Geniş Ölçekli Ağlar İçin Yeni Bir Dağıtık Ayrık Olay Tabanlı Benzetim Yaklaşımı

New Discrete Event Based Distributed Simulation Protocol for Large-Scale Networks

Bülent Çobanoğlu Ahmet Zengin Hüseyin Ekiz
SAKARYA Üniversitesi, Teknoloji Fakültesi, Bilgisayar Mühendisliği Bölümü
bcobanoglu@sakarya.edu.tr; azengin@sakarya.edu.tr; ekiz@sakarya.edu.tr

Özetçe

Modelleme ve benzetim teorisi ağ tasarım ve iletişim kurallarını sınama ve dinamik ağ davranışını anlamada önemli bir araçtır. En büyük ağ olan İnternetin baş döndürücü bir hızla büyümesi, yeni benzetim tekniklerini geliştirmeyi zorunlu kılmaktadır. Yapılan çalışmada geniş ölçekli ağlar için ayrık olay tabanlı yeni bir dağıtık ağ benzetim yaklaşımı geliştirilmiştir. Ayrık olay tabanlı modelleme yaklaşımı kullanılarak istemci / sunucu tabanlı, ölçeklenebilir, birden fazla işletim sisteminde çalışabilen, esnek 'D-DEVSNET' isimli yeni bir ağ benzetim aracının modelleme ve tasarımı gerçekleştirilmiş ve başarım çözümlemeleri yapılmıştır. Geliştirilen aracın, yaygın olarak kullanılan ağ benzetim araçlarından NS-2 ile karşılaştırması yapılmış, özellikle ölçeklenebilirlik ve bellek tüketimi bakımından avantajları gözlemlenmiştir.

Anahtar Kelimeler

DEVS, Geniş Ölçekli Ağlar, D-DEVSNET, Dağıtık Benzetim, İstemci-Sunucu

Abstract

Modeling and simulation theory is an important tool for testing network design and protocols as well as understanding the dynamic behavior of the network. Growth of the biggest network, Internet, at dazzling speed requires new simulation techniques. In this study, a tool was developed for design and implementation of a new discrete event based distributed simulation for large-scale networks. By using DEVS as a modeling approach, client / server based, scalable, platform-independent, flexible new network modeling and simulation tool called 'D-DEVSNET' was designed and performance analyzes were conducted. Developed network simulator is compared NS-2, in particularly the advantages in terms of scalability and memory consumption were observed.

Keywords

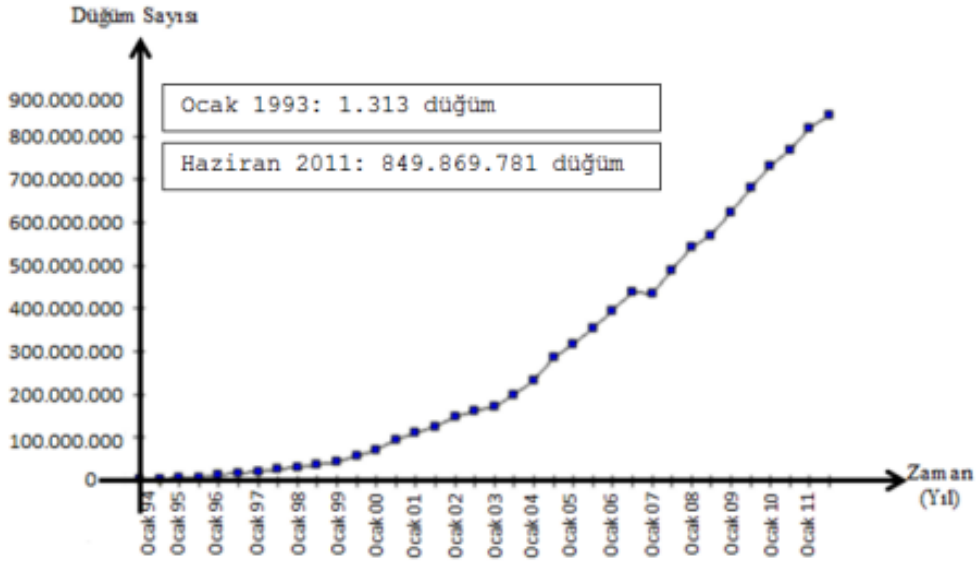
DEVS, Large Scale Networks, D-DEVSNET, Distributed Simulation, Client-Server

New Discrete Event Based Distributed Simulation Protocol for Large-Scale Networks

1 Giriş

Başlangıçta dört üniversite arasında bir sınama ağı olarak tasarlanan ilk ağ yapısı (ARPANET) [1], İnternet Sistemleri Konsorsiyumu'nun (ISC) altı ayda bir yayınlanan verilerine göre önümüzdeki birkaç

yıl içerisinde birkaç milyar düğümlü İnternet yapısına erişecektir (Şekil 1.) [2].



