ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696

Uluslararası Sürdürülebilir Mühendislik ve Teknoloji Dergisi
International Journal of Sustainable Engineering and Technology
Sayı: 1, Cilt: 9, (2025), 58 – 69
Araştırma Makalesi – Research Article

BULUT BİLİŞİM AĞ TOPOLOJİLERİNDE PERFORMANS VE HATA TOLERANSI ANALİZİ

*Hüseyin Taner GÖKMEN[1], Mevlüt ERSOY[2]

[1]Süleyman Demirel Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, Isparta
[2]Süleyman Demirel Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Isparta

(Geliş/Received: 21.05.2025, Kabul/Accepted: 15.06.2025, Yayınlanma/Published: 30.06.2025)

ÖZ

Günümüzde artan veri trafiği ve ölçeklenebilir sistem gereksinimleri doğrultusunda, bulut bilişim altyapılarında kullanılan ağ topolojilerinin performansı büyük önem taşımaktadır. Bu bağlamda, özellikle veri merkezlerinde tercih edilen Fat Tree, Z-Fat Tree ve RUFT-PL (Reduced Unidirectional Fat Tree with Parallel Links) topolojileri, çok katmanlı yapıları hem ağ içi paket iletim performansı hem de bağlantı arızalarına karşı gösterdikleri dayanıklılık açısından dikkat çekmektedir. Bu çalışmada, söz konusu topolojilerin performansları karşılaştırmalı olarak değerlendirilmiş ve farklı bağlantı arızası senaryoları altında gösterdikleri iletim başarısı, gecikme süreleri ve ağ dayanıklılığı analiz edilmiştir. Elde edilen bulgular, paralel bağlantıların hata toleransını artırdığını, tam bağlantılı yapılarda dengeli performans sağlandığını ve düşük bağlantı yoğunluğunun arıza hassasiyetini yükselttiğini göstermiştir.

Anahtar Kelimeler: Bulut Bilişim, Ağ Topolojileri, Hata Toleransı, Ağ Performansı

PERFORMANCE AND FAULT TOLERANCE ANALYSIS IN CLOUD COMPUTING NETWORK TOPOLOGIES

ABSTRACT

With the increasing volume of data traffic and the growing need for scalable systems, the performance of network topologies used in cloud computing infrastructures has become critically important. In this context, the Fat Tree, Z-Fat Tree, and RUFT-PL (Reduced Unidirectional Fat Tree with Parallel Links) topologies, which are especially preferred in data centers, stand out with their multilayered structures in terms of both packet transmission performance and resilience against link failures. In this study, the performance of these topologies is evaluated comparatively under various link failure scenarios, focusing on transmission success, latency, and network robustness. The findings reveal that parallel links enhance fault tolerance, fully connected structures offer balanced performance, and reduced connection density increases sensitivity to failures.

Keywords: Cloud Computing, Network Topologies, Fault Tolerance, Network Performance

***Sorumlu Yazar/Corresponding Author:** huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696

1. Giriş (Introduction)

Bulut bilişim, bilgi teknolojisinin bir üründen hizmete dönüşmesini sağlayarak yazılım, platform ve altyapı kaynaklarını internet üzerinden talep üzerine ölçeklenebilir hizmetler olarak sunar. Uzak bir konumdaki kaynaklara erişim, bunların yapılandırılması ve yönetilmesi, işletmelere fiziksel donanım ve yazılım yükleme ihtiyacını ortadan kaldırır. ABD Ulusal Standartlar ve Teknoloji Enstitüsü (NIST)'e göre, bulut bilişim; sunucular, ağlar ve uygulamalar gibi bilgi işlem kaynaklarına kolayca erişim sağlayan, minimum yönetim çabasıyla hızla sağlanabilen bir modeldir. Bulut bilişim, işletmelerin ihtiyaçlarına esnek çözümler sunarak büyük miktarda veriyi yönetmeyi kolaylaştırır [1-4]. Bulut bilişim altyapısı, kullanıcılara bulut bilişim kaynakları ve hizmetleri sunmak için gerekli olan bilgisayarları, depolama aygıtlarını, ağ tesislerini ve diğer ilgili bileşenleri içerir [5].

