Yazılım Tanımlı Ağlar İçin Güç Verimli Yol Atama
Flow Based Power Efficient Routing for Software Defined Networks

Yiğitcan Aydoğmuş[1], Berna Özbek[1], Onur Koyuncu[2], Kazım Ulusoy[2], Özgür Karakaya[2]
[1]Elektrik ve Elektronik Mühendisliği Bölümü, İzmir Yüksek Teknoloji Enstitüsü, İzmir Türkiye
{yigitcanaydogmus,bernaozbek}@iyte.edu.tr
[2]Argela A.Ş. İstanbul, Türkiye
{onur.koyuncu,kazim.ulusoy,ozgur.karakaya}@argela.com.tr

***Özetçe* —Mobil uygulamaların gittikçe yaygınlaşması ile artan trafik çeşitliliği ve hacmi, ağlarda taşınan trafiğin yönetilmesi ihtiyacını kuvvetlendirdi. Yazılım tanımlı ağlar, trafik yönetimini kullanarak belirlenen gereksinimleri karşılarken, verimi maksimuma çıkararak ağları yönetebilir. Bu bildiride, ağdaki aktif anahtar sayısına dayanan güç tüketimini minimuma indiren bir yol atama algoritması öneriyoruz. Bağlantı kapasitesi kısıtlamalarını göz önüne alarak, akışların veri hacmi gereksinimlerini karşılayan en iyi yolu bulmak için genetik algoritma kullanıp, düşük karmaşıklıklı yeni bir yol atama yaklaşımı öneriyoruz. Önerilen algoritmanın verilen ağ topolojisinde çeşitli akış veri hacmi kısıtlamalarına göre performans değerlendirmelerini sunuyoruz.**

Anahtar Kelimeler—Yazılım tanımlı ağ, yol atama, genetik algoritma.

***Abstract*—Since traffic diversity and volume increase with growing popularity of mobile applications, there is the strong need to manage the traffic carried by networks. Software defined networking can manage network while enabling new services by employing traffic management whose goal is to maximize the utility objective while satisfying given requirements. In this paper, we propose an efficient routing to minimize the cost based on power consumption determined by the number of active switches in a software defined networks. An optimum solution obtained by genetic algorithm and a reduced complexity routing approach are proposed while satisfying throughput requirements of flows for given constraints on link capacities in the network. We provide the performance evaluations of the proposed algorithms with respect to different throughput constraints of flows in a given network topology.**

Keywords—component, formatting, style, styling, insert (key words).

I. GİRİŞ

Yazılım tanımlı ağlar, ağ kontrol düzlemini bağımsız bir yazılım olarak ayırmaktadır. Bu özellik gezgin ağlar gibi çok yönlü ortamlar için yararlıdır [1] [2]. Geleneksel ağlar anahtar ve yönlendiricilerden oluşur. Yazılım tanımlı ağ, kontrol düzlemini veri düzleminden ayırarak ağ inovasyonlarını hızlandırma potansiyeline sahiptir. Yazılım tanımlı ağların kontrol düzlemi merkezileştirilmiştir ve bütün kontrol kararlarından sorumludur. Ayrıca bu kontrol bilgisini veri düzlemine gönderir. Bu mimaride kontrol birimi ağa tamamen hakim olduğu için bütün ağı en iyileyebilmektedir.

Trafik yönetimi, ağ performansını gönderilen verinin davranışını analiz ve tahmin edip, düzenleyerek en iyileme tekniğidir. Yol atama en iyilemesini ve güç kullanım kontrolünü kapsar. Yazılım tanımlı ağlar, ağa geniş çaplı bakış açısı sağlar. Ağın durumu ve akış karakteristiğinden faydalanarak özgün trafik yönetimi teknikleri sunar. Trafik yönetiminin amaçlarından biri, çeşitli performans kriterlerine göre ağdaki trafiği nasıl yönlendireceğine karar vermektir. Bu amaçlar, veri hacmini maksimuma çıkarma, bağlantı kullanımını dengeleme, güç tüketimini minimuma indirgeme, bant genişliğini kontrol etme, bekleme süresini azaltma, ağdaki trafiğin değişmesi veya ağın bir kısmının hata vermesi gibi durumlarda bile işleyişi güvenilir kılma olabilir. [3] 'da, bant genişliğini ve akış tablosunu göz önüne alarak kontrol uygulamaları için gecikmeyi minimuma indirgeyen adil bir yerleştirme modeli sunulmuştur. [4]'te akış bazlı uçtan uca hizmet kalitesini en iyileyen dağıtılmış çözüm sunulmuştur. Çoklu trafik matrisleri için en iyiye yakın yük dengelemeye ulaşmak için düşük karmaşıklıklı melez bir yol atama şeması [5]'te verilmiştir. Kabul edilebilir performans düzeyini korurken her akış için aktif ağ aygıtı sayısını azaltan enerji verimli yol atama şeması [6]'da sunulmuştur. [7]'de, ağ nispeten boşken trafiği alternatif yollardan yeniden yönlendirip bağlantı yüklerini dengeleyerek güç yönetiminin nasıl en iyilenebileceğini anlatılmıştır. [8]'de, bütün anahtarların sürekli aktif olması yerine en az sayıda ağ aygıtı kullanarak trafiği taşıyan enerji tasarruf mekanizması önerilmiştir. Bu çalışmada, problemi MILP(Mixed-integer linear programming) olarak formüle edip 4 farklı strateji kullanan sezgisel algoritma verilmiştir.