Şekil-1. İnternete bağlı düğüm / bilgisayar sayısı

Ağların bu şekilde hızla büyümesi, ağlarda ölçeklenebilirlik konusunun yeni anlamlar kazanması yanında, yeni kavramların / konuların teknolojiye eklenmesi ile sonuçlanmıştır. Ağ hızında ve işlem yapma gücündeki hızlı artış gereksinimi, ağ içerisinde gerçekleştirilen yönlendirme ve yönetim işlemlerinin merkezi bir yapıdan dağıtık bir yapıya doğru kaymasını zorunlu kılmaktadır. Başarım / maliyet oranının göz önünde tutulması zorunluluğu araştırmacıları, ağ üzerinde yeni işlem kapasitelerinin eklenmesi ve yeni yöntemlerin tasarlanması konusuna daha fazla yöneltmektedir.

Yapılan çalışmada, geniş ölçekli ağların benzetimi için yeni bir paralel ve dağıtık benzetim algoritması geliştirilmiş ve geliştirilen algoritmanın DEVS-Suite [3] benzetim ortamı altında modellemesi yapılmıştır. Modellenen benzetim aracına 'D-DEVSNET' ismi verilmiş ve yaygın İnternet iletişim kurallarından OSPF ve BGP örnek alınarak dağıtık bir şekilde çalıştırılmıştır. Ayrıca D-DEVSNET ortamının kapasitesini belirlemek ve DEVS yaklaşımının dağıtık uygulamalardaki gücünü göstermek amacıyla, farklı ölçekteki ağlar incelenerek benzer ağ benzetim araçları ile kıyaslaması yapılmıştır.

2 Dağıtık Geniş Ölçekli Ağ Benzetim Algoritması Geliştirme Süreci

Ağ tasarımı (topolojik yapısı, iletişim kurallarını, trafik akış denetimi, bağlantı teknolojisi, yönlendirme algoritmaları, vb.) oldukça karmaşık ve zor bir süreçtir. Tasarımcılar bu karmaşıklığı azaltmak için, belli bir soyutlama seviyesinde modelleme ve benzetim tekniklerinden yararlanırlar [4].

DEVS (Ayrık Olaylı Sistem Tanımı) modelleme yaklaşımının, hiyerarşik / modüler

bir yapıyı ve dağıtık çalışmayı desteklemesi, karmaşık geniş ölçekli sistemlerin (atomik ve bileşik modellerden oluşan) modellenmesinde kolaylıklar sağlamaktadır. DEVS modelleme yaklaşımı kullanılarak geliştirilen paralel ve dağıtık benzetim algoritmasında paralellik, paralel DEVS atomik ve birleşik model tanımı kullanılarak, dağıtık yaklaşım ise istemci-sunucu tabanlı mimari ile sağlanmıştır ve bu algoritma, DEVS tabanlı bir ağ benzetim aracının geliştirilmesinde kullanılmıştır.

Dağıtık ve geniş ölçekli bir ağ sisteminin modellenmesi işlemi;

- Gereksinimlerin belirlenmesini,
- Dağıtık ayrık olay tabanlı modelleme yaklaşımına sahip bir benzetim ortamının geliştirilmesini, diğer bir ifade ile ağ bileşenlerinin, bu bileşenlerde çalışacak yazılım nesnelerinin ve nesneler arasındaki etkileşimlerin tanımlanmasını,
- Yazılım nesnelerinin işlem yapan farklı coğrafik alanlardaki düğümlere dağıtılmalarını; sunucu-istemci uygulama alt yapısının kurulmasını,
- Ağ topolojileri ile yönlendirme iletişim kurallarının tanımlanmasını,
- Değişken ağ boyutu ve trafik şartları altında sınanmasını içermektedir.

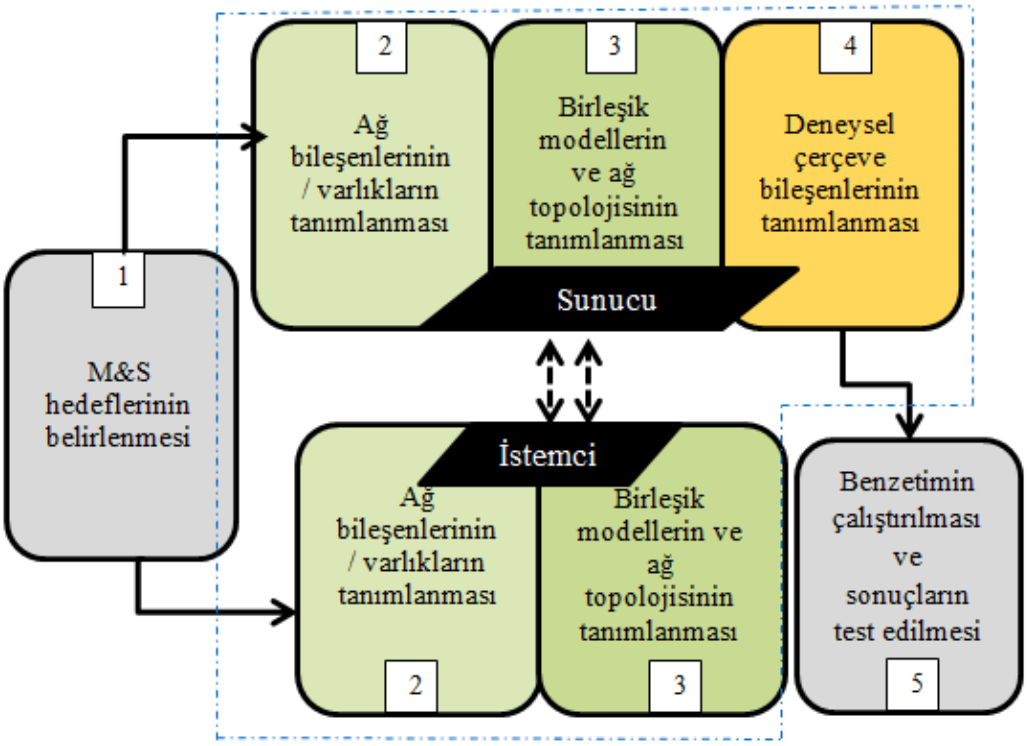
Buna göre DEVS tabanlı ağ modelleme ve benzetim ortamının gelişim/tasarım süreci Şekil 2'de gösterilmiştir.