Bulut bilişim, son kullanıcılara talep üzerine çeşitli hizmetler sunarak, işletmelerin ve kullanıcıların uygulamaları fiziksel makinelerde barındırmadan kullanmalarını sağlar ve internet üzerinden gerekli kaynaklara kolay erişim imkanı tanır. Hızla gelişen bu teknoloji, giderek daha fazla işletme ve kurumsal uygulamanın barındırılması için tercih edilmektedir [6,7]. Ancak, bulut tabanlı hizmetlerin yaygınlaşması, hem servis sağlayıcılar hem de kullanıcılar için hizmet güvenilirliği ve kullanılabilirlik sorunlarını gündeme getirmektedir. Yüksek performans, ölçeklenebilirlik, güvenilirlik ve verimlilik gibi avantajlar sunduğu halde, performansla ilgili sorunlar, özellikle hata toleransı gibi kritik unsurların etkin bir şekilde ele alınmasını zorunlu kılmaktadır. Bulut ortamında, arızalar ve performans düşüşleri, çeşitli hata türlerinin bir sonucu olarak ortaya çıkabilir [8,9].

Genellikle hata, arıza ve başarısızlık terimleri birbiriyle karıştırılarak kullanılır. Ancak, Hata Toleranslı Hesaplama terimlerinde, bu kavramların her birinin kendine özgü bir anlamı vardır. Fault, bir sistemde herhangi bir anda ortaya çıkabilen temel kusur ya da bozukluktur. Eğer bu arıza ele alınmazsa, sistemin başarısız olmasına yol açabilir. Error, bir arızanın sistemde gözlemlenebilir etkisidir. Failure, arızaların ve hataların ele alınamaması durumunda, sistemin beklenen işlevini yerine getirememesi durumudur. Bulut bilişim, dağıtılmış bilgi işlem yapısı nedeniyle kısmi arızalarla karakterize edilir. Bu arızalar, sistemin tamamen çökmesine veya performans düşüşlerine yol açabilir. Sistem bileşenlerinin, düğümlerin veya ağ elemanlarının herhangi birinde meydana gelen arızalar, sistemin çalışmasını tamamen durdurmaz, ancak kısmi bir aksama oluşturabilir. Bu durum, bulut sistemlerinin güvenilir ve sağlam olmasına rağmen, yüksek performanslı bilgi işlem için uygun arıza toleransı mekanizmalarının zorunlu hale gelmesini gerektirir. Hata toleransı (HT), bir sistemin arızalara rağmen beklenen işlevleri yerine getirme yeteneğini ifade eder ve sistemin, donanım veya yazılım bileşenlerinde oluşan arızalara, güç kesintilerine veya diğer olumsuzluklara karşı dayanıklı olmasını sağlar. Bu şekilde, sistemin arızalardan veya hatalardan etkilenmeden sürekli hizmet sunması sağlanır.

Literatürde, bulut bilişim ortamlarında hata toleransı sağlamak amacıyla birçok yöntem kullanılmıştır. Bu yöntemler arasında; kontrol noktası oluşturma [9], görev kopyalama, iş göçü, kendi kendini iyileştirme [10], güvenlik çantası kontrolleri, S-koruması, görevlerin aynı veya farklı kaynaklarda yeniden denenmesi ve yeniden gönderilmesi, zamanlama kontrolü, kurtarma iş akışı, yazılım yenileme, yeniden yapılandırma, sanal makineler (VM) için yük paylaşımı ve devralma [1] yer almaktadır. Ayrıca kötü arka uç algılama, yumuşak sistemler için kesin olmayan hesaplamaların bir sonraki aşamaya geçirilmesi, kodlama ve kod çözme gibi çeşitli teknikler de uygulanmaktadır. Hata ya da arıza algılama tekniklerinin evrimi, başlangıçta sistem performans kayıtlarının analizine dayanmıştır. Ancak bu yöntemlerin bazı dezavantajları ve veri madenciliğine dayalı algılama sınıflandırmasındaki verimsizlikler nedeniyle, istatistiksel öğrenme teorileri hata algılama amacıyla kullanılmaya başlanmıştır. Bu bağlamda, makine öğrenimi tabanlı tekniklerin kullanımı giderek yaygınlaşmıştır. Destek Vektör Makineleri (SVM), Destek Vektör Veri Açıklaması (SVDD), K-ortalama kümeleme, Bayes sinir ağları gibi yöntemler bu alanda yaygın olarak tercih edilmektedir [11]. Bu yöntemler, hata tespitinde daha yüksek doğruluk oranları ve sistemlere uyum sağlayabilme özellikleri nedeniyle öne çıkmaktadır.

