begin of links                                   */
  /* considers down-stream effect of in-links over nodes              */
/*********************************************************************/
  long n, o;
  fvar qsum, vn, vnx;
  origspec *WITH2;
  nodespec *WITH1;
  linkspec *WITH;

  for (n = 0; n < nrofnodes; n++)
  {
   WITH1 = &node[n];
   switch (WITH1->n_iocfg)
   {
    case 1:   /* one in-, one out-link, pass speed through */
     if (WITH1->n_in1_lx != 0 && WITH1->n_in1_lx <= nroforigs)
     {
      WITH = &link[WITH1->n_in1_lx - 1];
      if (WITH->l_vnM >= v[WITH1->n_out1_lx - 1][1])
       v[WITH1->n_in1_lx - 1][0] = v[WITH1->n_out1_lx - 1][1];
      if (WITH->l_vnM < v[WITH1->n_out1_lx - 1][1])
       v[WITH1->n_in1_lx - 1][0] = WITH->l_vnM;
     }
    break;

    case 2:   /* two in-, one out-link */
     if (WITH1->n_in1_lx != 0 && WITH1->n_in1_lx <= nroforigs)
     {
      WITH = &link[WITH1->n_in1_lx - 1];
      if (WITH->l_vnM >= v[WITH1->n_out1_lx - 1][1])
       v[WITH1->n_in1_lx - 1][0] = v[WITH1->n_out1_lx - 1][1];
      if (WITH->l_vnM < v[WITH1->n_out1_lx - 1][1])
       v[WITH1->n_in1_lx - 1][0] = WITH->l_vnM;
     }
     if (WITH1->n_in2_lx != 0 && WITH1->n_in2_lx <= nroforigs)
     {
      WITH = &link[WITH1->n_in2_lx - 1];
      if (WITH->l_vnM >= v[WITH1->n_out1_lx - 1][1])
       v[WITH1->n_in2_lx - 1][0] = v[WITH1->n_out1_lx - 1][1];
      if (WITH->l_vnM < v[WITH1->n_out1_lx - 1][1])
       v[WITH1->n_in2_lx - 1][0] = WITH->l_vnM;
     }
    break;

    case 3:   /* one in-, two out-links */
     if (WITH1->n_in1_lx != 0 && WITH1->n_in1_lx <= nroforigs)
     {
      WITH = &link[WITH1->n_in1_lx - 1];
      vnx = (v[WITH1->n_out1_lx - 1][1] + v[WITH1->n_out2_lx - 1]
            [1]) / 2;
      if (WITH->l_vnM >= vnx)  v[WITH1->n_in1_lx - 1][0] = vnx;
      if (WITH->l_vnM < vnx)  v[WITH1->n_in1_lx - 1][0] = WITH->l_vnM;
     }
```

```
    break;


    case 4:   /* two in-, two out-links */
     if (WITH1->n_in1_lx != 0 && WITH1->n_in1_lx <= nroforigs)
     {
      WITH = &link[WITH1->n_in1_lx - 1];
      vnx = (v[WITH1->n_out1_lx - 1][1] + v[WITH1->n_out2_lx - 1][1])
            / 2;
      if (WITH->l_vnM >= vnx)  v[WITH1->n_in1_lx - 1][0] = vnx;
      if (WITH->l_vnM < vnx)  v[WITH1->n_in1_lx - 1][0] = WITH->l_vnM;
     }
     if (WITH1->n_in2_lx != 0 && WITH1->n_in2_lx <= nroforigs)
     {
      WITH = &link[WITH1->n_in2_lx - 1];
      vnx = (v[WITH1->n_out1_lx - 1][1] + v[WITH1->n_out2_lx - 1][1])
             / 2;
      if (WITH->l_vnM >= vnx)  v[WITH1->n_in1_lx - 1][0] = vnx;
      if (WITH->l_vnM < vnx)  v[WITH1->n_in1_lx - 1][0] = WITH->l_vnM;
     }
    break;
   }
  }
}
```

EK-3

Ağdan Çıkış Bölümlerindeki Yoğunluk Değerlerinin Hesaplanması