Bu bildiride, ağdaki trafiğin düşük olduğu gece saatlerinde aktif anahtar sayısına dayalı güç tüketimini minimuma indirgeyen yol atama algoritması öneriyoruz. Bu amaç doğrultusunda, atlama sayısını minimuma indirgemek her zaman yeterli olmayabilir. Bazı durumlarda yeni anahtarlar aktif etmek gerekebilir. Belirlenen bağlantı kapasitelerine göre akışların veri hacmi ihtiyaçlarını karşılayan genetik algoritma çözümü ve basitleştirilmiş bir yaklaşım öneriyoruz. Verilen ağ topolojisindeki çeşitli veri hacmi ihtiyaçlarına göre önerilen algoritmanın güç tüketim performansını değerlendiriyoruz.

II. bölümde genel ağ modeli verilmektedir. III. bölümde

978-1-5090-1679-2/16/$31.00

genetik algoritma ve basitleştirilmiş yaklaşım detaylarıyla anlatılmaktadır. IV. bölümde önerilen algoritmanın performans değerlendirmesi ve V. bölümde sonuç başlığı yer almaktadır.

II. AĞ MODELİ

Yazılım tanımlı ağ $G < \mathbb{V}, \mathbb{Z} >$ çizgesi şeklinde modellenebilir. $\mathbb{V}$ anahtar setini and $\mathbb{Z}$ ise anahtarlar arasındaki müsait portlardan kurulabilecek bağlantı setini temsil eder. u ve t anahtarları arasında müsait port sayısına ve toplam kapasite $C_{(u,t)}$'ye bağlı olarak $N_{u,t}$ adet bağlantı kurulabilir. Her bir bağlantı $(u_x, t_x) \in \mathbb{Z}$, $C_{(u_x,t_x)}$ kapasitesine sahiptir ve $\sum_{x=1}^{N_{u,t}} C_{(u_x,t_x)} = C_{(u,t)}$ eşitliğini sağlar. Atanan akış veri hacmine göre u_x ve t_x arasındaki bağlantı kapasitesinin dinamik olarak ayarlanabileceği varsayılmıştır.

$\mathbb{F}$ akış seti, K sayıda akışa sahip olmak üzere her bir akış $f_k \in \mathbb{F}$ bir kaynağa, s_{f_k}, bir varış noktasına, d_{f_k}, ve veri hacmi gereksinimine, R_{f_k}, sahiptir. Bizim çözümümüz $\mathrm{p}_{f_k} \in \mathbb{Z}$ 'nin kaynak ve varış noktası arasındaki yolu temsil ettiği, $\mathbf{p} = \{\mathrm{p}_{f_1}, \mathrm{p}_{f_2}, ..., \mathrm{p}_{f_K}\}$ yollarını bulmaktır. Bu yollar, her akışın veri hacmi gereksinimlerini karşılarken ağdaki aktif anahtar sayısını minimuma indirgeyecek şekilde seçilmektedir.