Bu çalışmada, bulut bilişim altyapılarında yaygın olarak kullanılan üç farklı çok katmanlı ağ topolojisi olan Fat Tree, Z-Fat Tree ve RUFT-PL modelleri incelenecek ve karşılaştırmalı olarak analiz edilecektir. Söz konusu topolojiler, veri merkezlerinde yüksek bant genişliği, düşük gecikme, esneklik ve hata toleransı gibi temel gereksinimleri karşılamaya yönelik olarak tasarlanacaktır. Çalışmada, bu yapıların performansları, özellikle paket iletim başarısı, gecikme süreleri ve bağlantı arızalarına karşı dayanıklılık

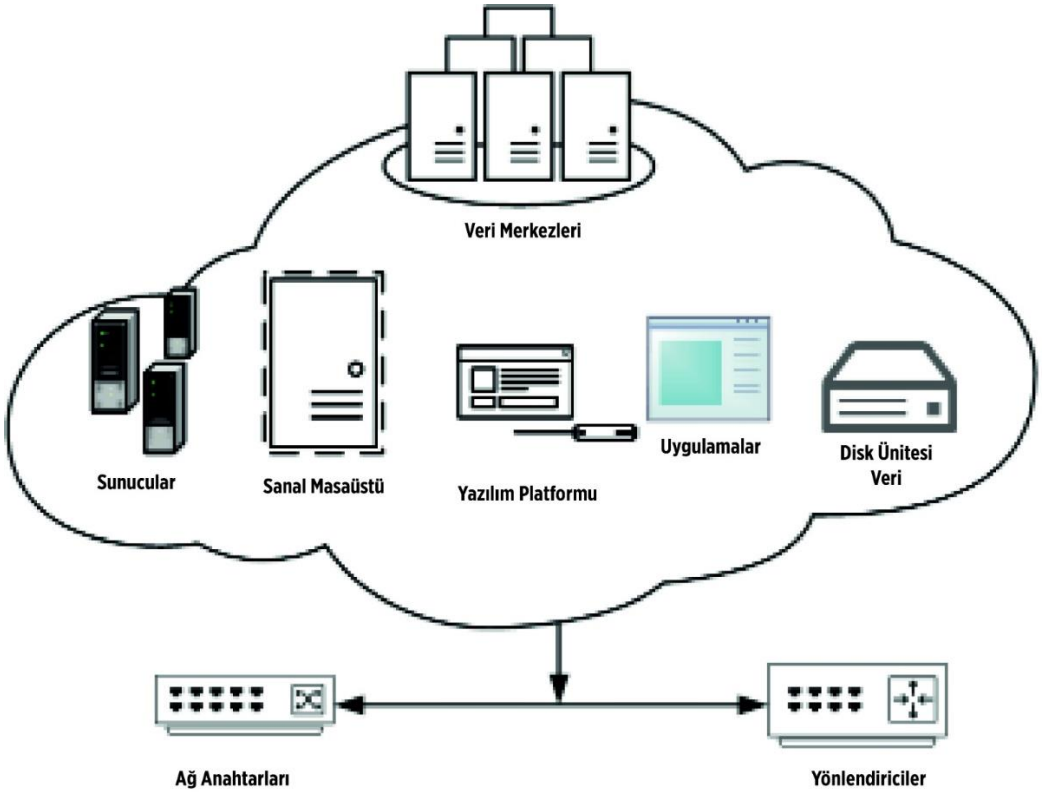
*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696

kriterleri çerçevesinde değerlendirilecektir. Her bir topoloji için Python ortamında geliştirilen simülasyon senaryoları ile, belirli arıza oranlarında ağın başarımı test edilecek ve sonuçlar grafiksel olarak sunulacaktır. Elde edilen veriler, ağ mimarisi seçiminde topolojik yapıların sistem güvenilirliği ve performansı üzerindeki etkilerini ortaya koyacaktır.

