Mühendislik Bilimleri ve Tasarım Dergisi
8(4), 999 – 1009, 2020
e-ISSN: 1308-6693

Journal of Engineering Sciences and Design
DOI: 10.21923/jesd.676110

Araştırma Makalesi Research Article

YAPAY ZEKÂ TEKNİKLERİYLE YAZILIM TANIMLI AĞ UYGULAMASI

Şükran DEĞİRMENCİ, Derya YILTAŞ KAPLAN*

İstanbul Üniversitesi-Cerrahpaşa, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, İstanbul, Türkiye

Anahtar Kelimeler

Doğrusal Arama,
Tabu Arama,
Yapay Sinir Ağları,
Yapay Zekâ,
Yazılım Tanımlı Ağ.

Öz

Bu çalışma yazılım tanımlı ağ denetleyicisi kullanarak ağ trafiğini takip etme amacıyla yapılmıştır. Bu doğrultuda denetleyici olarak Floodlight ve çalışma platformu olarak Eclipse kullanılmıştır. Doğruluğu artırmak adına beş ayrı topoloji üzerinde gerçekleştirilen takipler sonrasında paket istatistik bilgileri elde edilmiştir. İstatistik verileri MATLAB üzerinde işlenerek istenilen veri setleri oluşturulmuştur. Bu veri setleri MATLAB'de yer alan nntool vasıtasıyla yapay sinir ağlarında kullanılarak belirli zamanlardaki paket akışı tahmin edilmiştir. Yapay sinir ağları beş ayrı topoloji üzerinde denenmiş ve doğrulukları MAPE ve R^2 denklemleri ile test edilmiştir. Yapılan testler sonucunda doğruluğu en yüksek çıkan topoloji üzerinde optimizasyon işlemleri uygulanmıştır. Bu amaç doğrultusunda doğrusal arama, tabu arama, değiştirilmiş tabu arama ve tavlama benzetimi ile tabu arama algoritmasının karışımı olmak üzere toplamda dört ayrı optimizasyon tekniği ile optimizasyon testleri gerçekleştirilmiştir. Sonuç olarak çalışmada önerilen karışım algoritması ile gerçekleştirilen optimizasyon sonucunda en yoğun olan güzergâh en kısa sürede tespit edilmiştir.

SOFTWARE DEFINED NETWORK APPLICATION WITH ARTIFICIAL INTELLIGENCE TECHNIQUES

Keywords

Linear Search,
Tabu Search,
Artificial Neural Networks,
Artificial Intelligence,
Software Defined Network.

Abstract

This study has been done to monitor the network traffic using a software defined network controller. For this purpose, Floodlight was used as the controller and Eclipse was used as the development platform. In order to increase the accuracy, the statistical information on packets was obtained after observing five separate topologies. The statistical data were processed on MATLAB and the desired data sets were created. These data sets are used in artificial neural networks via nntool on MATLAB to predict the packet flow at a particular time. Artificial neural networks were tested on five separate topologies and their accuracies were tested with MAPE and R^2 functions. The optimization operations were applied to the topology having the highest accuracy derived from the performed tests. For this purpose, the optimization tests were verified by four optimization techniques which were linear search, tabu search, modified tabu search and combination algorithm (simulated annealing-tabu search). As a result, with the optimization performed with the proposed combination algorithm, the densest route was determined at the smallest time duration.

Alıntı / Cite

Değirmenci, Ş., Yıltaş-Kaplan, D., (2020). Yapay Zekâ Teknikleriyle Yazılım Tanımlı Ağ Uygulaması, Mühendislik Bilimleri ve Tasarım Dergisi, 8(4), 999-1009.

Yazar Kimliği / Author ID (ORCID Number)

Ş. Değirmenci, 0000-0002-8571-0401
D. Yıltaş Kaplan, 0000-0001-8370-8941

Makale Süreci / Article Process	
Başvuru Tarihi / Submission Date	16.01.2020
Revizyon Tarihi / Revision Date	14.09.2020
Kabul Tarihi / Accepted Date	21.09.2020
Yayım Tarihi / Published Date	25.12.2020

* İlgili yazar / Corresponding author: dyiltas@istanbul.edu.tr, +90-212-473-7070

1. Giriş (Introduction)

Gelişen teknoloji ile beraber veri merkezleri gittikçe büyümektedir. Büyüyen veri merkezlerinde ise büyük hacimli, karışık ve düzensiz bilgiler yer almaya başlar. Büyük veri içerisindeki bu bilgilerin anlamlı ve değerli olabilmeleri için işlenmeleri gerekir. Büyük veri geleneksel yöntemlerle işlenemez, yönetilemez ve depolanamaz. Yani geleneksel ağ yönetimi yaklaşımı bu aşamada yetersiz kalır. Daha iyi bir ağ yaklaşımıyla, yeni metotlarla ve daha geniş bir bant aralığıyla bu veriler işlenebilir hale gelmektedir. Yazılım Tanımlı Ağ (YTA) bu ihtiyaçları karşılayan bir yöntem olarak karşımıza çıkar. YTA yönetim kolaylığını, donanım bağımsızlığını, dinamik, esnek ve ölçeklenebilir ağ mimarisini temin eder. Dolayısıyla bu geniş ve karmaşık ağ yönetimine etkili bir çözüm sunar.

Ağların kurulumu ve yönetimi için birden fazla ağ elemanının konfigürasyonu konusunda yetenekli uzmanlar gerekir. Ağ elemanları (anahtarlar, yönlendiriciler, vb.) arasındaki iletişimin karmaşık olduğu durumlarda sistem tabanlı bir yaklaşıma ihtiyaç duyulmaktadır. Günümüzün ağ donanımlarının çoğundaki mevcut programlama arabirimleri ile bunu gerçekleştirmek zordur. Bunu sağlamak için yeni bir ağ modeline ihtiyaç duyulmuştur ve bu esnada YTA kavramı ortaya çıkmıştır (Sezer vd., 2013). YTA'nın geliştirilmesine, standardizasyonuna ve ticarileştirilmesine yönelmiş olan Açık Ağ Oluşturma Vakfı (Open Networking Foundation) kâr amacı gütmeyen bir kuruluştur. Vakıf, YTA'yı en iyi açıklayan tanımı şu şekilde yapmaktadır: YTA, ağ denetiminin iletimden ayrıldığı ve doğrudan programlanabilen yeni bir ağ mimarisidir (Xia vd., 2015).

YTA mimarisi uygulama, kontrol ve veri katmanı olmak üzere üç tane katman ile uygulama-kontrol ve kontrol-veri katmanları arasında olmak üzere iki tane arayüzden oluşmaktadır. Kontrol katmanı temel olarak paketlerin gönderilme işlemlerinin gerçekleştiği yerdir. Veri katmanında paketlerin iletimi esnasında oluşan trafik akışı düzenlenmektedir.

Geleneksel ağ trafiğinde paketin gideceği yeri yönlendirici ve anahtarlar belirlemektedir. Bunlar da kontrol ve veri katmanlarında birbirlerine entegre edilmiş şekilde aynı donanım üzerinde bulunurlar. YTA temel olarak bu iki katmanı birbirinden ayırmaya odaklanmıştır. YTA'da kontrol düzlemi yüksek performanslı bir sunucuya taşınır ve ağ yönetimi merkezi bir denetleyici yazılımı ile gerçekleştirilir. Veri katmanı yönlendiricilerin ve anahtarların yalnızca akış yönlendirme işleminden sorumlu olmasını sağlamaktadır. Kontrol katmanı ise ağ işletim sistemi olarak bilinmektedir. Bu katmanda ağ uygulamaları ve veri katmanı arasındaki iletişim gerçekleşmektedir. Kontrol katmanı ve veri katmanı arasındaki iletişim ise açık kaynak ağ protokolü olan OpenFlow ile sağlanmaktadır (Niyaz vd., 2015). YTA mimarisi ağın doğrudan programlanabilmesini sağlamanın yanında ağ servisleri ve uygulamaları için de gerekli olan alt yapıyı oluşturmaktadır.