Başlangıç aşamasında, modelleme ve benzetim hedefleri belirlendi. Modelleme hedefleri, modellenen sistemin tasarımında, yönetiminde ve denetiminde tanımlanan rollerle ilişkilidir. Hedefler ifadesi, model tasarlama işleminin belirli problemler üzerine odaklanması vazifesini görür.

İkinci aşamada, ağ tasarımı konusunda karar vermeyi sağlayan bir model ve ağ benzetim aracı yazılımı DEVS-Suite ve Java programlama dili ile geliştirildi. Bu aşamada benzetimi oluşturan temel ağ bileşenleri, varlıkları ve parametreleri istemci ve sunucu taraflı olmak üzere ayrı ayrı tanımlanır. Temel ağ bileşenlerinin tanımlanmasında, bir bağlantı durumu iletişim kuralı olan OSPF [5] düğümlere yerleştirildi. OSPF iletişim kuralını seçmemizin sebebi geniş ölçekli ağlarda RIP yerine OSPF yönlendirme iletişim kuralının kullanılmasıdır. Ayrıca otonom sistemler arası yönlendirme iletişim kuralı olarak BGP iletişim kuralı seçilmiş ve uygulanmıştır.

Üçüncü aşamada, geliştirilen varlıklar ve düğümler / yönlendiriciler birleştirilir. Bu bileşenler DEVS-Suite modelleme ve benzetim ortamında birbiriyle bağlanarak değişik topolojiler ve ağ yapıları meydana getirilebilir.

Dördüncü aşamada, modelleme ve benzetim hedeflerine ve geliştirilen modellere uygun bir deneysel çerçeve geliştirildi. Deneysel çerçeve, geliştirilen modelin bir takım koşullar altında sınanmasına ve gözlem yapılmasına yardımcı olur. Programlama aşamasında öncelikle istemci-sunucu modelleri ve genel bir ağ modelinin temel bileşenleri (düğümler / yönlendiriciler) belli bir soyutlama seviyesinde DEVS yaklaşımı kullanılarak oluşturuldu. OSPF ve BGP yönlendirme iletişim kuralları çalıştığı yüksek başarımlı, modüler ve hiyerarşik yapıda bir ağ benzetim aracı gerçekleştirildi. Daha sonra ise geliştirilen benzetim aracına BRITE [6] topoloji üretim aracı ile entegre edilerek büyük ölçeklerde topolojiler geliştirildi.