2. Bulut Bilişim Modelleri (Cloud Computing Models)

Bulut bilişim donanım bileşenleri çoğunlukla kurumsal veri merkezlerinde bulunur. Bunlar, güvenlik duvarları, anahtarlar ve yönlendiriciler gibi kararlı depolama ve ağ aygıtları sunan çok çekirdekli sunucuları, katı hal sürücülerini ve sabit disk sürücüsünü kapsamaktadır. Şekil 1.'de bulut bilişim genel mimarisi görülmektedir.

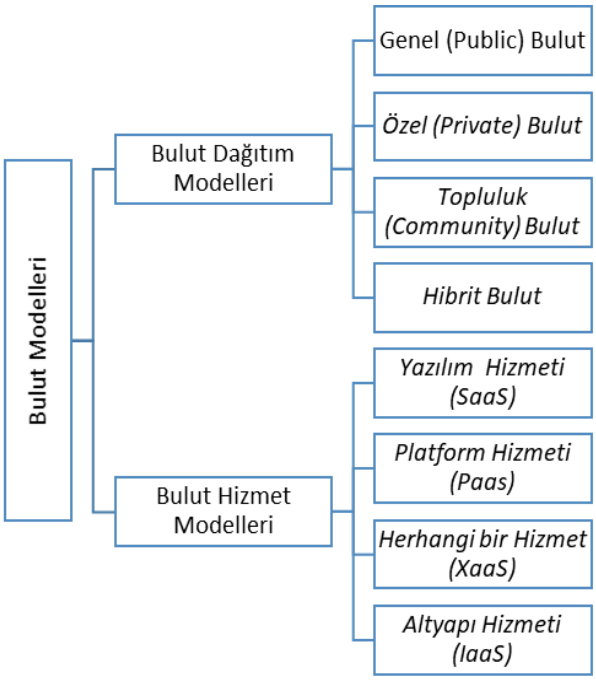


Şekil 1. Bulut Bilişim genel mimarisi

Bu donanım bileşenlerinin dışında, sanallaştırma yazılımı gibi bulut hizmeti modelini destekleyen yazılım bileşenleri de bulut bilişim altyapısı olarak adlandırılır. Sanallaştırma yazılımı, bulut kaynaklarının bir soyutlamasını sağlar ve bu kaynakları genellikle Uygulama Program Arayüzleri (API – Application Programming Interfaces) veya diğer komut satırı ve/veya grafik arayüzleri kullanarak kullanıcılara sunar. Bulut hizmet sağlayıcıları (CSP – Cloud Service Providers) tarafından barındırılan sanallaştırılmış kaynaklar, genellikle İnternet üzerinden kullanıcılara sunulur.

*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696



Şekil 2. Bulut Bilişim Modelleri

Genel, Özel, Topluluk ve Hibrit Bulut olmak üzere dört temel bulut bilişim dağıtım modeli bulunmaktadır [12]. Genel Bulut, altyapı ve yazılım kaynaklarının internet üzerinden üçüncü parti sağlayıcılar (örneğin AWS, Azure, GCP) aracılığıyla sunulduğu, çok kullanıcılı ve maliyet etkin bir modeldir. Kaynaklar ortak kullanılarak kullanıcılar yalnızca tükettikleri hizmet kadar ödeme yapmaktadırlar [13]. Özel Bulut yapıları, yalnızca bir organizasyona tahsis edilen, genellikle kurum içi veri merkezlerinde ya da özel altyapılarda barındırılan bir modeldir [14]. Daha fazla kontrol, güvenlik sağlamaktadır. Bu durum finansal kurumlar, devlet kurumları gibi yüksek güvenlik gerektiren yapılar için tercih edilmektedir. Ancak, kurulum ve bakım maliyetleri yüksektir. Topluluk Bulut, benzer regülasyonlara, güvenlik ve işlem gereksinimlerine sahip organizasyonlar arasında paylaşılan bir model olup, sağlık, eğitim ve kamu gibi sektörlerde maliyetleri paylaşarak güvenli bir ortak ortam sunmaktadır [15]. Son olarak, Hibrit Bulut modeli, özel ve genel bulut yapılarının birlikte kullanıldığı esnek bir çözümdür. Kritik veriler özel bulutta saklanırken, daha az hassas işlemler için genel bulut kaynakları kullanılır [16]. Bulut bilişim hizmet modelleri, kuruluşların dijital altyapı ve yazılım ihtiyaçlarını esnek, ölçeklenebilir ve maliyet etkin bir şekilde karşılamak amacıyla farklı katmanlarda sunulmaktadır.