İletişim ağının temel amacı, bilgi paketlerini bir noktadan diğerine aktarmaktır. Ağ içinde birden çok düğüme iletim gerçekleştiği için bu da yoğun bir trafik akışına sebep olmaktadır. Bu esnada YTA kullanılarak denetleyici sayesinde, etkin ve verimli trafik akışı sağlanabilmektedir. Böylece yoğunluğa ve çeşitliliğe sebep olan trafiğin oluşturduğu karmaşanın önüne geçilerek, daha basit ve yönetimi daha kolay olan bir anlayış sunulmaktadır.

Bu çalışmada ağlardaki trafik akışının YTA denetleyicisine bağlı olarak gerçekleştirilmesi ve burada elde edilen verilerin yapay zekâ optimizasyon teknikleri kullanılarak optimize edilmesi amaçlanmıştır. Elde edilen veriler ışığında yapay sinir ağları (YSA'lar) kullanılarak tahmin üzerine bir uygulama geliştirilmesi amaçlanmıştır. Çalışmanın içeriği şöyle devam etmektedir: 2. bölümde kullanılan araç ve yöntemlerle ilgili genel bilgiler, 3. bölümde önerilen metodun uygulama adımları, 4. bölümde uygulamadan elde edilen sonuçlar, son olarak 5. bölümde konu ile ilgili sonuç ve tartışma kısmı yer almaktadır.

2. Genel Bilgiler (General Information)

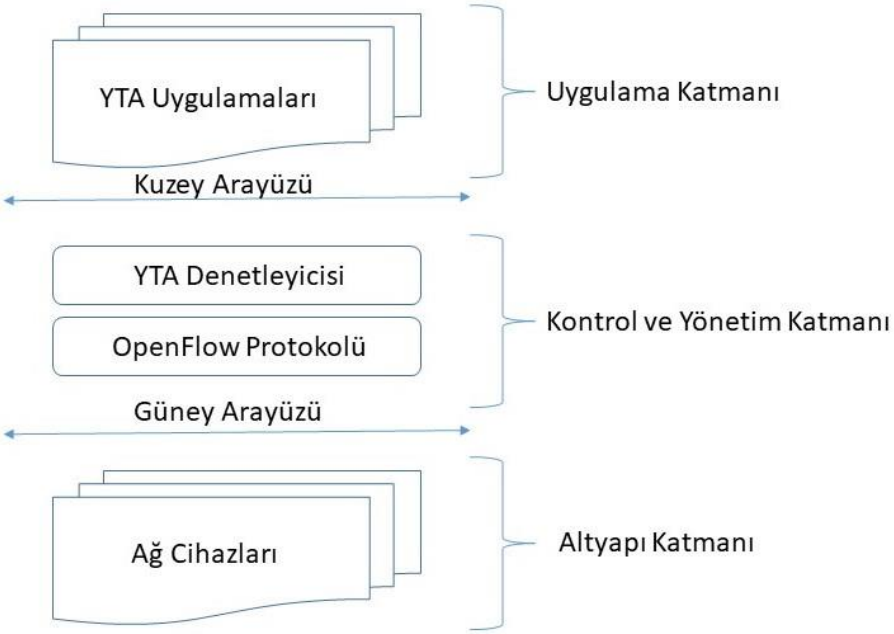
Bu bölümün alt başlıklarında sırasıyla uygulama kısmı için gerekli verilerin elde edildiği denetleyici kavramının ne olduğu ve ağ ortamındaki işleyişi, toplanan ağ verilerinin kullanılması için tercih edilen YSA yöntemi, ağ üzerinde akışın en yoğun olduğu bölgelerin tespiti için faydalanılan yapay zekâ optimizasyon teknikleri arasında yer alan Tabu Arama Algoritması ve Tavlama Benzetimi Algoritması ile ilgili temel açıklamalar yapılmaktadır.

2.1. Denetleyicinin Çalışması (Controller Operation)

YTA mimarisinde kontrol ve yönetim işlemleri merkezi bir denetleyici tarafından gerçekleştirilmektedir. Bu denetleyici ağ topolojisine ve adres bilgisinin tamamına hâkimdir. Bir anahtar, akış tablosunda bulunmayan bir paket için denetleyiciye paket gönderim isteği yolladığı zaman, denetleyici bu istekleri alır. Sonrasında denetleyici eylemleri alır ve trafiği dinleyerek talep üzerine paket iletim rotalarını tanımlar. OpenFlow denetleyicileri ağ operatörleri tarafından bir kontrol arabirimi ile programlanmaktadır.

Şekil 1'de gösterildiği gibi büyük ağlar daha fazla YTA denetleyicisi, yani YTA uygulama katmanı içerecek şekilde oluşturulmaktadır. Bu uygulama katmanı, ağ sanallaştırması, trafik mühendisliği, yönlendirme, izleme ve hizmet kalitesi servislerinden sorumludur.

Uygulama katmanı ve YTA denetleyicileri arasındaki iletişim, Kuzey Arayüzü olarak adlandırılan bir dizi uygulama programlama arayüzü (API) tarafından gerçekleştirilir. Benzer şekilde YTA denetleyicisi ve OpenFlow ağ aygıtları arasındaki arabirime de Güney Arayüzü denmektedir (Kutay ve Ercan, 2016). YTA mimarisi, karmaşık ağlar üzerinde iletim, denetim ve yönetim kolaylığı sağlamaktadır. Ayrıca ağ katmanları arasındaki etkileşimler sayesinde de trafik akışı açısından güçlü bir kontrol platformu olarak günümüz teknolojisinde önemli bir yer bulmaktadır.



Şekil 1. YTA Katmanları (SDN Layers)

2.2. Yapay Sinir Ağları (Artificial Neural Networks)

Yapay zekâ farklı türde problemlere getirdiği çözümlere göre çeşitli alt başlıklara ayrılmıştır. Bunlardan en yaygın olan yapay zekâ türleri ise bilgi tabanlı uzman sistemler, bulanık mantık yaklaşımı ve YSA'lardır. Bu çalışmada kullanımı tercih edilen yöntem YSA'lardır.

İnsan beyninin nörolojik yapısı incelendiğinde dijital bilgisayarlardan tamamen farklı bir çalışma mekanizmasına sahip olduğunun anlaşılması üzerine YSA kavramı ortaya çıkmıştır. İnsan beyninin bilgi işleme sistemi oldukça karmaşık bir yapıdadır. Beyinde bulunan ve nöron olarak adlandırılan sinir hücresi günümüzde insanın en hızlı kabul ettiği bilgisayardan bile hızlı olacak şekilde belirli hesaplamalar gerçekleştirmektedir. Sinir ağı, deneyimsel bilginin depolanması ve kullanımı için uygun hale getirildiği doğal bir eğilime sahip olan, basit işlem birimlerinden oluşan, büyük ölçüde paralel dağıtılmış bir işlemcidir (Haykin, 2009). YSA'lar ise insan beynindeki nöronlar gibi yapay nöronların aynı şekilde bir araya gelerek problemle ilgili bilgi topladıkları ve problemi çözdükleri sistemlerdir. Kısaca YSA, yapay nöronların farklı geometrik yollar kullanarak birbirlerine bağlanmasıyla oluşan ağ sistem yapısıdır (Staub vd., 2015). YSA insan beynine iki temel açıdan benzemektedir. Birincisi, bilginin çevreden bir öğrenme süreci ile ağ tarafından edinilmesidir. İkincisi ise edinilen bilgiyi saklamak için sinaptik ağırlıklar olarak da bilinen dâhili nöron bağlantı kuvvetlerinin kullanılmasıdır. YSA yapısı, doğrusal olmayan ilişkileri öğrenmeye olanak tanırken aynı zamanda değişen ortamlara da uyum sağlamaktadır. YSA'ların en çok tercih edilen özellikleri arasında eksik parametre değerlerini idare edebilme kabiliyeti bulunmaktadır. Bu özelliği sayesinde sınırlı sayıda eğitim setine dayanan bir YSA iyi bir sınıflandırıcı olabilmektedir (Mourrain vd., 2006). YSA'lar bir problemle karşılaştıklarında öğrendiklerini kullanarak karar verebilme yeteneğine sahiptirler. YSA'larda bilgi, ağ tarafından edinilir ve elde edilen bu bilgi nöronlar arasındaki bağlantılar kullanılarak saklanır. YSA'larda ağ, bilgiyi saklayan ve bu bilgiyi işlevsel hale getiren bir işlemci olarak tanımlanabilir (Taşhan, 2017).