```
/*********************************************************************/
void DESTDENSITY()
{
  /* calc. density at end of links                                    */
  /* considers up-stream effect of out_link densityies over nodes     */
/*********************************************************************/
  long n;
  nodespec *WITH;
  linkspec *WITH1;
  fvar TEMP, TEMP1;

  for (n = 0; n < nrofnodes; n++)
  {
   WITH = &node[n];
   switch (WITH->n_iocfg)
   {
    case 1:   /* one in-, one out-link, pass density through */
     if ((WITH->n_out1_lx > nroflinks - nrofdests) &&
        (WITH->n_out1_lx <= nroflinks))
     {
      WITH1 = &link[WITH->n_out1_lx - 1];
      ro[WITH->n_out1_lx - 1][1] = q[WITH->n_in1_lx - 1]
         [link[WITH->n_in1_lx - 1].l_nr_segm] *
         comps[WITH->n_in1_lx - 1][(link[WITH->n_in1_lx -
         1].l_nr_segm) *
         nrofdests + WITH->n_out1_lx - nroflinks + nrofdests - 1] /
         ((WITH1->l_lamda) * (WITH1->l_vno));
     }
    break;


    case 2:   /* two in-, one out-link */
     if ((WITH->n_out1_lx > nroflinks - nrofdests) &&
        (WITH->n_out1_lx <= nroflinks))
     {
      WITH1 = &link[WITH->n_out1_lx - 1];
      TEMP = q[WITH->n_in1_lx - 1]
             [link[WITH->n_in1_lx - 1].l_nr_segm] *
             comps[WITH->n_in1_lx - 1][(link[WITH->n_in1_lx -
             1].l_nr_segm) * nrofdests + WITH->n_out1_lx - nroflinks +
             nrofdests - 1];
      TEMP1 = q[WITH->n_in2_lx - 1][link[WITH->n_in2_lx -
              1].l_nr_segm] * comps[WITH->n_in2_lx - 1]
              [(link[WITH->n_in2_lx - 1].l_nr_segm) *
              nrofdests + WITH->n_out1_lx - nroflinks +
              nrofdests - 1];
      ro[WITH->n_out1_lx - 1][1] = (TEMP + TEMP1) /
                   ((WITH1->l_lamda) * (WITH1->l_vno));
     }
    break;


    case 3:   /* one in-, two out-links */
     if ((WITH->n_out1_lx > nroflinks - nrofdests) &&
        (WITH->n_out1_lx <= nroflinks) && (WITH->n_out2_lx <
        nroflinks - nrofdests))
```

```
 {

  WITH1 = &link[WITH->n_out1_lx - 1];
  ro[WITH->n_out1_lx - 1][1] = q[WITH->n_in1_lx - 1]
     [link[WITH->n_in1_lx - 1].l_nr_segm] *
     comps[WITH->n_in1_lx - 1][(link[WITH->n_in1_lx -
     1].l_nr_segm) *
     nrofdests + WITH->n_out1_lx - nroflinks + nrofdests - 1] /
     ((WITH1->l_lamda) * (WITH1->l_vno));
 }
 if ((WITH->n_out2_lx > nroflinks - nrofdests) && (WITH->n_out2_lx
  <= nroflinks) && (WITH->n_out1_lx < nroflinks - nrofdests))
 {
  WITH1 = &link[WITH->n_out2_lx - 1];
  ro[WITH->n_out2_lx - 1][1] = q[WITH->n_in1_lx - 1]
     [link[WITH->n_in1_lx - 1].l_nr_segm] *
     comps[WITH->n_in1_lx - 1][(link[WITH->n_in1_lx -
     1].l_nr_segm) *
     nrofdests + WITH->n_out2_lx - nroflinks + nrofdests - 1] /
     ((WITH1->l_lamda) * (WITH1->l_vno));
 }
break;