Bulut bilişim hizmet modelleri, organizasyonların dijital altyapı ve yazılım ihtiyaçlarını esnek, ölçeklenebilir ve maliyet etkin bir şekilde karşılamak amacıyla farklı katmanlarda sunulmaktadır. Altyapı Hizmeti Olarak Sunulan Hizmet (IaaS) modeli, kullanıcıların fiziksel donanım yatırımı yapmadan sanal sunuculara, veri depolama alanlarına ve ağ kaynaklarına internet üzerinden erişmesini sağlamaktadır [17]. Bu modelde, donanımın yönetimi hizmet sağlayıcısı tarafından gerçekleştirilirken, kullanıcılar sistemler üzerinde geniş kontrol imkanına sahip olur. Platform Hizmeti Olarak Sunulan Hizmet (PaaS) ise altyapı yönetimini tamamen soyutlayarak yazılım geliştiricilere uygulama geliştirme, test etme ve dağıtma süreçlerini yönetebilecekleri bütünleşik bir ortam sunmaktadır [18]. Yazılım Hizmeti Olarak Sunulan Hizmet (SaaS) modeli ise son kullanıcıya, herhangi bir kurulum yapmaya gerek kalmadan doğrudan internet üzerinden erişilebilen yazılım çözümleri sunmaktadır [19]. Yazılımların tüm bakım, güncelleme, yedekleme gibi teknik işlemler hizmet sağlayıcısı tarafından yürütülmektedir. Herhangi bir Hizmet Olarak Sunulması (XaaS) yaklaşımı, güvenlik, analitik, veri yönetimi, ağ servisleri gibi çok daha geniş bir bilgi teknolojisi kaynaklarının servis olarak sunulmasını mümkün kılmaktadır [20].

3. Bulut Bilişim Ağ Topolojileri (Cloud Computing Network Topologies)

Bulut bilişim ağ topolojileri, veri merkezleri ve dağıtık sistemler arasında yüksek bant genişliği, düşük gecikme ve ölçeklenebilirlik sağlayarak, bulut hizmetlerinin güvenilir ve verimli bir şekilde sunulmasını mümkün kılan mimari yapılardır [21]. Bu bağlamda, farklı performans ihtiyaçlarına ve sistem

*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696

gereksinimlerine göre çeşitli topoloji modelleri geliştirilmiştir. Bu modeller arasında donanım seviyesinde yüksek kapasiteli anahtarların omurgayı oluşturduğu anahtar (switch) merkezli topolojiler (örneğin Spine-Leaf [22], Fat-Tree [23] ve RUFT-PL [24]) ile her sunucunun birden çok ağ arabirimi üzerinden paket iletimi ve yönlendirme işlevine katıldığı sunucu-merkezli topolojiler (BCube [25], DCell [26], Jellyfish [27] vb.) öne çıkar. Switch-merkezli mimariler, ASIC tabanlı switch donanımları sayesinde uçtan uca düşük gecikme ve yüksek bant genişliği sunmaktadır. Sunucu-merkezli yaklaşımlar her bir sunucuyu hem hesaplama hem de yönlendirme düğümü haline getirerek ağın ölçeklenebilirliğini ve hata toleransını artırmaktadır. Her iki model de omurga-yaprak (spine-leaf) veya çok katmanlı Clos benzeri düzenler temelinde şekillenir. Böylece artan trafik yükü ve dağıtık uygulama gereksinimleri karşılanırken, kaynak kullanımı ve yedeklilik dengesi optimize edilir. Fat Tree, Z-Fat Tree, RUFT, RUFT-PL, VL2, DCell, BCube ve Clos gibi topolojiler [28], ağ yapısının esnekliği, hata toleransı, maliyet etkinliği ve trafik yönetimi gibi farklı avantajlar sunmaktadır. Bu topolojiler, özellikle bulut bilişim ortamlarında büyük veri trafiğini dengelemek, ağ darboğazlarını önlemek ve hizmet sürekliliğini sağlamak amacıyla tercih edilmektedir. Bu modeller, hem klasik hem de gelişmiş yapıları temsil ederek bulut bilişim altyapılarının güçlü ve zayıf yönlerinin analizine olanak sağlamaktadır.