Farklı bir problem alanıyla ilgili YSA'ların yardımıyla tasarlanmış olan bir çalışma içerisinde YSA örneğinin giriş katmanı, gizli katman ve çıkış katmanı şeklindeki genel gösterimi bulunmaktadır (Abu Salam ve Keskin, 2018). Burada ayrıca geri yayılım gibi farklı YSA modellerinin olduğu vurgulanmaktadır.

2.3. Yapay Zekâ Optimizasyon Teknikleri (Artificial Intelligence Optimization Techniques)

Optimizasyon problemlerinin çözümü için meta-sezgisel olarak da bilinen yapay zekâ optimizasyon teknikleri kullanılarak etkili sonuçlar elde edilmektedir. Sezgisel bir algoritma, bir sezgisel işlevi dönüşümlü çağırarak kârı maksimize eden belirli bir çözüm örneğini bulmaya çalışır. Kârı maksimize eden örnek çözüm, optimizasyon problemi için optimal çözüm olacaktır. Sezgisel teknikler ya üstün olan ya da daha üstün olan bir çözüme ulaşmak için farklı çözüm modifikasyonlarını gerçekleştiren yöntemlerdir (Oommen ve Rueda, 2005). Yapay zekâ optimizasyon teknikleri, optimizasyon problemlerini çözmek için ümit verici bir tekniktir. Çünkü beynin operasyonlarını taklit edebilmekte ve hesaplama süresini kaydetmek için paralel işlemeyi kullanmaktadır (Shih vd., 2004). Bilgisayar algoritmalarından oluşan bu teknikler en iyi çözüme ulaşmak için var olan çeşitli hamlelerden en iyi sonuç verecek olanı bulmayı hedeflemektedir. Bazı problemlerin pek çoğunun nihai bir çözümü olmadığından ve bu problemlerin çözümü için gerekli olan sürenin gayet büyük olmasından dolayı, araştırmacılar nihai çözüme yakın bir sonuç veren en hızlı algoritmayı bulmaya çalışmışlardır. Sonuç olarak da yapay zekâ optimizasyon teknikleri ortaya çıkmıştır. Bu teknikler kesin çözüm metotları için bir alt veya üst sınır oluşturmaları nedeni ile de önemlidir. Ayrıca kesin bir çözüme ulaşmamakla beraber muhtemel en iyi çözümü bulmayı hedeflemektedirler.

2.3.1. Tabu Arama Algoritması (Tabu Search Algorithm)

1986 yılında Fred Glover tarafından ortaya atılan tabu arama algoritması, yerel arama metotlarının eksikliklerini gidermek üzere oluşturulmuştur. Bazı karmaşık problemlere yerel arama metotları ile en iyi çözümler getirilememekteydi. Tabu arama bu tip problemlere en etkili sezgisel çözümü getirmeyi hedeflemektedir. Tabu arama, ilk olası bir noktadan başlayıp bir sonlu dizi oluşturarak optimal bir çözüme ulaşmaya çalışan akıllı bir arama prosedürüdür. Başlangıçta seçilen olası çözüm noktasının komşu alt kümeleri oluşturulur ve bir sonraki nokta en iyi olan çözüm olarak belirlenir. Tabu arama, yükselme hareketlerine izin verdiği için önceki bazı noktalara geri dönüşü önlemek amacıyla tabu hareketleri tabu listesi adı altında listelenir. Çıkış hareketinden dolayı prosedür, yerel optimumdaki tuzakları önler (Kovačević-Vujčić vd., 1999). Kısaca özetlemek gerekirse tabu aramanın çalışma prensibi, yerel optimumla karşılaşıldığında en iyi olmayan bu çözüme izin vererek yerel arama üzerinden takibe devam etmektir (Gendreau ve Potvin, 2010). Tabu arama algoritması arama yaparken birtakım kısıtlamalar kullanır. Bu sebepten dolayı algoritmaya "tabu" yani "yasaklı arama" denmiştir. Kısıtlamalar ise hafıza yapılarından faydalanılarak oluşturulur (Aladağ, 2009). Bu kısıtlamalar sayesinde normalde kabul edilebilen zor durumlar atlanarak en iyi sonuca ulaşılması sağlanır. Tabu aramada problem çözme işleminin akıllı olarak nitelendirilebilmesi için uyarlanabilir bir hafıza ve duyarlı bir araştırmanın olması gerekir. Bu özellikler ile probleme özel çözümler üretebilme yetisi kazanan tabu arama algoritması, meta-sezgisel bir arama algoritması olarak kabul edilmektedir (Michalska vd., 2016).

2.3.2. Tavlama Benzetimi Algoritması (Simulated Annealing Algorithm)

Olasılık tabanlı sezgisel bir algoritma olan tavlama benzetimi algoritması Kirkpatrick, Gelatt ve Vecchi tarafından oluşturulmuştur. Tavlama benzetimi algoritması mevcut çözüm etrafında rastgele bir varyasyon kullanarak optimizasyon probleminin iyi olan bir çözümünü bulmak için oluşturulan bir tekniktir (Zhao vd., 2013). Katı cisimlerin fiziksel tavlanma işlemi esas alınarak oluşturulan bu algoritma, karmaşık optimizasyon problemlerinin çözümünde kullanılmaktadır. Katı cismin erime noktasına kadar ısıtıldıktan sonra cisim kristalize olana kadar yavaş yavaş cismin soğutulması işlemine tavlama denmektedir. Yüksek sıcaklıkta enerji seviyesi de yüksek olan cisimlerin atomları, düzgün yapılı bir kristal haline geçtiği esnada sistemin enerjisi minimum düzeye inmektedir. Sistemin enerjisini düşürmek için de sıcaklık azaltılmaktadır. Bu soğutma işlemi esnasında çok hızlı davranılırsa, cismin kristalize yapısında bozulma gerçekleşmekte ve cismin yapısında düzensizlik oluşmaktadır (Kalınlı, 2003).

Çok değişkenli fonksiyonlarda fonksiyonun en büyük ya da en küçük değerlerinin bulunması amacıyla tasarlanan tavlama benzetimi algoritması en kısa sürede en iyi çözümü bulmaktadır. Bu sebeple matematiksel bir modelle ifade edilemeyen problemlerin optimizasyonunda tercih edilmektedir. Tavlama benzetimi algoritmasında amaç fonksiyon değeri azalma eğilimindedir (Çakır, 2006). Fakat yerel minimuma takılı kalmamak için bazı durumlarda bu fonksiyon da yüksek değerlerde kabul görebilmektedir. Bu şekilde yerel minimuma takılmayıp daha iyi global bir çözüm aranabilmektedir (Kılıçaslan, 2019).