case 4:   /* two in-, two out-links */
 if ((WITH->n_out1_lx > nroflinks - nrofdests) && (WITH->n_out1_lx
  <= nroflinks) && (WITH->n_out2_lx < nroflinks - nrofdests))
 {
  WITH1 = &link[WITH->n_out1_lx - 1];
  ro[WITH->n_out1_lx - 1][1] = (q[WITH->n_in1_lx - 1]
     [link[WITH->n_in1_lx - 1].l_nr_segm] *
     comps[WITH->n_in1_lx - 1][(link[WITH->n_in1_lx -
     1].l_nr_segm) *
     nrofdests + WITH->n_out1_lx - nroflinks + nrofdests - 1] +
     q[WITH->n_in2_lx - 1]
     [link[WITH->n_in2_lx - 1].l_nr_segm] *
     comps[WITH->n_in2_lx - 1][(link[WITH->n_in2_lx -
     1].l_nr_segm) *
     nrofdests + WITH->n_out1_lx - nroflinks + nrofdests - 1]) /
     ((WITH1->l_lamda) * (WITH1->l_vno));
 }
 if ((WITH->n_out2_lx > nroflinks - nrofdests) && (WITH->n_out2_lx
  <= nroflinks) && (WITH->n_out1_lx < nroflinks - nrofdests))
 {
  WITH1 = &link[WITH->n_out2_lx - 1];
  ro[WITH->n_out2_lx - 1][1] = (q[WITH->n_in1_lx - 1]
     [link[WITH->n_in1_lx - 1].l_nr_segm] *
     comps[WITH->n_in1_lx - 1][(link[WITH->n_in1_lx -
     1].l_nr_segm) *
     nrofdests + WITH->n_out2_lx - nroflinks + nrofdests - 1] +
     q[WITH->n_in2_lx - 1]
     [link[WITH->n_in2_lx - 1].l_nr_segm] *
     comps[WITH->n_in2_lx - 1][(link[WITH->n_in2_lx -
     1].l_nr_segm) *
     nrofdests + WITH->n_out2_lx - nroflinks + nrofdests - 1]) /
     ((WITH1->l_lamda) * (WITH1->l_vno));
```

```
        }
       break;
      }
    }
  }
```

EK-4

Birinci Benzetim İçin Hazırlanmış Olan Giriş Dosyaları

ag1.CTR
ag1.NWD
ag1.INI
ag1.MSD
ag1.ODM

```
H Simulation Control File for AG.1
C
C Sim.start time, Sim.end, Sim.step[sec]
| 04:00           10:00    10
C output:
C type, start, end,  time step
| c     04:00  10:00  00:06
C link name       link km ...
C| L13  1.35
C| U1
C| U2
C| Z1
C| Z2
C| L3              0.0
C| L27             0.0
C| L7              0.0
C| L29             0.0
C| L12             0.0
C| L28             0.0
C| Z3
C| L15             0.0
C| Z4
C| L17             0.0
C| L22             0.0
C| L31             0.0
C| Z5
C| L25             0.0
E
```

```
C NETWORK DESCRIPTION FILE FOR AG.1
C
C PARAMETER TAU, KAPPA, NUE, VMIN, ROMAX, DELTA, PHI
|           20   13     35   7.0   180    0.8    2.0
C
C ORIGIN NAME       LANES  FREE-SPEED          vnM      rnmax
| U1                3           109            60       5400
| U2                3           109            60       5400
| U3                1           109            60       2160
| U4                1           109            60       2160
| U5                1           109            60       2160
E
C LINK NAME         LANES CAP/LANE FREE-SPEED ROCRIT   LENGTH  Nr. of
SEGMENTS
| L2                3     2214.7   109        33.5      0.8     2
| L3                2     2214.7   109        33.5      1.0     2
| L4                3     2214.7   109        33.5      3.15    7
| L6                3     2511.0   115        36.0      1.5     3
| L7                3     2214.7   109        33.5      0.95    2
| L8                3     2148.6   109        32.5      1.8     4
| L11               3     2380.0   109        36.0      1.0     2
| L12               2     2214.7   109        33.5      0.25    0
| L13               3     1698.3    80        35.0      1.8     4
| L15               3     2214.7   109        33.5      0.85    2
| L17               2     2214.7   109        33.5      1.3     3
| L19               3     2214.7   109        33.5      0.4     1
| L21               3     2214.7   109        33.5      0.95    2
| L22               3     2214.7   109        33.5      0.5     1
| L23               3     2214.7   109        33.5      1.25    3
| L25               2     1486.0    80        35.0      0.75    1
| L27               2     2214.7   109        33.5      0.4     1
| L28               3     2214.7   109        33.5      1.33    3
| L29               2     2214.7   109        33.5      0.2     0
| L30               1     2214.7   109        33.5      0.5     1
| L31               2     2214.7   109        33.5      0.5     1
E
C DESTINATION NAME   LANES    FREE-SPEED          vno
| Z1                 3             109           80
| Z2                 2             109           70
| Z3                 2             109           55
| Z4                 1             109           60
| Z5                 1             109           60
E
C NODE NAME
C   INLINK NAMES
C   OUTLINK NAMES
| N1
|   U1
|   L2
| N2
|   L2
|   L27 L3
```