Bulut bilişim ağ topolojileri, genellikle çok katmanlı bir mimari yapı üzerine kuruludur. Bu yapı veri merkezlerinde ağ trafiğinin verimli, güvenli ve ölçeklenebilir bir şekilde yönetilmesini sağlar. Özellikle switch merkezli topoloji olan Fat Tree, Z-Fat Tree ve RUFT-PL gibi topolojilerde yaygın olarak kullanılan bu yapı, Uç (edge) katmanı, Toplama (aggregation) katmanı ve Çekirdek (core) katmanı olmak üzere üç temel katmandan oluşur [28]. Edge katmanı, ağın en alt seviyesinde yer alır ve sunucuların doğrudan bağlandığı anahtarlardan oluşur. Yerel veri trafiği bu katmanda başlar ve üst katmanlara iletilir. Toplama katmanı, uç ile çekirdek katmanları arasında köprü görevi görür ve verilerin yönlendirilmesi, filtrelenmesi gibi işlemleri gerçekleştirir. En üstte yer alan core katmanı ise ağın omurgasını oluşturur. Veri merkezi bölümleri arasında yüksek bant genişliğine sahip veri iletimini sağlamaktadır. Bu katmanlı yapı, ağın modülerliğini ve ölçeklenebilirliğini artırırken, aynı zamanda yük dengeleme ve yedeklilik gibi özelliklerle ağın güvenilirliğini güçlendirir. Bulut bilişim altyapılarında binlerce sunucunun etkin ve esnek biçimde birbirine bağlanabilmesi için bu hiyerarşik yapı kritik bir rol oynamaktadır [28].

3.1. Fat tree topolojisi (Fat tree topology)

Fat Tree topolojisi, veri merkezleri ve büyük ölçekli ağ altyapıları için tasarlanmış, çok katmanlı, simetrik ve ölçeklenebilir bir ağ mimarisidir. Bu yapı, geleneksel ağaç (tree) topolojisinin tıkanma (congestion) sorununu çözmek amacıyla geliştirilmiştir. [29]

Fat Tree topolojisi, her biri k portlu anahtarlardan (switch) oluşan simetrik ve çok katmanlı bir ağ mimarisi olarak tanımlanır. k değeri, topolojinin boyutunu belirleyen temel parametredir. Verilen $k \in N$ (tek sayı olmamak üzere), aşağıdaki sayısal özellikler elde edilir;

Her biri $\frac{k}{2}$ edge ve $\frac{k}{2}$ aggregation anahtarı içeren k adet pod bulunur. Toplam edge ve aggregation anahtar sayısı Denklem (1)' e göre tespit edilir.

$$S_{edge} = S_{agg} = kx\frac{k}{2} = \frac{k^2}{2} \quad (1)$$

Toplam çekirdek anahtar sayısı Denklem (2)' ye göre;

$$S_{core} = \left(\frac{k}{2}\right)^2 \quad (2)$$

Toplam sunucu sayısı Denklem (3)' e göre;