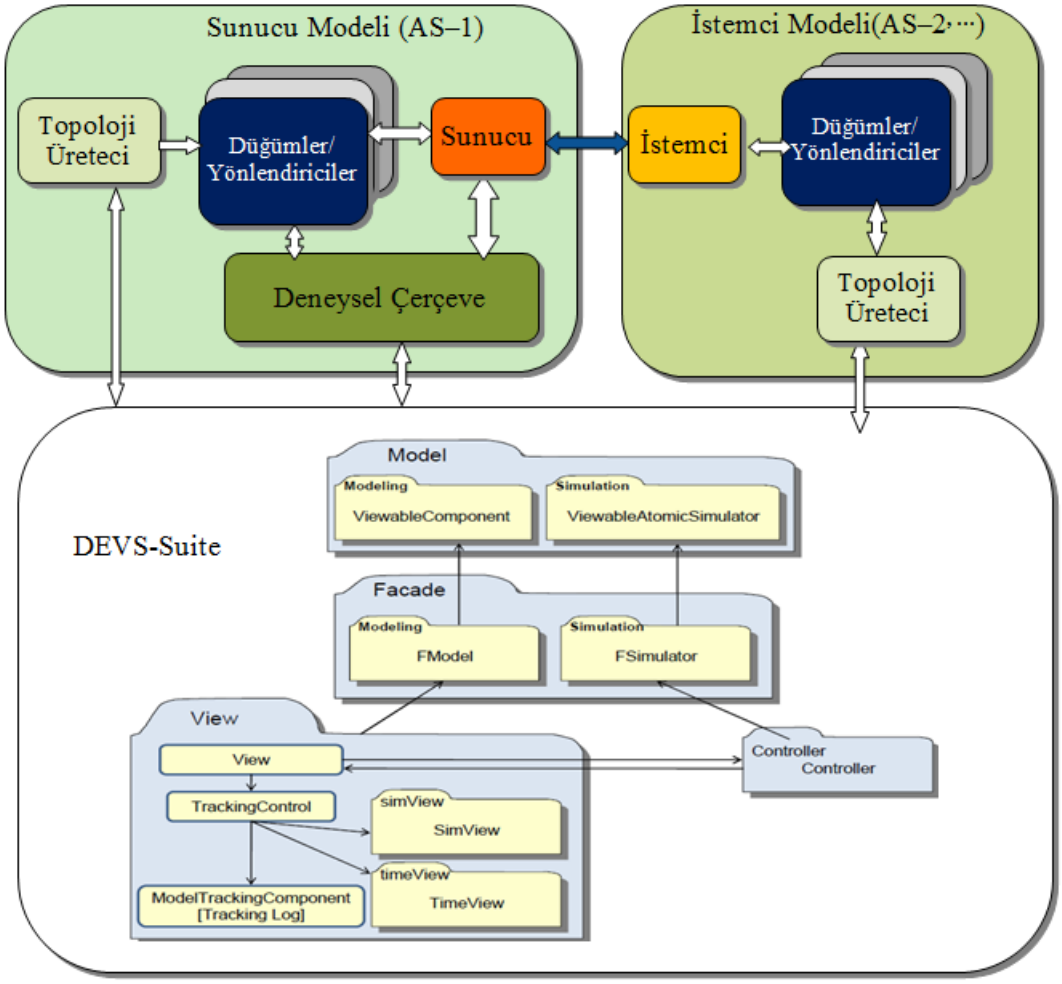
Şekil-2. İstemci - Sunucu mimarili dağıtık DEVS tabanlı ağ modelleme ve benzetim süreci

Son aşamada, benzetim aracı çalıştırılır ve sonuçlar DEVS-Suite ortamında gözlemlenir / izlenir. Sonuçlar kabul edilebilir bir aralıkta ise benzetim işlemi sonlandırılır. Aksi halde, tekrar başa dönülür ve geliştirilen modellerin parametreleri ayarlanır. Böyle bir süreçte ileri geri hareket edilerek en uygun model geliştirilmiş olur [7].

3 Ağ Bileşenlerinin Tanımlanması Ve Tasarımı

Geniş ölçekli DEVS tabanlı ağ benzetim modelindeki temel bileşenlerin tasarımına, paralel DEVS atomik modeli kullanılarak başlanmıştır. Ağ benzetim aracını modelleme amacına yönelik olarak oluşturulan düğümler ve bu düğümlerin haberleşmesini sağlayan diğer nesneler (IP paketleri, vb.), 'temel ağ bileşenleri' olarak tanımlanır. Daha sonra bu bileşenlerin bir araya gelmesi ile 'DEVS birleşik ağ modeli' oluşturulur [7].

Geliştirilen algoritmanın uygulandığı ağ benzetim modeli, sunucu ve istemci taraflı içerdiği ağ bileşenleri ile Şekil 3'te gösterilmiştir. Şekil 3'te görüldüğü gibi sunucu modelinin ağ bileşenlerini, otonom sistem (AS–1) içerisindeki düğümler / yönlendiriciler, sunucu, deneysel çerçeve ve topoloji üreteci oluşturmaktadır. İstemci modelinin ağ bileşenleri ise otonom sistem (AS–2) içerisindeki düğümler / yönlendiriciler, istemci ve topoloji üreteci oluşturmaktadır. Aynı zamanda hem sunucu tarafı hem de istemci tarafında temel ayrık olay işlemcisi olan DEVS çekirdeği yer almaktadır.



Şekil-3. Geliştirilen algoritmanın uygulandığı ağ benzetim modeli [7].

Geliştirilen ağ benzetim modelinde yer alan ağ bileşenleri sunucu ve istemci taraflı olarak ayrı ayrı tanımlanmıştır. Sunucu ve istemci düğümler üzerinden bağlı bulundukları otonom sistemlerin trafik geçişlerine imkân sağlayacak, bir anlamda transit geçiş hizmeti veren düğümler şeklinde tasarlanmışlardır.