En iyileme problemi aşağıdaki gibi tanımlanabilir.

$$\min \quad T(\mathbf{p}) \tag{1}$$

bağlı olarak

$$C_{(u_x,t_x)\in \mathrm{p}_{f_k}} \geq R_{f_k} \quad \forall f_k \tag{2}$$

$$\sum_{x=1}^{N_{u,t}} C_{(u_x,t_x)} \leq C_{u,t} \quad \forall u,t \tag{3}$$

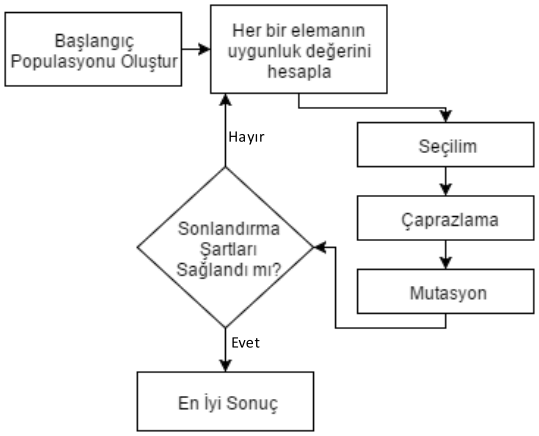
$T(\mathbf{p})$ ağdaki bütün akışlara yol atamaları tamamlandıktan sonra aktif durumdaki toplam anahtar sayısıdır.

İlk kısıt, akışlara atanan bağlantılarının kapasitesinin akışların veri hacmi gereksinimlerine eşit ya da daha büyük olması gerektiğini söyler. İkinci kısıt u ve t anahtarları arasındaki toplam bağlantı kapasitesinin ($C_{u,t}$), kurulan $N_{u,t}$ bağlantıya atanan kapasite toplamına eşit ya da bu toplamdan büyük olması gerektiğini söyler. Daha önce bahsedildği gibi $C_{(u_x,t_x)}$ kapasitesinin, akışların veri hacmine göre ayarlandığı varsayılmıştır.

Ağ trafiğinin düşük olduğu gece saatlerini değerlendirdiğimiz için ağın toplam kapasitesi, veri hacmi gereksinimlerine oranla çok daha fazladır. Bu sebeple belli sayıdaki anahtarları kapatmak mümkün olabilmektedir.

III. ÖNERILEN YOL ATAMA ALGORITMASI

En iyi yol atamayı yapabilmek için genetik algoritma (GA) kullanılmıştır. Genetik algoritma biyolojik evriminin seçilim(selection), çaprazlama(crossover) ve mutasyon(mutation) gibi özelliklerinden esinlenen bir arama metodudur. Şekil 1'de genetik algoritmanın genel işleyişi verilmiştir. Çaprazlama olasılığı, P_c, mutasyon olasılığı, P_m, popülasyon büyüklüğü, M, gibi parametreler ayarlanarak genetik algoritmanın yerel (local) çözüme yakınsaması önlenebilir ve böylece evrensel (global) çözüme ulaşabilir.



Şekil 1: Genetik Algoritma

A. Genetik Algoritma Çözümü

En iyi yollar, $\mathbf{p}$, genetik algoritma ile denklem (2) ve denklem (3) göz önüne alınarak denklem (1) 'de tanımlanan T'yi minimuma indirgeyerek elde edilmiştir.

- *Başlangıç popülasyonu oluşturma:* Rastgele M adet birey oluşturulur. Her bir birey K adet farklı yola sahip ve bütün nesiller olası bir çözüm adayıdır. Bu sebeple, bütün bireyler denklem (2) ve denklem (3)'de verilen kısıtları sağlamak zorundadır.
- *Seçilim:* Bireyler, denklem (1)'de tanımlanan toplam aktif anahtar sayısına göre sıralanıp en iyi $M/2$ tanesi seçilir.
- *Çaprazlama:* Akışlara atanan yolların orta noktasından tek noktalı çaprazlama yapılmıştır. 2 bireyde de bulunan ortak bir anahtar seçilip bu noktadan sonraki değerler karşılıklı yer değiştirir. Oluşturulan yeni yollar popülasyona eklenir.
- *Mutasyon:* Rastgele seçilen bir anahtarın, yerine gelebilecek başka bir anahtar ile değiştirilmesidir.
- *Yakınsama:* Yakınsama kriteri, belirlenen sayıda değişmeden tekrar eden aktif anahtar sayısı olarak seçilmiştir.

B. Düşük Karmaşıklı Yaklaşım

En iyi yol atama çözümünün ağdaki anahtar ve akış sayısına göre giderek artan karmaşıklığını azaltmak için basitleştirilmiş verimli bir yol atama algoritması öneriyoruz. Düşük karmaşıklıklı yaklaşımda, akışlara genetik algoritma çözümündeki gibi eş zamanlı olarak değil sırayla yol atanır. Bu yöntem aynı zamanda akış isteklerinin geldiği ve bırakıldığı haberleşme sistemlerinde gerçek zamanlı olarak kullanılabilir.