```
| N3
|   L28 L3
|   L4
| N4
|   L4 U3
|   L6
| N5
|   L6
|   L7 L29
| N6
|   L7 L30
|   L8
| N7
|   L8
|   Z1
| N8
|   U2
|   L11
| N9
|   L11
|   L28 L12
| N10
|   L27 L12
|   L13
| N11
|   L13
|   L15 Z3
| N12
|   L15
|   L17 Z4
| N13
|   L17 U4
|   L19
| N14
|   L19 U5
|   L21
| N15
|   L21
|   L22 L31
| N16
|   L22 L29
|   L23
| N17
|   L23
|   L25 Z5
| N18
|   L25
|   Z2
| N19
|   L31
|   L30
E
C alfa_m^l degerleri
| L2        Z1          Z2          Z3          Z4          Z5
|           3.0         3.0         3.0         3.0         3.0
| L3        Z1          Z2                                  Z5
```

```
|            3.0         3.0                                    3.0
| L4         Z1          Z2                                     Z5
|            3.0         3.0                                    3.0
| L6         Z1          Z2                                     Z5
|            3.0         3.0                                    3.0
| L11        Z1          Z2          Z3          Z4             Z5
|            3.0         3.0         3.0         3.0            3.0
| L12        Z1          Z2          Z3          Z4             Z5
|            3.0         3.0         3.0         3.0            3.0
| L13        Z1          Z2          Z3          Z4             Z5
|            3.0         3.0         3.0         3.0            3.0
| L15        Z1          Z2                      Z4             Z5
|            3.0         3.0                     3.0            3.0
| L17        Z1          Z2                                     Z5
|            3.0         3.0                                    3.0
| L19        Z1          Z2                                     Z5
|            3.0         3.0                                    3.0
| L21        Z1          Z2                                     Z5
|            3.0         3.0                                    3.0
| L22                    Z2                                     Z5
|                        3.0                                    3.0
| L23                    Z2                                     Z5
|                        3.0                                    3.0
| L27        Z1          Z2          Z3          Z4             Z5
|            3.0         3.0         3.0         3.0            3.0
| L28        Z1          Z2                                     Z5
|            3.0         3.0                                    3.0
| L29        Z2                                                 Z5
|            3.0                                                3.0
E
C beta_n^l degerleri
| N1         Z1          Z2          Z3          Z4             Z5
|            0.025       0.025       0.025       0.025          0.025
| N4         Z1          Z2                                     Z5
|            0.025       0.025                                  0.025
| N8         Z1          Z2          Z3          Z4             Z5
|            0.025       0.025       0.025       0.025          0.025
| N13        Z1          Z2                                     Z5
|            0.025       0.025                                  0.025
| N14        Z1          Z2                                     Z5
|            0.025       0.025                                  0.025
E
C S_n^l kumesi elemanlarinin oncelik sirasi
| N2         Z1          Z2          Z3          Z4             Z5
|            2           1           0           0              1
| N9         Z1          Z2          Z3          Z4             Z5
|            1           2           0           0              2
E
```