3.1 Sunucu düğüm atomik modeli

Şekil 3'ten de görüleceği üzere sunucu düğümün istemci düğümden tek farkı deneysel çerçeveye sahip olmasıdır. Dolayısı ile bütün trafik sonuçları sunucu tarafındaki tek bir dosyada tutulmaktadır. Sunucu düğüm içerisinde ağ arabirimi (Nerwork Interface Card - NIC), DEVS varlıkları, veri ve denetim paketleri yer almaktadır. Sunucu düğüm, aynı zamanda istemci ile bağlantıyı sağlayan soket

yapısına sahiptir. DEVS tabanlı bir sunucu düğüm atomik modeli (M_{server_node}) matematiksel olarak;

M_{server_node} = <X, Y, S, δext, δint, δcon, λ, ta >

şeklinde tanımlanır. Sunucu düğümün davranış (çalışma mantığı) algoritmasını ise aşağıdaki gibi ifade edebiliriz;

1. Başla,
2. Sunucu düğüm ağ ara yüzü giriş port değişkenlerini tanımlama ve başlangıç değerlerini ata,

X = inport ×invalues
invalues : {packet, HELLO, LSA };
inports : { in, inF};

3. Sunucu düğüm ağ ara yüzü çıkış port değişkenlerini tanımlama ve değer ata,

Y = outport ×outvalues
outvalues : {packet, , HELLO, LSA };
outports : {out, outF };

4. Sunucu düğüm durum değişkenini ve alacağı değerleri belirle,

S= phase × σ
phase :{ 'startup', 'idle', 'p_prep_to_server'}
$\sigma = \Re^{+}_{0,\infty}$

5. Harici durum değişkenini (δext) ayarla;

Eğer giriş port değeri 'in' ve gelen paket LSA paketi değilse paketi ayrıştır ve istemciye gönder,
Eğer giriş port değeri 'inF' ise paketi ayrıştır ve istemciye gönder,

δext ((phase, σ), e, X))=

$$\begin{cases} \text{if packet} \neq \text{LSA and output } x \leftarrow (in), \\ \quad \text{packetDeAssembly} \\ \text{else if output } x \leftarrow (inF), \\ \quad \text{packetDeAssembly} \\ \text{//paketi istemciyegönder} \end{cases}$$

6. Dâhili durum değişkenini (δint) ayarla;

Eğer kuyruk boş değilse paketi kuyruktan al ve durum değişkenini (S) *'p_prep_to_server' yap,*
Değilse durum değişkenini (S) *'idle' yap,*

δint (phase, σ , Q) =

$$\begin{cases} \text{if qFromNet} \neq \text{isEmpty}, \\ \quad \text{packet.remove and} \\ s \leftarrow (\text{'p\_prep\_to\_server'}, \sigma', x) \\ \quad \text{else} \\ \quad s \leftarrow (\text{'idle'}, \sigma', x) \\ \quad \text{where } s \in S \end{cases}$$

7. Çakışma durumunda önce dâhili durum geçiş fonksiyonunu (δint) çalıştır,

δcon ((phase, σ), e, X)) =δint (δext (phase, σ), 0, X))

8. Çıkış fonksiyonunu (λ) ayarla;

Eğer portlar iletişime hazır ve paket de varsa paketleri birleştir,
Eğer paket boş değilse ve port ismi 'inF' veya 'outEvent' ise çıkış port değerini 'outF' değilse 'out' yap,

λ(phase, σ , Q) =
if inPort=isReady(),
rPacket←(inPort.readLine()),
packetAssembly
if rPacket≠null, and
getPort_name='"outEvent'||'inF'",
output y←(outF, null) else y←(out,null)
where y ∈Y)

9. Zamanı (ta) ilerlet,
10. Dur.

3.2 İstemci düğüm atomik modeli

İstemci düğümü, sunucu taraflı ağla iletişimi sağlamakla görevli bir yönlendirici olarak tasarlanmıştır. İstemci düğüm içerisinde ağ arabirimi, DEVS varlıkları, veri ve denetim paketleri yer almaktadır. İstemci düğüm ile sunucu arasındaki veri iletiminde Java soket yapısı kullanılmıştır. Sunucu ve istemci düğümleri otonom sistemleri birbirlerine

bağlayan sınır düğüm işlevine sahiptir. DEVS tabanlı bir istemci düğüm atomik modeli (M_{client_node}) matematiksel olarak;

M_{client_node} = <X, Y, S, δext, δint, δcon, λ, ta >

şeklinde tanımlanır. İstemci düğümün davranış (çalışma mantığı) algoritmasını ise aşağıdaki gibi ifade edebiliriz;

1. Başla,
2. İstemci düğüm ağ ara yüzü giriş port değişkenlerini tanımlama ve başlangıç değerlerini ata,

X = inport ×invalues
invalues : {packet, HELLO, LSA };
inports : { in, inF};

3. Sunucu düğüm ağ ara yüzü çıkış port değişkenlerini tanımlama ve değer ata,

Y = outport ×outvalues
outvalues : {packet, , HELLO, LSA };
outports : {out, outF };

4. Sunucu düğüm durum değişkenini ve alacağı değerleri belirle,

S= phase × σ
phase :{'idle', 'startup', 'p_prep_to_server' , 'p_prep_to_network'}
$\sigma = \Re^{+}_{0,\infty}$