Her bir akışa yol atandıktan sonra bağlantı kapasiteleri aşağıdaki gibi güncellenir:

$$C_{(u,t)} = C_{(u_x,t_x)\in \mathrm{p}_{f_y}} - R_{f_y}. \tag{4}$$

Her bir akış için $f_k \in F$, en iyileme problemi

$$\min \quad T(\mathrm{p}_{f_k}) \tag{5}$$

$$C_{(u_x,t_x)\in \mathrm{p}_{f_k}} \geq R_{f_k} \tag{6}$$

denklem (6)'ya bağlı olarak tanımlanır.

Bu problemi çözmek için aşağıdaki yöntem izlenmiştir.

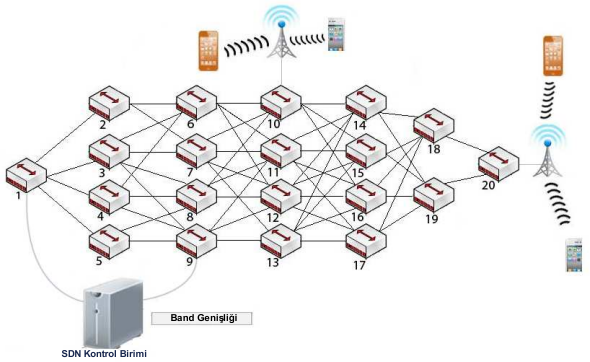
- *Başlangıç*: $S = \emptyset$ aktif anahtar seti olarak tanımla.
- *İlk akış için*, f_1:
  - Denklem (5)'e göre denklem (6)'yı da sağlayan en iyi yol (p_{f_1}), GA ile bulunur.
    Genetik algoritmada, başlangıç popülasyonu sadece ilk akış için rastgele üretilmiş denklem (6)'da verilen kısıtı sağlayan $\bar{M}$ adet bireye sahiptir. $\bar{M}/2$ bireyin seçilimi denklem (5)'e göre yapılır ve her bir iterasyonda tek noktalı çaprazlama ve mutasyon uygulanır.
  - p_{f_1}'deki bütün anahtarları S setine eklenir.
  - Ağdaki bağlantı kapasitelerini denklem (4)'e göre güncellenir.
- $k = 2, \cdots, K$ için
  - Genetik algoritma kullanarak S setindeki anahtar kullanımını maksimuma çıkaran yol bulunur.
    Genetik algoritmada, eğer kaynak, varış noktası ve veri hacmi kısıtları sağlanıyorsa önceki atanan k adet yol p_{f_k} başlangıç popülasyonuna eklenir ve denklem (6)'ya göre popülasyon doldurulur. Amaç mümkün olduğunca yeni anahtar aktif etmemek olduğu için $\bar{M}/2$ adet birey, S seti kullanımını maksimuma çıkaranlardan seçilir. Her bir iterasyonda tek noktalı çaprazlama ve mutasyon uygulanır.
  - p_{f_k}'daki yeni anahtarlar S setine ekle.
  - Ağdaki bağlantı kapasitelerini denklem (4)'e göre güncellenir.
- Bitiş.

IV. BENZETİM SONUÇLARI

Kullanılan 20 anahtarlı ağ topolojisi Şekil 2'de gösterilmiştir. Bağlantı kapasiteleri 0.1 Gbps, 0.2 Gbps, 0.5 Gbps ve 1 Gbps olarak ayarlanmıştır. Başlangıç popülasyonu büyüklüğü GA çözümü için 1024, düşük karmaşıklıklı yaklaşım için ise 128 olarak seçilmiştir. Çaprazlama olasılığı $P_c = 0.9$, mutasyon olasılığı ise $P_m = 0.1$ dir.

Ağ yükünün düşük olduğu gece saatleri göz önüne alındığı aşağıda tanımlanan senaryolar için performans değerlendirmesi yapılmıştır:

- Senaryo 1 (S1): Bütün akışlar aynı kaynak, varış noktası ve veri hacmi gereksinimine sahip olduğu düşünülerek yol ataması yapılmıştır. Kaynak noktaları 1. ve 9. anahtarlar, varış noktaları ise 10. ve 20. anahtarlar olarak seçilmiştir. Veri hacmi gereksinimleri ise 100 Mbps'dir.