```
C INITIAL VALUES FOR DENSITIES IN NETWORK LINKS
C LINK NAME robegin roend ---> roend is omitted for destinations
| Z1          8
| Z2         10
| Z3         10
| Z4         10
| Z5         10
| L2         10        10
| L3          8         8
| L4          6         6
| L6          8         8
| L7          3.5       3.5
| L8          8         8
| L11        10        10
| L12        10        10
| L13        15        15
| L15         9         9
| L17        10        10
| L19         8         8
| L21        10        10
| L22         5         5
| L23        13        13
| L25        14        14
| L27        10        10
| L28         8         8
| L29         8         8
| L30        18        18
| L31        15        15
E
C LINK NAME DESTINNAME1 DESTINNAME2 DESTINNAME3 ...
C   GAMMALINKBEGIN
C   GAMMALINKEND
| L2         Z1          Z2          Z3          Z4          Z5
|            0.3         0.3         0.3         0.05        0.05
|            0.3         0.3         0.3         0.05        0.05
| L3         Z1          Z2                                  Z5
|            0.45        0.45                                0.1
|            0.45        0.45                                0.1
| L4         Z1          Z2                                  Z5
|            0.45        0.45                                0.1
|            0.45        0.45                                0.1
| L6         Z1          Z2                                  Z5
|            0.45        0.45                                0.1
|            0.45        0.45                                0.1
| L11        Z1          Z2          Z3          Z4          Z5
|            0.3         0.3         0.3         0.05        0.05
|            0.3         0.3         0.3         0.05        0.05
| L12        Z1          Z2          Z3          Z4          Z5
|            0.3         0.3         0.3         0.05        0.05
|            0.3         0.3         0.3         0.05        0.05
| L13        Z1          Z2          Z3          Z4          Z5
|            0.45        0.4         0.05        0.05        0.05
```

	0.45	0.4	0.05	0.05	0.05
L15	Z1	Z2		Z4	Z5
	0.45	0.45		0.05	0.05
	0.45	0.45		0.05	0.05
L17	Z1	Z2			Z5
	0.45	0.45			0.1
	0.45	0.45			0.1
L19	Z1	Z2			Z5
	0.45	0.45			0.1
	0.45	0.45			0.1
L21	Z1	Z2			Z5
	0.45	0.45			0.1
	0.45	0.45			0.1
L22		Z2			Z5
		0.9			0.1
		0.9			0.1
L23		Z2			Z5
		0.9			0.1
		0.9			0.1
L27	Z1	Z2	Z3	Z4	Z5
	0.3	0.3	0.3	0.05	0.05
	0.3	0.3	0.3	0.05	0.05
L28	Z1	Z2			Z5
	0.4	0.4			0.2
	0.4	0.4			0.2
L29	Z2				Z5
	0.8				0.2
	0.8				0.2

E

```
P Simulation traffic input data
C
F 5(Q)
C
C Collection Start Sample Interval/step
C hh:mm      hh:mm:ss
T 04:00      00:06
C
C Names of Measurement Locations
N  U1      U2      U3      U4      U5
C  Q       Q       Q       Q       Q
|  1490    1550    1000    1210    1000
|  1330    1450    1100    1220    1040
|  1630    1450    1110    1220    1240
|  1420    1410    1110    1220    1230
|  1590    1590    1200    1310    1440
|  1620    1580    1100    1480    1540
|  1650    1520    1200    1500    1640
|  1620    1790    1300    1690    1740
|  1810    1900    1400    1790    1840
|  1780    2070    1500    1800    1940
|  1800    2100    1600    1900    2050
|  1850    2250    1700    2030    2150
|  1840    2390    1810    2060    2250
|  1900    2400    1900    2190    2360
|  2060    2540    2000    2210    2360
|  2180    2680    2100    2350    2470
|  2210    2780    2200    2430    2590
|  2380    2700    2100    2500    2600
|  2400    2870    2300    2570    2710
|  2430    2910    2200    2690    2840
|  2550    2950    2300    2760    2950
|  2510    3000    2360    2840    2970
|  2500    3140    2400    2940    2890
|  2790    3470    2500    3060    2810
|  3090    3500    2600    3150    2830
|  4110    4290    2600    3320    2600
|  3750    4870    2700    3460    2740
|  4570    5370    2750    3650    2830
|  5770    5360    2800    3910    2880
|  4800    5480    2900    3960    2900
|  6110    5600    3000    4000    2940
|  6170    5450    3100    4220    2860
|  5950    5290    3000    4160    2700
|  6280    5060    3000    4070    2730
|  6310    4830    2900    3980    2620
|  5900    4670    2800    3870    2410
|  5610    4340    2700    3790    2200
|  5200    4060    2500    3640    2100
|  5090    3840    2400    3530    2030
|  4900    3630    2300    3430    1910
|  4820    3580    2330    3320    1890
```

4700	3410	2100	3110	1660
4650	3320	2050	3090	1580
4510	3250	1900	2950	1450
4440	3170	1750	2870	1300
4210	2940	1570	2760	1230
4100	2870	1420	2660	1190
4080	2780	1340	2520	1180
3500	2690	1260	2500	1160
3390	2500	1000	2420	1140
3130	2460	950	2340	1150
2890	2320	750	2250	1150
2730	2290	850	2160	1110
2670	2000	770	2010	1120
2590	1980	700	1580	1140
2430	1800	790	1410	1100
2550	1770	770	1480	1040
2510	1680	850	1440	1050
2410	1560	820	1390	1020
2570	1560	870	1350	1020
2570	1460	870	1350	1000