5. Harici durum değişkenini (δext) ayarla;

Eğer giriş port değeri 'in' ve gelen paket LSA paketi değilse paketi ayrıştır ve sunucuya gönder,
Eğer giriş port değeri 'inF' ise paketi ayrıştır ve sunucuya gönder,

δext ((phase, σ), e, X))=

```
if packet ≠ LSA and output x ← (in),
        packetDeAssembly
    else if output x ← (inF),
        packetDeAssembly
    //paketi sunucuya gönder
```

6. Dâhili durum değişkenini (δint) ayarla;

Eğer paket boş değilse durum değişkenini (S) *'p_prep_to_net' yap,*
Eğer kuyruk boş değilse paketi kuyruktan al ve durum değişkenini (S) *'p_prep_to_server' yap,*
Değilse durum değişkenini (S) *'idle' yap,*
δint (phase, σ , Q) =

```
if rpacket ≠ null,
  s ← ('p_prep_to_net', σ' ,x)
else if qFromNet ≠ isEmpty,
    packet.remove and
  s ← ('p_prep_to_server', σ' ,x)
        else
    s ← ('idle', σ' ,x)
      where s ϵ S
```

7. Çakışma durumunda önce dâhili durum geçiş fonksiyonunu (δint) çalıştır,

δcon ((phase, σ), e, X)) =δint (δext (phase, σ), 0, X))

8. Çıkış fonksiyonunu (λ) ayarla;

Eğer portlar iletişime hazır ve paket de varsa paketleri birleştir,
Eğer paket boş değilse ve port ismi 'inF' veya 'outEvent' ise çıkış port değerini 'outF' değilse 'out' yap,

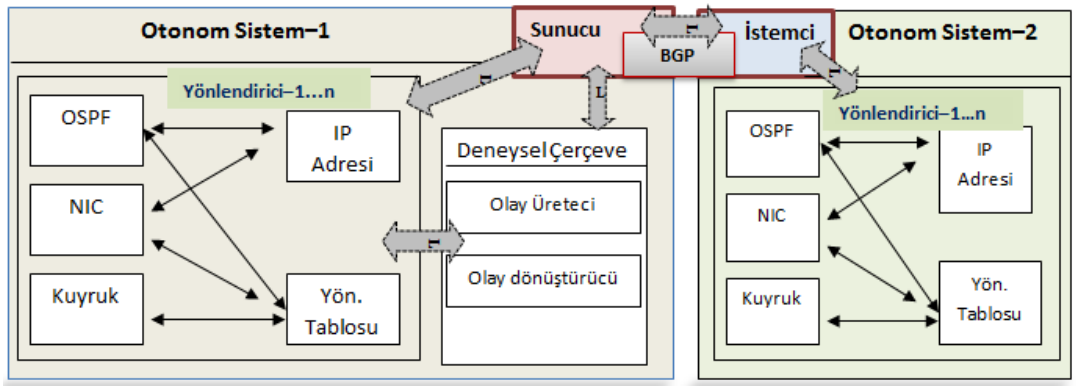
λ(phase, σ , Q) =
if inPort=isReady(),
rPacket←(inPort.readLine()),
packetAssembly
if rPacket≠null, and
getPort_name='"outEvent'||'inF'",
output y←(outF, null) else y←(out,null)
where y ∈Y)

9. Zamanı (ta) ilerlet,
10. Dur.

3.3. Otonom Sistem modeli

D-DEVSNET benzetim ortamı istemci ve sunucu taraflı olmak üzere iki otonom

sistemden oluşmaktadır. Ancak istenildiğinde daha fazla istemci oluşturularak ikiden fazla otonom sistem rahatlıkla oluşturulabilir. Otonom sistemler, kendi içerisinde özerk olan sistemlerdir. Günümüz İnternetinin trafik altyapısında otonom sistem içerisinde OSPF iletişim kuralı kullanılırken otonom sistemler arasında ise BGP iletişim kuralı kullanılır. Benzer şekilde D-DEVSNET benzetim ortamında da otonom sistem içerisinde DEVS tabanlı OSPF iletişim kuralı koşturulurken, otonom sistemler arasında BGP iletişim kuralı koşturulur / çalıştırılır. BGP, otonom sistemleri birbirine bağlamak amacı ile otonom sistemler arası bağlantıda veri trafiğinin hangi yönlendiriciler üzerinden gerçekleştirileceğine karar verilmesini sağlayan bir yönlendirme iletişim kuralıdır. Şekil 4'te D-DEVSNET otonom sistem birleşik modeli görülmektedir. Şekilde görüldüğü üzere Sunucu ve İstemci düğümleri otonom sistemleri birbirlerine bağlayan sınır düğümlerdir. Bu düğümler arasındaki veri/paket trafiğini BGP iletişim kuralı ile sağlanır.



L:Link(Hat)

Şekil 4. D-DEVSNET otonom sistem birleşik modeli

4 Geliştirilen Algoritmanın Geçerlemesi

Geliştirilen algoritma, istemci - sunucu tabanlı D-DEVSNET [7] adı verilen bir ağ benzetim aracının tasarımında kullanılmıştır. Algoritma başarımını ölçmek için D-DEVSNET ile NS-2 ağ benzetim aracı arasında bir karşılaştırma yapılmıştır.