Şekil 2: Ağ topolojisi

- Senaryo 2 (S2): Akışların hepsi aynı kaynak noktasına, fakat yarısı farklı, diğer yarısı farklı varış noktasına sahiptir. Veri hacmi gereksiniminleri ise 100 Mbps'dir. Kaynak noktaları 1. ve 9. anahtarlar, varış noktaları ise 10. ve 20. anahtarlar olarak seçilmiştir.
- Senaryo 3 (S3): Bütün akışlar aynı kaynak, varış noktasına sahiptir. Veri hacmi gereksinimleri akışların %50'si için 50 Mbps, %25'i için 75 Mbps ve kalan %25'i içinse 100 Mbps olarak kabul edilip yol ataması yapılmıştır. Kaynak noktaları 1. ve 9. anahtarlar, varış noktaları ise 10. ve 20. anahtarlar olarak seçilmiştir.
- Senaryo 4 (S4): Akışların hepsi aynı kaynak noktasına fakat yarısı farklı, diğer yarısı farklı varış noktasına sahiptir. Veri hacmi gereksinimleri akışların %50'si için 50 Mbps, %25'i için 75 Mbps ve kalan %25'i için ise 100 Mbps olarak kabul edilip yol ataması yapılmıştır. Kaynak noktaları 1. ve 9. anahtarlar, varış noktaları ise 10. ve 20. anahtarlar olarak seçilmiştir.

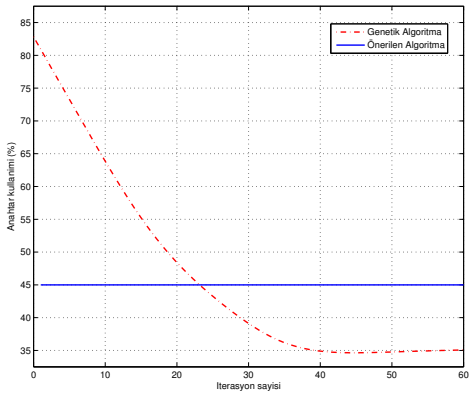
Bütün senaryolarda, kaynak ve varış noktalarına göre bütün kombinasyonlar düşünülüp ortalama sonuçlar elde edilmiştir. İlk olarak, GA çözümünün ve önerilen yaklaşımın değişik akış sayısına göre karmaşıklık analizi yapılmıştır. Örnek olarak kaynak noktası 1. anahtar, varış noktası 20. anahtar ve veri hacmi talebi 100Mbps için genetik algoritma ve önerilen algoritmanın çözüme yakınsamalarından bir örnek şekil 3'te gösterilmiştir.

Ortalama olarak önerilen algoritma 1 ila 2 iterasyonda çözüme ulaşırken, genetik algoritmanın bütün senaryolar için çözümüme ulaşması için gereken ortalama iterasyon sayısı Tablo I'de gösterilmiştir.

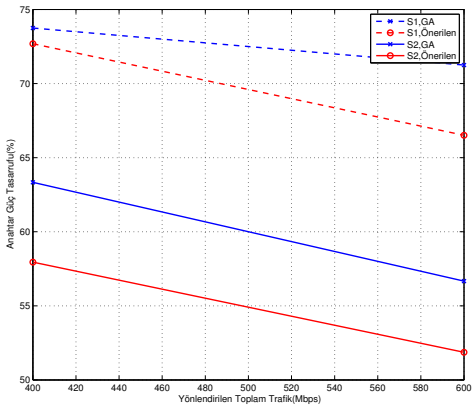
Tablo I: GA sonuca ulaşması için ortalama iterasyon sayısı

Akış Sayısı	S1	S2	S3	S4
4 akış	11.1200	12.2267	11.0950	11.1333
6 akış	13.8875	15.9867	17.7000	17.3600

Önerilen yaklaşımın karmaşıklığı, başlangıç popülasyon büyüklüğü ve iterasyon sayısı baz alındığında, 6 adet akış için GA sadece %5'idir. Buna karşılık düşük karmaşıklıklı önerilen yaklaşım Şekil 4'te gösterildiği üzere güç tüketimi açısından senaryoya bağlı olarak sadece %5 ila %10 arasında daha iyi performans vermektedir.