3. Önerilen Metodun Uygulaması (Application of the Proposed Method)

Bu çalışma kapsamında YTA denetleyicisine bağlı olarak akış trafiğinin gözetlenmesi üzerine uygulama geliştirilirken Floodlight VM, Eclipse, MATLAB ve nntool araçlarından yararlanılmıştır. YTA denetleyici yazılımları arasında sürekli gelişmekte olan bir proje olarak dünya çapında yaygınlaşan Floodlight, açık kaynak kodludur ve Java programlama dilini destekleyen bir uygulamadır (Floodlight, 2020). Floodlight projesinin resmi internet

sitesinde yazılımcılar için gerekli tüm adımlar ve kurulumlar ayrıntılı şekilde sunulmaktadır. Önerilen metodun program aşamaları için yine açık kaynak kodlu ve özellikle Java tabanlı yazılımların tasarlandığı Eclipse ortamı kullanılmaktadır. Eclipse, kurulum sırasında Floodlight VM ile entegreli olarak gelmektedir. Aslında bu çalışma sırasında kurulan Floodlight VM içerisinde Floodlight v1.0, Eclipse, Mininet v2.2.0, Open vSwitch v3.2.1, Wireshark w/openflow yazılımları bulunmaktadır. Uygulamada oluşturulan topolojiler Mininet üzerinde Python programlama dili ile oluşturulmaktadır. Floodlight ile karşılıklı işlem yapması için Java platformu olarak tercih edilebilecek farklı bir alternatif örnek IntelliJ IDEA'dır. Bütün bu uygulama ortamlarının yerine farklı denetleyici yazılımları için farklı araçlar kullanılabilmektedir. Örneğin POX denetleyicisi Python programlama dili ile tasarlanmıştır (Yazar, 2013). Aynı şekilde Ryu denetleyicisi de Python temel bilgisini gerektirmektedir. Önerilen çalışmada en uygun denetleyici ve programlama ortamlarına karar verildikten sonra verilerin yapay zekâ tekniklerine dayalı işlemler için kullanımı daha etkin ve kolay olan MATLAB ve nntool tercih edilmiştir.

Uygulamada trafiği gözetlemek için gerekli olan veriler anahtar, port numarası, alınan ve iletilen paketlerin miktarı, bant genişliği, düşmeler, çarpışmalar ve bunların gerçekleştiği saniye (sn) cinsinden süre bilgileridir. Bu verilere Floodlight VM içerisinde bulunan Floodlight projesi sayesinde erişilmektedir. Doğruluğu artırmak adına çeşitli topolojiler üzerinden istatistik verileri elde edilmiştir. Bunun için Mininet ortamında farklı sayıda anahtar ve makinelerden (hostlardan) oluşan beş ayrı topoloji oluşturulmuştur. Oluşturulan topolojilerin her biri 27 sn çalıştırılarak beş ayrı veri seti elde edilmiştir. Sürenin 27 sn seçilmesi ile binlerce satır veri elde edilmiştir ve bu da verilerin işlenmesi sürecinde yeterli görülmüştür. Sürenin arttırılması ile veri miktarında aşırı bir artış gerçekleştiği gözlemlendiğinden ve bu da verilerin işlenmesi esnasında problem teşkil edeceğinden 27 sn'lik süre tercih edilmiştir. MATLAB'de matris yapıları üzerine aktarılan verilerin anahtar, port, alınan ve iletilen paket miktarı ve süre sütunları seçilip diğer sütunlar göz ardı edilmiştir. Burada belirtilmesi gereken ayrı bir nokta, YTA'larda veri iletiminin akışlar şeklinde olmasıdır. Bir akış birçok paket içeriğine sahip olduğu için klasik ağ yapılarıyla bağlantılı çalışmalarda kavramsal karışıklıklar ortaya çıkabilmektedir. Bu nedenle bu çalışma kapsamında Floodlight üzerinden elde edilen veri miktarlarını ifade etmek için klasik bilgisayar ağlarındaki kavram olan "paket miktarı" terimi tercih edilmiştir. Böylece asıl odak noktası, elde edilen veri setleri ve bunların üzerinde yapılacak uygulama işlemleri olmaktadır. Çalışmanın uygulama kısmı için farklılığı oluşturacak etmenler kullanılan algoritmalar ve sonuçların analizindeki yaklaşımdır.

Yapılan çalışmanın uygulama kısmı yapay zekâ üzerine kurulmuştur. Burada amaç belirli bir anda belirli anahtar ve porttan ne kadar paket alınıp iletildiğini tahmin etmeye çalışmaktır. Bu tahmin kısmı ise yapay zekâ yöntemlerinden olan YSA ile gerçekleştirilmiştir. MATLAB aracı olan nntool ile verilerin YSA üzerinde işlenmesi sağlanmıştır. MATLAB matris işlemleri yardımıyla veri setlerinin %80'lik kısmı eğitim, %20'lik kısmı ise test için ayrılmıştır. Veri setlerinin anahtar, port ve süre sütunları girdi verisi olarak, alınan paket ve iletilen paket sütunları ise çıktı verisi olarak ayarlanmıştır. nntool üzerinde gerekli kısımlar seçilerek ağlar oluşturulmuştur. İlk oluşturulan ağ üç girdisi iki çıktısı olan bir ağdır. Fakat elde edilen tahmin başarısı düşük olduğundan ağlar üç girişli bir çıkışlı olmak üzere tasarlanmıştır. Böylece her bir topoloji için alınan paket ve iletilen paketlere göre ayrı ağlar oluşturulmuştur. Çok karmaşık veri setlerine sahip olunmadığından on nöronluk iki katmanlı YSA yeterli olmuştur. Beş farklı topoloji için oluşturulan YSA'ların belirli anlardaki tahminlerinin doğruluğu kıyaslanmıştır. Doğruluklar ise MAPE (Mean Absolute Percentage Error) ve R^2 (R-squared) fonksiyonları kullanılarak hesaplanmıştır.

Uygulamanın son kısmı ise optimizasyondan oluşmaktadır. Optimize edilecek problem, yani optimum sonuç olarak elde edilmek istenen değer, topolojide hangi güzergâhta en fazla yoğunluğun olduğudur. Bunun için dört farklı yöntemle optimizasyon gerçekleştirilmiş ve bunların sonuçları karşılaştırılmıştır. Bunlardan ilki geleneksel arama metodu olan doğrusal arama, ikincisi tabu arama, üçüncüsü değiştirilmiş tabu arama, dördüncüsü ise tavlama benzetimi ile tabu arama algoritmalarının karışımıdır (bundan sonrasında karışım algoritması olarak isimlendirilecektir). Son iki yöntem ilk kez bu çalışma çerçevesinde ortaya çıkarılan metotlardır. Bu dört farklı optimizasyon yöntemi sonucunda paket trafiğinin en yoğun olduğu güzergâh tespit edilmiştir. Bu aşamada kullanılan algoritmaların tercihi için önemli olan kriterler fonksiyonel olarak ağ problemine uyarlanabilmeleri, yani ağ üzerindeki bölgelerde yoğunluğun bulunmasını birbirleriyle uyumlu şekilde kıyaslayabilmeleri ve hızlı olmalarıdır. Bu çalışmada optimizasyon adımıyla oluşturulan işlem, problemin doğru çözümüne gitmek için en kısa sürenin ön plana alınmasıdır. Bu amaçla en uygun metotların bulunmasına çalışılmıştır. Önceki farklı bir ağ çalışmasında (Yıltaş, 2007) evrimsel algoritmalar arasında yer alan genetik algoritma kullanılmıştır. Genetik algoritma özellikle yönlendirme (rotalama), atama veya yerleştirme tabanlı optimizasyon problemleri için elverişli bir yöntemdir ve bu konuda Ünsal ve Yiğit (2018) ile Yiğit ve Aydemir (2018)'in çözümleri gibi literatürde başka örnekler de vardır. Ancak genetik algoritmada yer alan yeni jenerasyon oluşturma, çaprazlama, mutasyon gibi aşamalar, bu çalışmanın temel amacı olan yoğun ağ bölgelerinin kısa sürede tespiti konusuyla bağdaşmamaktadır. Bu çalışma içerisinde oluşturulan programlar aracılığıyla amaca uygun ve hızlı işlemler yapabildiği görülen tabu arama ve tavlama benzetimi algoritmaları ele alınarak yeni ek iki model (değiştirilmiş tabu algoritması ve karışım algoritması) önerilmiş, bunların doğrusal arama algoritması ile karşılaştırılmaları

planlanmıştır.