Çizelge-1'de benzer parametrelerle (iletişim kuralları, bant genişliği, paket boyutu, vs. gibi) karşılaştırılan benzetim araçlarının olay sıklığı ve bağlantı başına harcadıkları bellek miktarı Çizelge-2'de verilmiştir. Buna göre 10 saniyelik benzetim süresinde D-DEVSNET' de 1000 olay olurken NS-2'de 28388 olay kaydedilmiştir. Benzetim yazılımı, benzetim süresince paketin başından geçen her şeyi olay (event) olarak kaydeder. Paketin iletim hattından iletilmesi, kuyruğa alınması, kuyruktan ayrılması, işlenmesi, yönlendirilmesi, iletim hattından düşmesi gibi durumlarının tamamı birer olay olarak kaydedilir. Yine yaptığımız başka bir çalışmada D-DEVSNET çalışırken elde edilen bağlantı başına bellek tüketimi miktarının, NS-2' ye göre daha iyi olduğu görülmüştür [7].

Çizelge-1: Karşılaştırılan ağ benzetim araç parametreleri

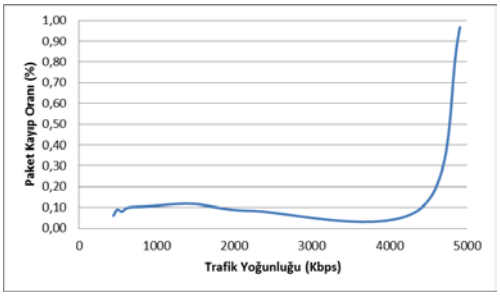
Benzetim Parametreleri	**D-DEVSNET**	**NS–2**
Paket Boyutu (byte)	552	552
Bant Genişliği (Mbps)	2	2
Gecikme (msn)	1	1
İletişim kuralları	OSPF	OSPF
Kuyruk Limiti (paket)	362	20
Süre (sn)	10	10

Çizelge-2: Karşılaştırılan benzetim araçlarının olay sıklığı ve bellek tüketimi

Benzetim Aracı	**D-DEVSNET**	**NS-2**
Olay Sayısı	1000	28388
Bağlantı başına bellek tüketimi	42,8 KB	93,3 KB

D-DEVSNET, gereksiz ayrıntıların göz ardı edildiği yüksek soyutlama seviyesine (daha az karmaşıklığa) sahip bir ağ benzetim aracı olduğu için olay sayısı oldukça düşüktür (NS-2'ye göre 30 kat daha az). Çizelge-2'den görüleceği üzere aynı işi yapmak için D-DEVSNET, NS-2'ye göre çok daha az olay üretir, böylece büyük boyutlu uygulamalarda doğruluktan fazlaca ödün verilmeden bellek ve işlemci kullanımı minimize edilebilir.

Şekil 5'te ise geliştirilen benzetim aracının uç / sınır değerlerdeki davranışını sınamak amacıyla 3000 düğümlü farklı trafik koşullarındaki (442Kbps–4,5Mbps arasındaki) bir ağda paket kaybı incelenmiştir. Bu grafiğe göre D-DEVSNET modelinde, yaklaşık 4,4 Mbps' da trafik yoğunluğu kararlı hale geliyor. Bu değerden sonra model doğruluğunu ve kararlılığını kaybediyor.



Şekil-5. 3000 düğümlü bir D-DEVSNET ağı için paket kaybı

Ağ benzetim araçlarının hız ve bellek tüketimi karşılaştırmasına yönelik birçok çalışma vardır. Bu konudaki literatür araştırmasında genelde farklı ağ benzetim araçlarına ait hız ve bellek tüketimi karşılaştırmaları yapılmıştır. Nicol [8], NS-2, JavaSim (J-Sim) ve SSFNet benzetim araçlarını 1000 saniyelik sürede bağlantı sayısını dikkate alarak (10-10000 arası) karşılaştırmıştır. Nicol bu çalışmasında, NS-2 benzetim aracının daha hızlı fakat daha fazla bellek tüketimine sahip olduğunu belirtmiştir. Weingartner ve arkadaşlarının yapmış olduğu çalışmada [9] NS-2, NS-3, OMNeT++, SimPy ve JİST benzetim araçları 600 saniyelik sürede düğüm sayıları (4-3025 düğüm) dikkate alınarak karşılaştırılmıştır. Bu çalışma sonucunda NS-2, OMNET++ ve SimPy araçlarının benzer bellek tüketimine sahip olduğu, JİST'in daha hızlı fakat daha fazla bellek tüketimine sahip olduğu vurgulanmıştır. Albeseder ve arkadaşlarının çalışmasında [10] OMNET++, NS-2, QualNet ve SimPy araçları çalışma zamanı süreleri dikkate alınarak karşılaştırılmıştır. Bu çalışmada da benzer sonuçlar elde edilmiş, özellikle OMNET'in hız bakımından başarımının iyi olmadığı vurgulanmıştır. Fujimoto ve arkadaşları [11], NS-2, PDNS ve GTNeTS benzetim araçlarını karşılaştırmış ve NS-2'nin daha hızlı çalıştığını belirtmiştir. Tüm bu çalışmalardan, daha hızlı çalışan benzetim araçlarının daha fazla bellek kapasitesine ihtiyaç duyduğunu söyleyebiliriz.