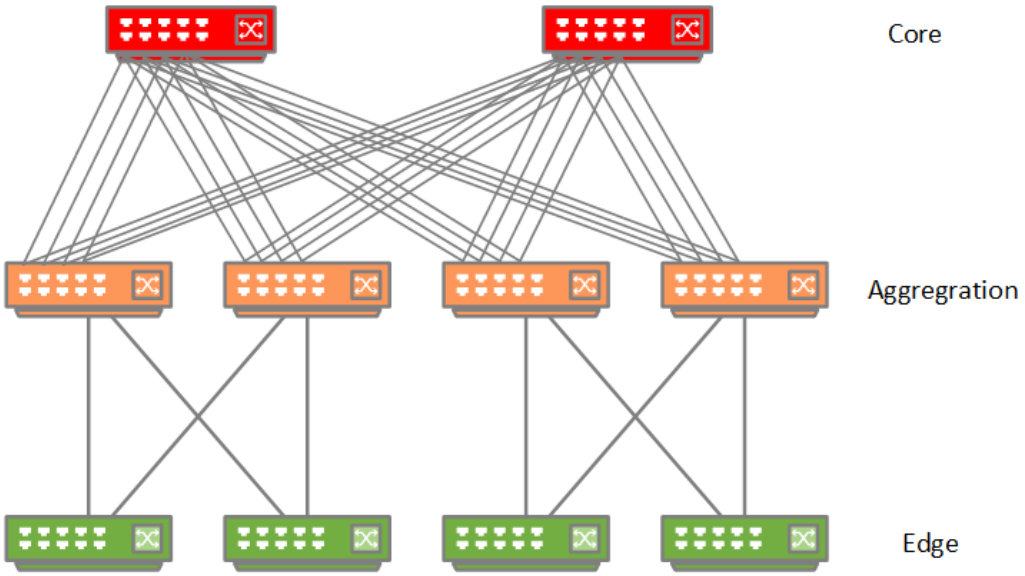
$$N_{host} = S_{edge}.\frac{k}{2} = \frac{k^2}{2}x\frac{k}{2} = \frac{k^3}{2} \quad (3)$$

Elde edilir.

*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696

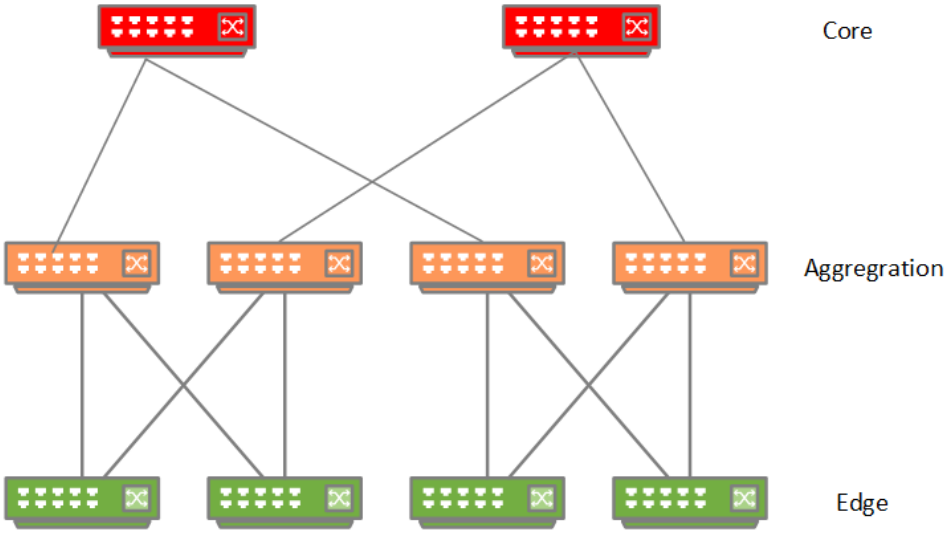
Şekil 3'te k=2 ile elde edilen Fat Tree topolojisi görülmektedir.