Şekil 3: Kaynak noktası 1. anahtar, varış noktası 20. anahtar ve veri hacmi talebi 100Mbps için sonuca yakınsama karşılaştırılması

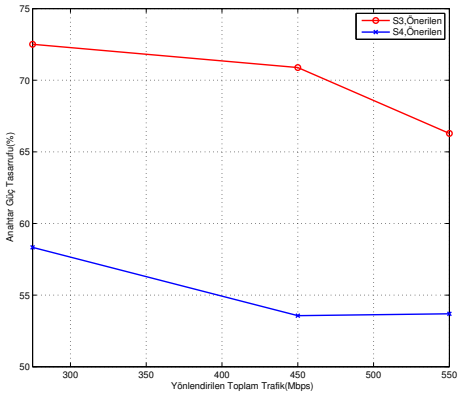


Şekil 4: Senaryo 1 ve Senaryo 2 için düşük karmaşıklıklı yaklaşımın GA çözümü ile performans karşılaştırması

Düşük karmaşıklıklı yaklaşımın, 4 farklı senaryo için çeşitli trafik yüklerinde güç tasarruf performansları Şekil 4 ve Şekil 5'de verilmiştir. Önerilen algoritmayla %50 ila %70 arasında güç tasarrufu sağlanmıştır.

V. SONUÇ

Yazılım tanımlı ağlar için güç verimli yol atama algoritması içeren bir mimari önerdik. Trafiğin düşük olduğu durumlarda ağdaki anahtarları kapatabilmek için en iyileme problemi tanımladık. Öncelikle, genetik algoritma kullanarak akışların veri hacmi gereksinimlerini karşılarken, ağdaki aktif anahtar sayısını minimuma indirerek elde ettik. Ayrıca akışlara sırasıyla yol atayan düşük karmaşıklıklı yol atama algoritması önerdik. Önerilen algoritmanın genetik algoritma çözüme çok yakın bir performans sergilediğini gösterdik. Performans karşılaştırmaları verilen ağ topolojisinde çeşitli veri hacmi gereksinimlerine göre aktif anahtar sayısına dayalı güç tüketimi hesaplanarak yapılmıştır. Bir sonraki çalışmada, yazılım tanımlı ağ kontrol biriminde akışlara trafik modeli, kullanıcıların yeri, karışım seviyesi gibi parametrelere göre



Şekil 5: Senaryo 3 ve Senaryo 4 için düşük karmaşıklıklı yaklaşımın performans sonuçları.

öncelik tanınması ve bu öncelik tanınmasının genel veri hacmi, güç tüketimi ve gecikmeye nasıl bir etkide bulunduğu incelenecektir.

VI. BİLGİLENDİRME

Bu çalısma CELTIC-Plus CP2012/2-5 SIGMONA projesi kapsamında Türk Telekom Argela Üniversite İşbirlikleri Programı tarafından desteklenmektedir.

KAYNAKÇA

[1] K.Hyojoon, N. Feamster,*Improving network management with software defined networking*, IEEE Commun. Mag., 51(2):114-119, February 2013.

[2] S. Sezer, S. Scott-Hayward, P. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, *Are we Ready for SDN? Implementation Challenges for Software-Defined Networks*, IEEE Communications Magazine, vol. 51, no. 7, pp. 36-43, 2013.

[3] T.Feng, J. Bi, K. Wang, *Joint Allocation and Scheduling of Network Resource for Multiple Control Applications in SDN*, 2014 IEEE Operations and Management Symposium (NOMS), pp.1-7, 2014.

[4] H.E. Egilmez, A.M. Tekalp,*Distributed QoS Architectures for Multimedia Streaming Over Software Defined Networks*, IEEE Transactions on Multimedia, Volume: 16 , Issue: 6, pp. 1597 - 1609, 2014.

[5] J. Zhang, K. Xi, M. Luo, H. J. Chao, *Load balancing for multiple traffic matrices using SDN hybrid routing,*, IEEE 15th International Conference on High Performance Switching and Routing, pp. 44-49, 2014.

[6] S. Oda, D. Nobayashi, Y. Fukuda and T. Ikenaga, *Flow-based Routing Schemes for Minimizing Network Energy Consumption using OpenFlow*, The Fourth International Conference on Smart Grids, Green Communications and IT Energy-aware Technologies, 2014.

[7] R.Wang, Z.Jiang, S. Gao, W. Yang, Y. Xia, M. Zhu, *Energy-Aware Routing Algorithms in Software-Defined Networks*, IEEE 15th International Symposium on World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014.

[8] Markiewicz, A, Phuong Nga Tran, Timm-Giel, A.,Energy consumption optimization for software defined networks considering dynamic traffic, IEEE 3rd International Conference on Cloud Networking CloudNet),Luxembourg, October 2014

[9] D. E. Goldberg and J. H. Holland, *Genetic algorithms and machine learning*, Machine learning, vol. 3, no. 2, pp. 95-99, 1988.