Değiştirilmiş tabu arama algoritması, tabu arama algoritmasından tabu listesi çıkartılarak elde edilmiştir. Tabu arama algoritması incelendiğinde tabu listesinin bu problem için gereksiz olduğu ve optimizasyonu yavaşlattığı gözlemlenmiştir. Liste algoritmadan çıkarıldığında performansta artış gözlemlenmiştir. Buradan da yapay zekâ optimizasyon tekniklerinin her ne kadar elverişli olsalar da probleme özgü olmalarından dolayı her zaman başarı gösteremedikleri gözlemlenmiştir. Başka bir problemde, örneğin gezgin satıcı probleminde, gayet elverişli olan bu algoritma, mevcut çalışmadaki problemde performans düşüklüğüne sebebiyet vermektedir.

Son yöntem olan karışım algoritmasının ilk kısmı tabu arama algoritmasından gelmektedir. Tavlama benzetimi algoritmasında rastgele seçilen başlangıç çözümünün yerine, tabu arama algoritmasındaki ilk değeri başlangıç çözümü olarak atama durumu tercih edilmiştir. Daha sonraki kısımlarda tabu aramaya yer verilmemiştir. Geri kalan kısımda komşu çözümler oluşturulmuş olup amaç fonksiyonundaki yani toplam paket değerindeki değişim gözlemlenmiştir. Ek olarak bu kısımda da bir değişikliğe gidilmiştir. Normalde amaç fonksiyonunda azalma söz konusu ise çözüm mevcut çözüm olurken bu algoritmada tam tersi göz önünde bulundurulmuştur. Yani amaç fonksiyonunda artış olduğu takdirde çözüm mevcut çözüm olmaktadır. Çünkü problemin temelinde ağdaki bölge yoğunlukları, yani paket akışının fazlalığı gözlemlenmektedir.

Uygulama kısmında yer alan işlem elemanlarının birbirleriyle bağlantısı Şekil 2'de özet olarak görülmektedir.

Şekil 2. Uygulama İşlemleri (Application Operations)

4. Araştırma Bulguları (Research Findings)

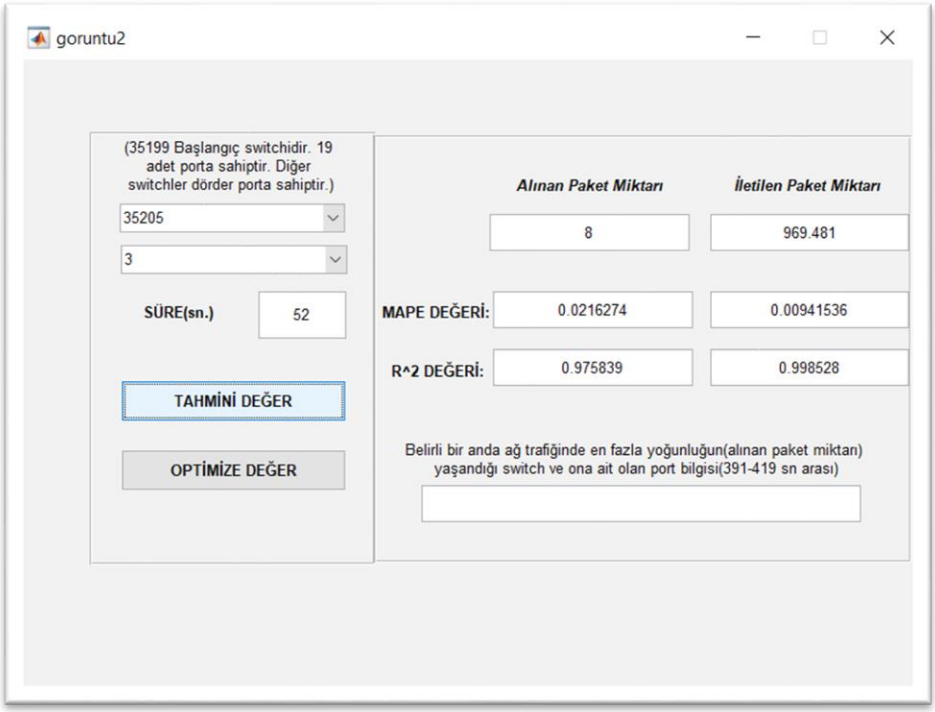
Bu bölümdeki sonuçlar Intel Core i7-7700HQ CPU 2.80GHz, 16 GB RAM, Windows 10/ 64 bit işletim sistemine sahip bir dizüstü bilgisayar üzerinde hazırlanan programlar aracılığıyla elde edilmiştir. Yapılan çalışma birbirine bağlı üç kısımdan oluşmaktadır. İlk kısım YTA sayesinde ağ topolojileri hakkındaki istatistik verilerini elde etmek, ikinci kısım elde edilen bu veriler ışığında YSA kullanılarak belirli bir andaki paket trafiğini tahmin etmek ve son olarak üçüncü kısım ise YSA sonuçlarına göre doğruluğu en yüksek olan topoloji üzerinde trafik yoğunluğunu optimize etmek olarak planlanmıştır.

İlk kısım olan YTA tarafında Python yardımıyla elde edilen beş farklı topoloji uygulama üzerinde çalıştırılarak denetleyicinin işlerliği gözlemlenmiştir. Tablo 1'de bu topolojilere ait bilgiler yer almaktadır. Denetleyicinin farklı türde topolojiler üzerinde etkin bir biçimde çalıştığı tespit edilmiştir. İlk kısım sonucunda beş ayrı veri seti elde edilmiştir.

Tablo 1. Python yardımıyla oluşturulan beş ayrı topolojiye ait bilgiler (Information about five separate topologies created by Python)

Topoloji Türü	Anahtar	Makine
Doğrusal	45	45
Ağaç	9	64
Dairesel	10	36
Dairesel	20	57
Dairesel	30	87

İkinci kısım olan yapay zekâ tarafında YSA kullanılarak bir tahmin uygulaması gerçekleştirilmiştir. Buradaki temel hedef, belirli bir zamanda belirli anahtar ve port üzerinden alınıp iletilen paket miktarını tahmin etmektir. Şekil 3'te uygulamanın tahmin kısmının çalışmasından bir kesit yer almaktadır.



Şekil 3. Alınan ve iletilen paket miktarlarına göre yapılan tahmin ve doğruluk oranları (Estimation and accuracy rates based on the received and transmitted packet amounts)

Yapılan tahminlerin doğruluk oranları MAPE ve R^2 fonksiyonlarına göre ölçülmüştür. MAPE fonksiyonu ortalama mutlak yüzde hatası anlamına gelmektedir ve Eşitlik (1)'de hesaplanmaktadır.

$$MAPE = \frac{100x\sum|\frac{(p_i-a_i)}{a_i}|}{\mathrm{n}} \qquad (1)$$

Burada *a* değeri belirli bir gözlemdeki gerçek değeri, *p* bu gözlem için modelin tahmin ettiği değeri ve *n* ise gözlemlenen durumların sayısını ifade etmektedir.

MAPE fonksiyonu hatanın artış veya azalış yönünde olmasıyla ilgilenmeyip sadece sapma miktarını hesaplamaktadır. Bu sebeple hatanın mutlak değeri alınmaktadır. MAPE fonksiyonuna göre elde edilen değerin küçük olması gerçek değerden sapma miktarının az olacağı anlamına gelmektedir.