```
C Origin-Destination Information
C
F T:, 2(5G), 3(3G)
C
N U1         Z1    Z2    Z3    Z4    Z5
  U2         Z1    Z2    Z3    Z4    Z5
  U3         Z1    Z2                Z5
  U4         Z1    Z2                Z5
  U5         Z1    Z2                Z5
C
| 04:00   0.20    0.20    0.20    0.20    0.20
          0.20    0.20    0.20    0.20    0.20
          0.30    0.30                    0.40
          0.40    0.30                    0.30
          0.30    0.40                    0.30
| 10:00   0.20    0.20    0.20    0.20    0.20
          0.20    0.20    0.20    0.20    0.20
          0.30    0.30                    0.40
          0.40    0.30                    0.30
          0.30    0.40                    0.30
```

EK-5

İkinci Benzetim İçin Hazırlanmış Olan Giriş Dosyaları

ag2.CTR
ag2.NWD
ag2.INI
ag2.MSD
ag2.ODM

```
H Simulation Control File for AG.2
C
C Sim.start time, Sim.end, Sim.step[sec]
| 04:00          10:00    10
C Correspondance between measuring locations in input and
origins/destinations
C measuring loc. name, link name, link km
| U1                   U1
| U2                   U2
| U3                   U3
| U4                   U4
E
C output:
C type, start, end,  time step
| c     04:00  10:00  00:06
C| L13  1.35
C| U1
C| U2
C| Z1
C| Z2
C| L3              0.0
C| L27             0.0
C| L7              0.0
C| L29             0.0
C| L12             0.0
C| L28             0.0
C| Z3
C| L15             0.0
C| Z4
C| L17             0.0
C| L22             0.0
C| L31             0.0
C| Z5
C| L25             0.0
E
```

```
|   L5 L14
| N3
|   L13 L22
|   L3 Z1
| N4
|   U2
|   L7
| N5
|   L7 L12
|   L8 L24
| N6
|   L8 L27
|   L9 L25
| N7
|   L9 L23
|   L11 Z2
| N8
|   L5 L11
|   L12 L13
| N9
|   U3
|   L16
| N10
|   L16 L21
|   L17 L23
| N11
|   L14 L17
|   L18 L22
| N12
|   L18 L24
|   L20 Z3
| N13
|   L20 L28
|   L21 L26
| N14
|   U4
|   L30
| N15
|   L30 L31
|   L27 L28
| N16
|   L25 L26
|   L31 Z4
E
C alfa_m^l degerleri
| L2        Z1          Z2          Z3          Z4
|           3.0         3.0         3.0         3.0
| L3                    Z2          Z3          Z4
|                       3.0         3.0         3.0
| L5        Z1          Z2          Z3          Z4
|           3.0         3.0         3.0         3.0
| L7        Z1          Z2          Z3          Z4
|           3.0         0.6         0.2         3.0
| L8        Z1          Z2          Z3          Z4
|           3.0         0.6         0.2         3.0
| L9        Z1          Z2          Z3          Z4
```

```
|            3.0         3.0         3.0         3.0
| L11        Z1                      Z3          Z4
|            3.0                     3.0         3.0
| L12        Z1          Z2          Z3          Z4
|            3.0         0.2         0.6         3.0
| L13        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L14        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L16        Z1          Z2          Z3          Z4
|            2.0         3.0         3.0         2.0
| L17        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L18        Z1          Z2          Z3          Z4
|            3.0         0.2         0.6         3.0
| L20        Z1          Z2                      Z4
|            3.0         3.0                     3.0
| L21        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L22        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L23        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L24        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L25        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L26        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L27        Z1          Z2          Z3          Z4
|            3.0         0.6         0.2         3.0
| L28        Z1          Z2          Z3          Z4
|            3.0         3.0         3.0         3.0
| L30        Z1          Z2          Z3
|            3.0         3.0         3.0
| L31        Z1          Z2          Z3
|            3.0         3.0         3.0
E
C beta_n^l degerleri
| N1         Z1          Z2          Z3          Z4
|            0.025       0.025       0.025       0.025
| N4         Z1          Z2          Z3          Z4
|            0.025       0.025       0.025       0.025
| N9         Z1          Z2          Z3          Z4
|            0.025       0.025       0.025       0.025
| N14        Z1          Z2          Z3          Z4
|            0.025       0.025       0.025       0.025
E
C S_n^l kumesi elemanlarinin oncelik sirasi
| N2         Z1          Z2          Z3          Z4
|            1           1           2           2
| N5         Z1          Z2          Z3          Z4
|            1           1           2           1
| N6         Z1          Z2          Z3          Z4
|            1           1           1           2
| N8         Z1          Z2          Z3          Z4
```