Ayrıca yaptığımız başarım çözümlemelerinde bir bilgisayarda NS-2 ile en fazla 1500 düğüme çıkılabilirken D-DEVSNET ile yaklaşık 3000 - 3500 düğüme çıkılabilmektedir. Bu da D-DEVSNET'in iyi bir ölçeklenebilirliğe sahip olduğunu göstermektedir.

5 Sonuçlar Ve Öneriler

Geniş ölçekli ağların benzetimi için yeni bir DEVS tabanlı paralel ve dağıtık benzetim algoritması geliştirilmiştir. DEVS yaklaşımına ve dağıtık istemci/sunucu mimarisine sahip algoritma ile ağ bileşenlerinin kolaylıkla coğrafik olarak farklı ağlar üzerine bölünebilmesi ve böylece oldukça büyük ölçekte ağ modellerinin benzetim çalışmasının yapılması mümkün olacaktır.

Geliştirilen algoritma kullanılarak yeni bir ağ benzetim aracı geliştirilmiştir. Geliştirilen ağ benzetim aracı bazı özellikleri ile basit kabullenmelere ve soyutlamalara tabii tutulmuştur. Sadece OSPF değil diğer yönlendirme iletişim kurallarının (RIP gibi), kablosuz ve duyarga ağlarda kullanılan iletişim kurallarının benzetim ortamına eklenmesi ile genel bir ağ benzetim aracı gerçekleştirilebilir. Bu soyutlamaların gerçeğe uygun bir şekilde modellenmesi ile geniş ölçekli bir ağın tüm yönlerini değerlendiren eksiksiz bir ağ benzetim aracı geliştirilebilir.

Java programlama dili kullanılarak gerçekleştirilen uygulamaların, web tarayıcılar altında rahatlıkla çalışabilmesi, D-DEVSNET benzetim aracının da İnternet üzerinden kullanılmasını sağlamakta ve uzaktan eğitim amaçlı uygulamalar için ideal bir altyapı oluşturmaktadır. Geliştirilen D-DEVSNET benzetim aracı, web ortamına taşınarak popüler tarayıcılar üzerinden modelleme ve benzetim çalışmalarına imkân sağlanabilir. Modelleme ve benzetim uygulamalarında web tabanlı araçların kullanım gereksinimi göz önüne alındığında yapılan çalışma web tabanlı geniş ölçekli ağ benzetim araçlarının tasarımına da öncülük edecektir.

6 Kaynakça

[1] **ROBERTS, L.G.,** The ARPANET and Computer Networks, A History of Personal Workstations, A. Goldberg, Adele, ed. New York: ACM, 1986.

[2] İnternet Sistemleri Konsorsiyomu, www.isc.org/solutions/survey, Erişim tarihi: Kasım 2011.

[3] **KIM, S., SARJOUGHIAN H. S., ELAMVAZHUTHI, V.,** DEVS-Suite: A Simulator Supporting Visual Experimentation Design and Behavior Monitoring, Spring Simulation Multiconference, Article no 161, San Diego, California, 2009.

[4] **RAHMAN, M.A., PAKŠTAS, A., WANG, F. Z.,** Network Modelling and Simulation Tools, Simulation Modelling Practice and Theory, 17, pp. 1011-1031, 2009.

[5] **ZENGİN, A.,** Large-Scale Integrated Network System Simulation with DEVS-Suite, KSII, Transactions on Internet and Information Systems Vol. 4, No. 4, 2010.

[6] BRITE Web adresi: http://www.cs.bu.edu/brite/, Erişim tarihi: Şubat 2012.

[7] **ÇOBANOĞLU, B.,** Geniş Ölçekli Ağlar İçin Yeni Bir Dağıtık Ayrık Olay Tabanlı Benzetim Yaklaşımı ve Uygulaması, Doktora Tezi, Fen Bilimleri Enstitüsü, Sakarya Üniversitesi, 2011.

[8] **NICOL, D. M.,** Scalability of Network Simulators Revisited, In Proceedings of the 2003 Conference on Networked and Distributed Systems (CNDS), Orlando, FL, January 2003.

[9] **WEINGARTNER, E., LEHN, H., WEHRLE** K., A Performance Comparison of Recent Network Simulators, ICC 2009: IEEE International Conference on Communications, Dresden, Germany, 2009.

[10] **ALBESEDER D., FUEGGER M.**, Small PC-Network Simulation – A Comprehensive Performance Case Study, Research Report, TU Wien, Institut für Technische Informatik, 2005.

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