R^2 ise değişkenler arasında doğrusal bir ilişki olup olmadığını ölçmemize olanak tanımaktadır. Fonksiyonun değeri 0 ile 1 arasındadır. Eğer sonuç sıfıra yakın çıkarsa sonuç doğruluktan uzaklaşmış, bire yakın çıkarsa tahminin doğru olma olasılığı yüksek demektir. YSA'larda kullanılan R^2 Eşitlik (2) ve Eşitlik (3)'ten yola çıkarak Eşitlik (4)'te olduğu gibi hesaplanmaktadır.

$$SSt = \sum_{i=1}^{n}(y_i - ort(y_i))^2 \qquad (2)$$

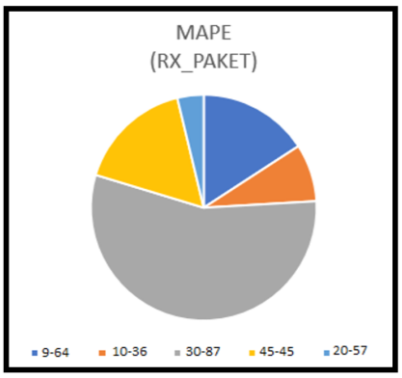
$$SSe = \sum_{i=1}^{n}(N_i - y_i)^2 \qquad (3)$$

$$R^2 = 1 - \mathrm{SSe/SSt} \qquad (4)$$

Temelde *SSt* gerçekleşmedeki sapmayı, *SSe* de hatalardaki sapmayı hesap etmektedir. Burada y_i gerçek hayatta olması gereken sonucu, *ort* ortalamayı, N_i YSA'nın elde ettiği sonucu, *n* ise önceden bahsedildiği gibi gözlemlenen durumların sayısını belirtmektedir.

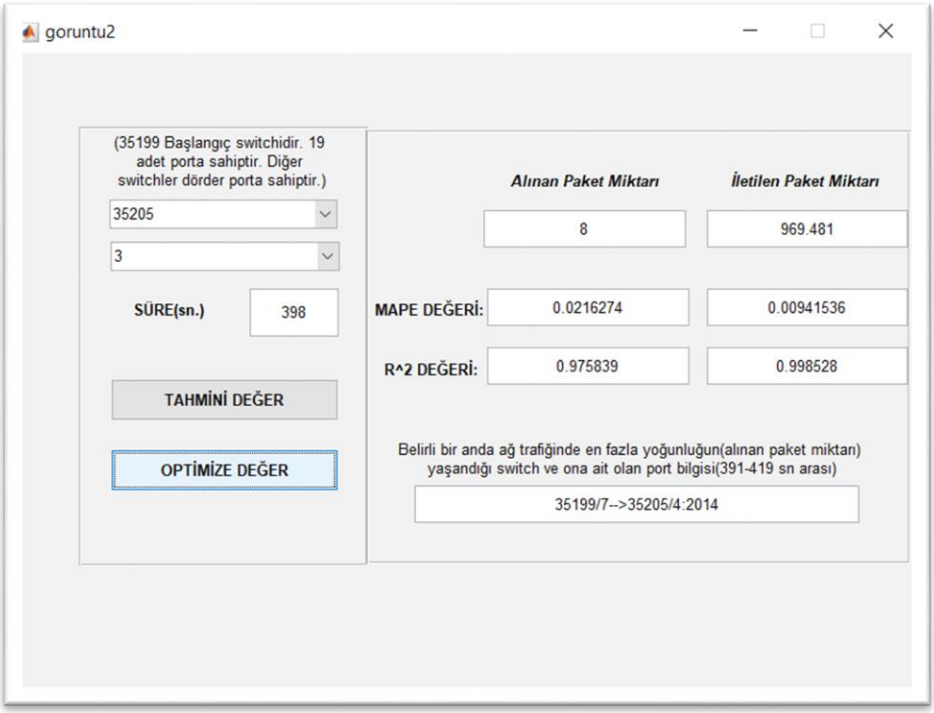
Uygulamadaki topolojilerin MAPE ve R^2 fonksiyonları baz alınarak alınan ve iletilen paket miktarları için doğruluk değerleri ayrı ayrı hesaplanmıştır. Bütün topolojiler için bu değerlerle ilgili ölçüm sonuçları Değirmenci (2018) tarafından hazırlanan tezde yer almaktadır. Örnek olması açısından alınan paket miktarlarına göre bulunan MAPE

ölçümleri Şekil 4'te grafiksel olarak gösterilmektedir. Tablo 1'deki değerler referans alınarak şekildeki ilk sayısal değerler anahtar sayısını, ikinci sayısal değerler ise makine sayısını ifade etmektedir. Alınan ve iletilen paket miktarlarının ikisi de gözlemlendiğinde ortaya çıkan sonuçlara göre tahmin başarısı en yüksek olan topolojinin 20 anahtar ve 57 makineden oluşan dairesel topoloji olduğu gözlemlenmiştir. Bu topoloji her iki veri miktarı için de en küçük MAPE fonksiyon değerini ve 1'e çok yakın bir R^2 değerini üretmiştir. Doğruluğu en yüksek ikinci topolojinin ise ağaç topolojisi olduğu tespit edilmiştir.



Şekil 4. Alınan paket miktarına göre yapılan tahminlerin doğruluk oranları (Accuracy of estimates based on the received packet amount)

Üçüncü ve son olan kısım ise optimizasyon işlemlerinden oluşmaktadır. Şekil 5'te uygulamanın bu tarafındaki çalışmaya ait görüntü yer almaktadır.



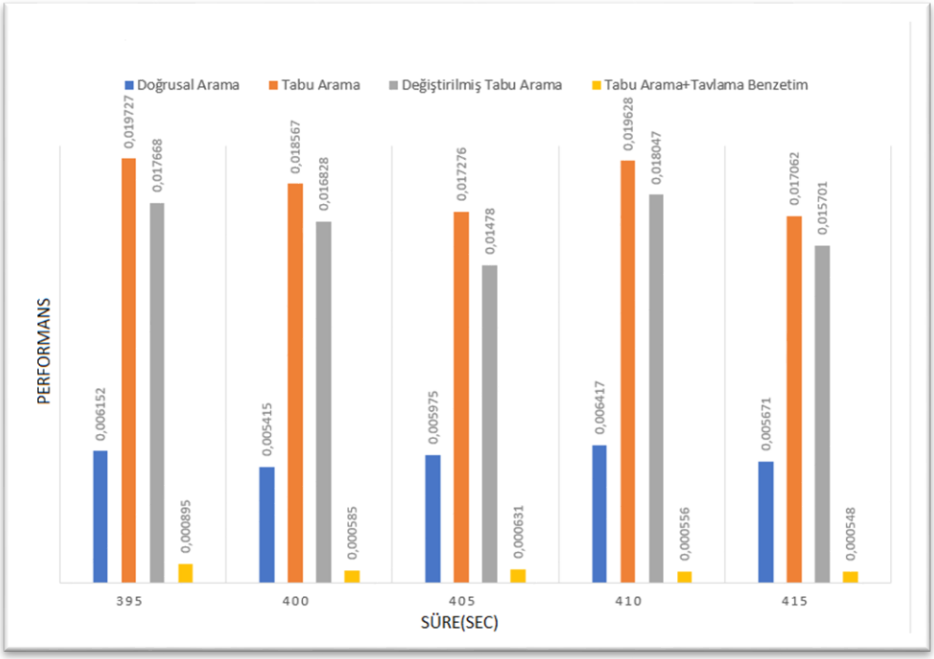
Şekil 5. 398. sn'de ağda en fazla paket akışının olduğu güzergâhın bulunması (Finding the route with the maximum number of packet streams in the network at 398th second)

Bu aşamada ise ikinci kısımda yapılan tahmin işleminin doğruluk oranına göre seçilen en iyi iki topoloji üzerinde paket yoğunluğu en fazla olan güzergâhı bulma problemi optimize edilmiştir. Bu amaç doğrultusunda dört ayrı optimizasyon yöntemi denenmiştir ve bunların performansları kıyaslanmıştır. Bu yöntemler önceden belirtildiği gibi tabu arama, doğrusal arama, değiştirilmiş tabu arama ve karışım algoritmasıdır. Bu algoritmalar farklı zamanlarda çalıştırılarak tek bir ana bağlı kalınmamış olup farklı zamanlardaki sonuçlarla doğruluk oranı artırılmak istenmiştir. Bunun için astgele seçilen 395. sn'deki sonuçlara ek olarak beşer saniye aralıklarla algoritmaların ürettiği sonuçlar gözlemlenmiştir. Böylelikle 395., 400., 405., 410. ve 415. sn'lerde algoritmaların ürettiği sonuçlar kıyaslanmıştır. Farklı zamanlarda elde edilen performans değerleri Tablo 2'de yer almaktadır. Sonuçlar değerlendirildiğinde beş farklı zaman diliminde de en başarılı performansın daima karışım algoritmasında olduğu gözlemlenmiştir. Performans değerleri grafiksel olarak Şekil 6'daki gibidir. Yöntemlerin

işlerliğini test etmek açısından en iyi ikinci doğruluğa sahip olan ağaç topolojisinde de bu fonksiyonlar denenmiştir. Sonuç olarak yine en iyi performansın karışım algoritmasına ait olduğu gözlemlenmiştir.