```
|          2          1          1          1
| N10      Z1         Z2         Z3         Z4
|          1          2          1          1
| N11      Z1         Z2         Z3         Z4
|          2          1          1          1
| N13      Z1         Z2         Z3         Z4
|          1          1          1          2
| N15      Z1         Z2         Z3         Z4
|          2          1          2          1
E
```

```
C INITIAL VALUES FOR DENSITIES IN NETWORK LINKS
C LINK NAME robegin roend ---> roend is omitted for destinations
| Z1          8
| Z2         10
| Z3         10
| Z4         10
| L2          8        8
| L3          6        6
| L5          8        8
| L7          8        8
| L8          6        6
| L9          6        6
| L11         6        6
| L12         6        6
| L13         6        6
| L14         6        6
| L16         9        9
| L17        10       10
| L18         8        8
| L20         5        5
| L21        10       10
| L22        10       10
| L23        10       10
| L24        10       10
| L25        10       10
| L26        10       10
| L27        10       10
| L28        10       10
| L30        10       10
| L31        10       10
E
C LINK NAME DESTINNAME1 DESTINNAME2 DESTINNAME3 ...
C   GAMMALINKBEGIN
C   GAMMALINKEND
| L2          Z1          Z2          Z3          Z4
|             0.1         0.7         0.1         0.1
|             0.1         0.7         0.1         0.1
| L3                      Z2          Z3          Z4
|                         0.4         0.3         0.3
|                         0.4         0.3         0.3
| L5          Z1          Z2          Z3          Z4
|             0.1         0.7         0.1         0.1
|             0.1         0.7         0.1         0.1
| L7          Z1          Z2          Z3          Z4
|             0.1         0.6         0.2         0.1
|             0.1         0.6         0.2         0.1
| L8          Z1          Z2          Z3          Z4
|             0.1         0.6         0.2         0.1
|             0.1         0.6         0.2         0.1
| L9          Z1          Z2          Z3          Z4
|             0.1         0.7         0.1         0.1
|             0.1         0.7         0.1         0.1
```

L11	Z1		Z3	Z4
	0.4		0.3	0.3
	0.4		0.3	0.3
L12	Z1	Z2	Z3	Z4
	0.1	0.2	0.6	0.1
	0.1	0.2	0.6	0.1
L13	Z1	Z2	Z3	Z4
	0.7	0.1	0.1	0.1
	0.7	0.1	0.1	0.1
L14	Z1	Z2	Z3	Z4
	0.1	0.1	0.7	0.1
	0.1	0.1	0.7	0.1
L16	Z1	Z2	Z3	Z4
	0.3	0.3	0.3	0.1
	0.3	0.3	0.3	0.1
L17	Z1	Z2	Z3	Z4
	0.3	0.3	0.3	0.1
	0.3	0.3	0.3	0.1
L18	Z1	Z2	Z3	Z4
	0.1	0.2	0.6	0.1
	0.1	0.2	0.6	0.1
L20	Z1	Z2		Z4
	0.3	0.3		0.4
	0.3	0.3		0.4
L21	Z1	Z2	Z3	Z4
	0.1	0.7	0.1	0.1
	0.1	0.7	0.1	0.1
L22	Z1	Z2	Z3	Z4
	0.7	0.1	0.1	0.1
	0.7	0.1	0.1	0.1
L23	Z1	Z2	Z3	Z4
	0.1	0.7	0.1	0.1
	0.1	0.7	0.1	0.1
L24	Z1	Z2	Z3	Z4
	0.1	0.1	0.7	0.1
	0.1	0.1	0.7	0.1
L25	Z1	Z2	Z3	Z4
	0.1	0.1	0.1	0.7
	0.1	0.1	0.1	0.7
L26	Z1	Z2	Z3	Z4
	0.1	0.1	0.1	0.7
	0.1	0.1	0.1	0.7
L27	Z1	Z2	Z3	Z4
	0.1	0.6	0.2	0.1
	0.1	0.6	0.2	0.1
L28	Z1	Z2	Z3	Z4
	0.3	0.3	0.3	0.1
	0.3	0.3	0.3	0.1
L30	Z1	Z2	Z3	
	0.3	0.3	0.4	
	0.3	0.3	0.4	
L31	Z1	Z2	Z3	
	0.3	0.3	0.4	
	0.3	0.3	0.4	

E