Tablo 2. 20 anahtar ve 57 makineden oluşan topolojiye ait optimizasyon performansı (Optimization performance for the topology of 20 switches and 57 hosts)

Süre	Doğrusal Arama (Geleneksel arama metodu)	Tabu Arama (Yapay zekâ optimizasyon tekniği)	Değiştirilmiş Tabu Arama	Karışım (Tavlama Benzetimi ve Tabu Arama)
395. sn	0,006152	0,019727	0,017668	**0,000895**
400. sn	0,005415	0,018567	0,016828	**0,000585**
405. sn	0,005975	0,017276	0,014780	**0,000631**
410. sn	0,006417	0,019628	0,018047	**0,000556**
415. sn	0,005671	0,017062	0,015701	**0,000548**



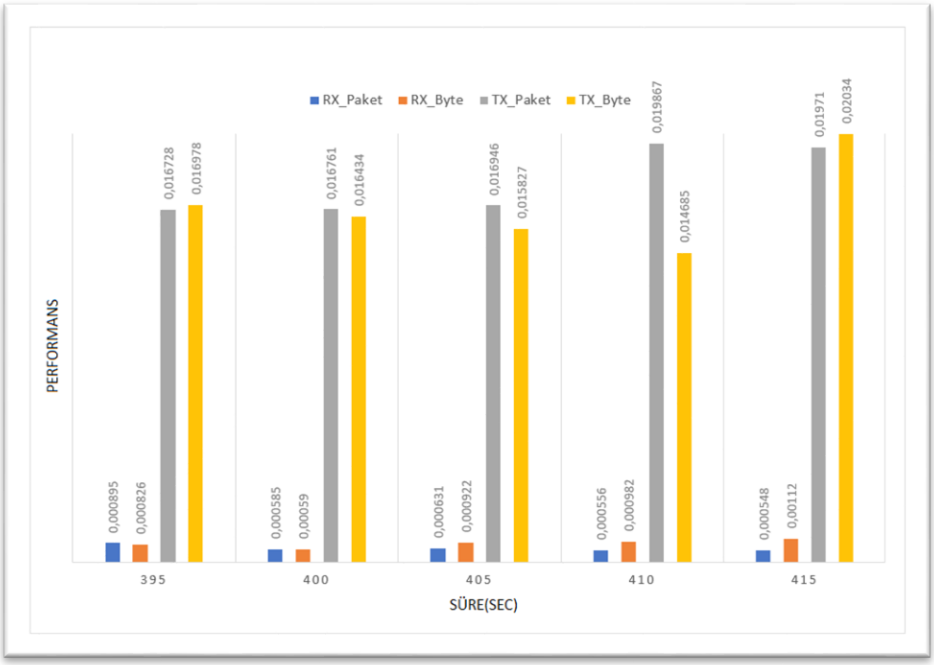
Şekil 6. 20 anahtar ve 57 makineden oluşan daire (halka) topolojisine ait dört ayrı optimizasyon tekniğine göre hesaplanmış performans grafiği (Performance chart computed according to four different optimization techniques of ring topology consisting of 20 switches and 57 hosts)

Alınan paket miktarı (RX_PAKET) göz önünde bulundurularak yapılan optimizasyon işlemlerinin sonucunda karışım algoritmasının en iyi performansı sergileyen algoritma olduğu görüldükten sonra bu algoritma üzerinde farklı değerler ile denemeler yapılmış ve ilgili performans değerleri incelenmiştir. Şöyle ki portlar arasında RX_PAKET'e göre yoğunluğun yani RX_PAKET toplam değerinin en fazla olduğu güzergâh tespit edilmiştir. Diğer denemelerde ise portlar arasında iletilen paket miktarına (TX_PAKET), alınan bayt miktarına (RX_BYTE) ve iletilen bayt miktarına (TX_BYTE) göre de yoğunluğun en fazla olduğu güzergâhlar tespit edilmiş ve sonuçlar karşılaştırılmıştır. Bu denemeler karışım algoritması üzerinden beş farklı zamanda gerçekleştirilmiştir. Denemeler sonucunda elde edilen değerler Tablo 3'te yer almaktadır. Yapılan denemeler sonucunda güzergâhların ve performansların değiştiği gözlemlenmiştir. Bunun sebebi ise RX_PAKET ve TX_PAKET üzerinde yapılan denemeler baz alındığında porta gelen ve porttan çıkan paket miktarının farklılık göstermesidir. Diğer bir senaryo olarak RX_PAKET ve RX_BYTE miktarlarına göre yapılan denemeler sonucunda farklı güzergâhlar ve farklı performans sonucu elde edilmesinin sebebi ise her bir paketin taşıdığı bilginin bayt cinsinden aynı büyüklükte olmamasıdır.

Optimizasyondaki amaç problemin çözümüne giden en doğru yolu en kısa sürede gitmektir. Bu doğrultuda dört ayrı kritere göre hesaplanmış optimizasyon performans değerlerinden en iyi performansın Şekil 7'de gösterildiği gibi RX_PAKET'e göre yapılan optimizasyon işleminden olduğu görülmektedir.

Tablo 3. 20 anahtar ve 57 makineden oluşan daire topolojisine ait dört ayrı kritere göre hesaplanmış optimizasyon performans değerleri (Optimization performance values based on four different criteria of circle topology consisting of 20 switches and 57 hosts)

Süre	RX_PAKET	RX_BYTE	TX_PAKET	TX_BYTE
395. sn	0,000895	**0,000826**	0,016728	0,016978
400. sn	**0,000585**	0,000590	0,016761	0,016434
405. sn	**0,000631**	0,000922	0,016946	0,015827
410. sn	**0,000556**	0,000982	0,019867	0,014685
415. sn	**0,000548**	0,001120	0,019710	0,020340



Şekil 7. 20 anahtar ve 57 makineden oluşan daire topolojisine ait dört ayrı kritere göre hesaplanmış optimizasyon performans grafiği (Optimization performance graph based on four different criteria of ring topology consisting of 20 switches and 57 hosts)

5. Sonuç ve Tartışma (Result and Discussion)

Büyük ağ topolojilerinde yer alan anlamlı ve değerli verinin geleneksel ağ yöntemleriyle ölçülmesi oldukça güçtür. Bu çalışmada YTA sayesinde yazılımlar aracılığıyla bu verilerin elde edilmesi ve işlenmesi üzerine bir uygulama yapılmıştır. YTA uygulamalarının klasik ağ yapılarına göre en temel farkı kullanılan denetleyici yazılımı ile fiziksel ağ cihazlarının birbirinden ayrılmasından dolayı programlar aracılığıyla işlem kontrollerinin etkin yapılabilmesidir. Ayrıca klasik ağlar üzerindeki paket iletimi kısmı YTA'da akış şeklinde gerçekleşmektedir. Yani ağ içinde birden çok düğüme paket iletimi gerçekleştiği için bu da yoğun bir akış trafiğine sebebiyet vermektedir. Bu esnada YTA denetleyicisi olarak Floodlight kullanılarak ağ akış bilgileri elde edilmiştir. Bu bilgiler uygulamada veri seti olarak kullanılmıştır. En yoğun paket trafiğinin yaşandığı güzergâh RX_PAKET, TX_PAKET, RX_BYTE ve TX_BYTE metriklerine göre ayrı ayrı bulunmuştur. Yapay zekâ optimizasyon teknikleri bu dört kriterin her birine ayrı ayrı uygulanarak en yoğun trafiğin yaşandığı güzergâh başarılı bir şekilde belirlenmiştir. Kriterlere göre elde edilen sonuçlar incelendiğinde RX_PAKET'e göre yapılan işlemlerin performans açısından daha başarılı olduğu görülmüştür. Optimizasyon kısmında ise dört farklı teknik kullanılarak sonuçlar değerlendirilmiştir. Bu tekniklerden birisi doğrusal arama metodu yani geleneksel yöntem iken geri kalanlar yapay zekâ optimizasyon tekniklerinden olan tabu arama ve tavlama benzetimi algoritmaları ve bunların önerilen türevleridir. Sonuçlar incelendiğinde geleneksel arama metodunun tabu arama metodundan daha iyi performans gösterdiği görülmektedir. Buradan da yapay zekâ optimizasyon tekniklerinin her problem için en iyi çözüm olmadığı sonucuna ulaşılmaktadır. Öte yandan bu çalışmada önerilen iki yöntemden biri olan karışım algoritması doğrusal arama metodundan daha iyi bir performans sergilemiştir. Böylece son aşama için karışım algoritmasıyla tümleşik olan YSA modeli kullanılarak ağın belirli bir anında yaşadığı paket trafik yoğunluğu tahmin edilmiştir. Bu kısım ise RX_PAKET, TX_PAKET, RX_BYTE ve TX_BYTE değerleri göz önünde bulundurularak gerçekleştirilmiştir. Tahminler neticesinde yüksek doğruluk elde edilmiştir.

Yapılan bu çalışma geliştirilmeye elverişlidir. Şöyle ki; ileri aşamalarda yoğunluğu tespit edilen güzergâh üzerinde YTA'lar kullanılarak trafik yoğunluğunu azaltma gerçekleştirilebilir. Diğer bir fikir ise farklı optimizasyon teknikleri ile bu çalışmadaki teknikleri kıyaslamaktır.

Çıkar Çatışması (Conflict of Interest)

Yazarlar tarafından herhangi bir çıkar çatışması beyan edilmemiştir. No conflict of interest was declared by the authors.

Kaynaklar (References)

Abu Salam, Z. K. A., Keskin, M. E., 2018. Yapay Sinir Ağları ile Dibis Barajı'nın Seviye Tahmini. Journal of Engineering Sciences and Design, 6(4), 564-569.

Aladağ, Ç. H., 2009. Yapay Sinir Ağlarının Mimari Seçimi İçin Tabu Arama Algoritması. Doktora Tezi. Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü, İstatistik Anabilim Dalı, Ankara, Türkiye.

Çakır, B., 2006. Stokastik İşlem Zamanlı Montaj Hattı Dengeleme İçin Tavlama Benzetimi Algoritması. Yüksek Lisans Tezi, Endüstri Mühendisliği, Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

Değirmenci, Ş., 2018. Yazılım Tanımlı Ağ Denetleyicisine Bağlı Paket Trafik Gözetleme Uygulaması ve Optimizasyonu. Yüksek Lisans Tezi. İstanbul Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, İstanbul, Türkiye.

Floodlight. Project Floodlight-Floodlight Controller-Installation Guide, [Online]. Available: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343544/Installation+Guide#InstallationGuide-Linux.

Gendreau, M., Potvin, J.-Y., 2010. Handbook of Metaheuristics, 2nd ed. Springer Science+Business Media.

Haykin, S., 2009. Neural Networks and Learning Machines, Third ed. New Jersey, USA: Pearson Education, Inc., [Online]. Available: http://dai.fmph.uniba.sk/courses/NN/haykin.neural-networks.3ed.2009.pdf.

Kalınlı, A., 2003. Elman Ağının Benzetilmiş Tavlama Algoritması Kullanarak Eğitilmesi. Erciyes Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 19(1), 28-37.

Kılıçaslan, K. Isıl İşlem Algoritması (Simulated Annealing), [Online]. Available: ders.kilicaslan.nom.tr/doc/19/52/Isıl%20İşlem%20Algoritması.docx

Kovačević-Vujčić, V. V., Čangalović, M. M., Ašić, M. D., Ivanović, L., Dražić, M., 1999. Tabu Search Methodology in Global Optimization. Computers & Mathematics with Applications, 37, 125-133.

Kutay, M., Ercan, T., 2016. An Overview of Software Defined Campus Networks. Selçuk Üniversitesi Mühendislik, Bilim ve Teknoloji Dergisi, 4(2), 155-164.

Michalska, M., Zufferey, N., Mattevelli, M., 2016. Tabu Search For Partitioning Dynamic Dataflow Programs. The International Conference on Computational Science, Procedia Computer Science, 80, 1577-1588.

Mourrain, B., Pavlidis, N. G., Tasoulis, D. K., Vrahatis, M. N., 2006. Determining the Number of Real Roots of Polynomials Through Neural Networks. Computers & Mathematics with Applications, 51(3-4), 527-536.

Niyaz, Q., Sun, W., Alam, M., 2015. Impact on SDN Powered Network Services Under Adversarial Attacks. The 2015 International Conference on Soft Computing and Software Engineering (SCSE 2015), Procedia Computer Science, 62, 228 – 235.

Oommen, B. J., Rueda, L. G., 2005. A Formal Analysis of Why Heuristic Functions Work. Artificial Intelligence, 164, 1-22.

Sezer, S., Scott-Hayward, S., Chouhan, P. K., Fraser, B., Lake, D., Finnegan, J., Viljoen, N., Miller, M., Rao, N., 2013. Are We Ready for SDN? Implementation Challenges for Software-Defined Networks. IEEE Communications Magazine, 51(7), 36-43.

Shih, H. S., Wen, U. P., Lee, E. S., Lan, K. M., Hsiao, H. C., 2004. A Neural Network Approach to Multiobjective and Multilevel Programming Problems. Computers & Mathematics with Applications, 48, 95-108.

Staub, S., Karaman, E., Kaya, S., Karapınar, H., Güven, E., 2015. Artificial Neural Network and Agility. World Conference on Technology, Innovation and Entrepreneurship, Procedia - Social and Behavioral Sciences, 195, 1477 – 1485.

Taşhan, B., 2017. Road Lane Detection System With Convolutional Neural Network. Master's Thesis. Bahçeşehir University, Graduate School of Natural and Applied Sciences, Computer Engineering, İstanbul, Turkey.

Ünsal, Ö., Yiğit, T., 2018. Optimization of School Bus Routing Problem by Using a Method with Artificial Intelligence and Clustering Techniques, Journal of Engineering Sciences and Design, 6(1), 7-20.

Xia, W., Wen, Y., Foh, C. H., Niyato, D., Xie, H., 2015. A Survey on Software-Defined Networking. IEEE Communication Surveys & Tutorials, 17(1), 27-51.

Yazar, S., 2013. Ağ Anahtarlarında OpenFlow Protokolü ve Pox Denetleyicisi Kullanımı. Yüksek Lisans Tezi. Trakya Üniversitesi. Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, Edirne, Türkiye.

Yıltaş, D., 2007. Alçak Yörüngedeki Uydu Sistemlerinde Yeni Bir Yönlendirme Algoritmasının Tasarımı. Doktora Tezi. İstanbul Üniversitesi. Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, İstanbul, Türkiye.

Yiğit, T., Aydemir, M., 2018. Container Loading Problem Optimization By Using Genetic Algorithm Without Rotating The Package, Journal of Engineering Sciences and Design, 6(1), 21-28.

Zhao, X., Lin, W., Yu, C., Chen, J., Wang, S., 2013. A New Hybrid Differential Evolution with Simulated Annealing and Self-Adaptive Immune Operation.