```
P Simulation traffic input data
C
F 4(Q)
C
C Collection Start Sample Interval/step
C hh:mm            hh:mm:ss
T 04:00            00:06
C
C Names of Measurement Locations (must not contain blancs)
N  U1      U2       U3    U4
C  Q       Q        Q     Q
| 1490    1450     2100  2210
| 1330    1450     2100  2220
| 1630    1450     2100  2220
| 1820    1410     2120  2220
| 1990    1590     2200  2210
| 2020    1580     2700  2180
| 2150    1520     2710  2200
| 2220    1790     2850  2490
| 2310    1900     2850  2690
| 2480    1870     2900  2700
| 2600    1900     2900  2830
| 2650    1750     2950  2930
| 4000    1890     3010  3160
| 4000    1900     3010  3290
| 4000    1840     3200  3310
| 4000    1980     4000  3350
| 6000    4000     5000  3430
| 6000    4000     5000  3500
| 6000    4000     6000  3570
| 6000    4000     6000  3690
| 6000    4000     6000  3760
| 6000    4000     6000  6840
| 6000    4000     6000  7000
| 6000    4000     6000  7000
| 6000    4000     6000  7000
| 6000    4000     6000  8000
| 8000    4000     6000  8000
| 8000    4000     6000  7000
| 8000    6000     6000  6000
| 8000    6000     8000  6000
| 8000    6000     8000  6000
| 6000    6000     8000  6000
| 6000    6000     8000  4260
| 6000    6000     8000  4170
| 4000    6000     5000  4180
| 4000    6000     3500  4170
| 4000    6000     3400  4090
| 4000    6000     3400  4040
| 4000    6000     3420  4030
| 3000    6000     3310  3830
| 3000    6000     3310  3620
```

3000	4000	3400	3410
3000	4000	3350	3190
3000	4000	3330	2950
3000	4000	3280	2770
2500	4000	3270	2760
2500	4000	3020	2760
2500	4000	2940	2740
2200	4000	2860	2730
2190	2200	2750	2720
2000	2000	2550	2700
1950	1950	2400	2550
1800	1900	2200	2200
1750	1900	1950	2100
1650	1900	1950	1900
1650	1900	1900	1900
1600	1900	1900	1900
1600	1600	1900	1900
1600	1600	1600	1600
1600	1600	1600	1600
1600	1600	1600	1600

```
C Origin-Destination Information
C
F T:, 4(4G)
C
N U1          Z1    Z2    Z3    Z4
  U2          Z1    Z2    Z3    Z4
  U3          Z1    Z2    Z3    Z4
  U4          Z1    Z2    Z3    Z4
C
| 04:00   0.0       0.30    0.30    0.40
          0.30      0.0     0.35    0.35
          0.35      0.30    0.0     0.35
          0.30      0.35    0.35    0.0

| 05:00   0.0       0.32    0.30    0.38
          0.31      0.0     0.37    0.32
          0.33      0.33    0.0     0.34
          0.40      0.30    0.30    0.0

| 06:00   0.0       0.50    0.10    0.40
          0.31      0.0     0.38    0.31
          0.25      0.25    0.0     0.50
          0.36      0.32    0.32    0.0

| 07:00   0.0       0.35    0.25    0.40
          0.30      0.0     0.40    0.30
          0.30      0.30    0.0     0.40
          0.28      0.44    0.28    0.0

| 08:00   0.0       0.10    0.50    0.40
          0.29      0.0     0.42    0.29
          0.34      0.33    0.0     0.33
          0.32      0.36    0.32    0.0

| 09:00   0.0       0.30    0.30    0.40
          0.32      0.0     0.36    0.32
          0.35      0.30    0.0     0.35
          0.30      0.30    0.40    0.0

| 10:00   0.0       0.30    0.30    0.40
          0.35      0.0     0.35    0.30
          0.35      0.30    0.0     0.35
          0.32      0.32    0.36    0.0
```