Şekil 3. $k = 2$ için Fat Tree Topolojisi

3.2. Z- Fat tree topolojisi (Z- fat tree topology)

Z-Fat Tree topolojisi, klasik Fat Tree mimarisinin bağlantı yoğunluğunu azaltmak ve sistem kaynaklarını daha verimli kullanmak amacıyla geliştirilmiştir. Desen tabanlı (pattern-based) bir ağ mimarisidir. Fat Tree' den farklı olarak edge ile aggregation katmanları arasındaki bağlantılar, belirli bağlantı desenlerine göre sınırlandırılmıştır. Bu sayede ağdaki toplam bağlantı sayısı azaltılmıştır. Bu nedenle donanım maliyetleri ve enerji tüketimi optimize edilmiştir. Full, Cross, Diagonal ve Upper/Lower üçgen Z – Fat Tree topolojisne ait desenlerdir. Full deseninde edge ve aggregation kombinasyonlarının tümü bağlanır. Cross desende yalnızca $(i + j)\ mod\ 2\ = 0$ koşulunu sağlayan edge i ve aggregation j anahtarları bağlanır. Diagonal desende ise sadece $i = j$ veya $i + j = k - 1$ olan çiftler bağlanır[30]. Upper/Lower üçgen desenlerinde ise edge anahtarları sadece üst/alt üçgen bölgedeki aggregation anahtarları ile bağlantılıdır. Şekil 4'te k=2 ile elde edilen Full desenli Z-Fat Tree topolojisi görülmektedir.



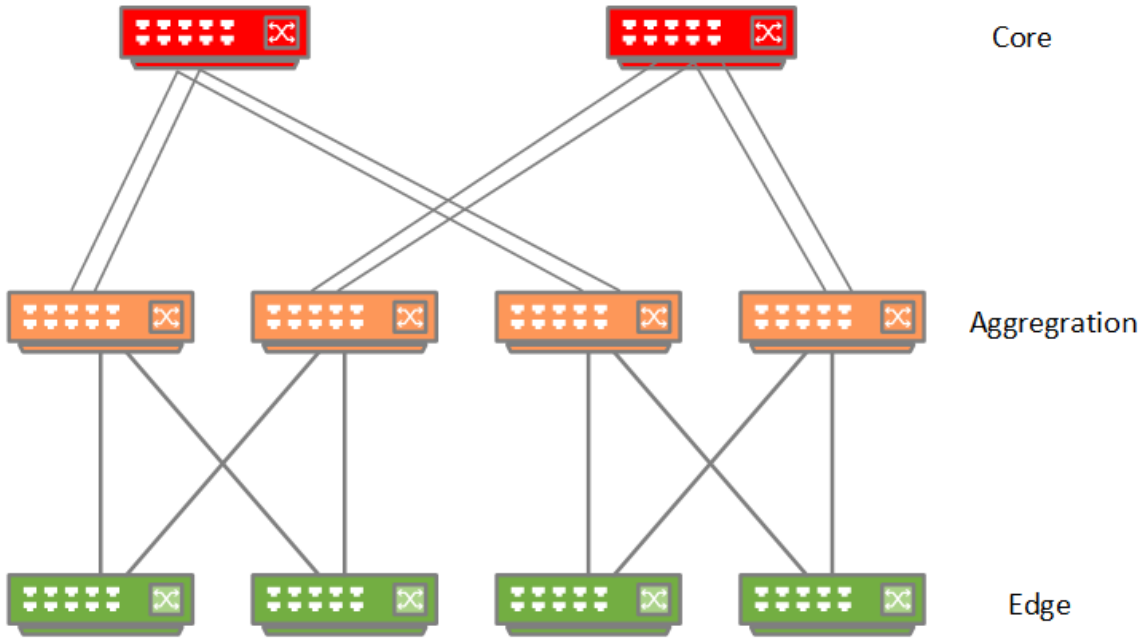
Şekil 4. $k = 2$ için Z-Fat Tree Cross Topolojisi

*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696

3.3. RUFT-PL topolojisi (RUFT-PL topology)

RUFT-PL (Reduced Unidirectional Fat Tree with Parallel Links) topolojisi, klasik Fat Tree mimarisinin basitleştirilmiş ve iyileştirilmiş bir türevidir. "Reduced" ifadesi, gereksiz bağlantıların ortadan kaldırılmasını; "Unidirectional", yönlendirilmiş (tek yönlü) veri akışını; "Parallel Links" ise iki düğüm arasında birden fazla bağlantının (paralel linkin) kullanılmasını ifade etmektedir. Edge katmanındaki anahtarlar, doğrudan aggregation katmanındaki anahtarlara ve core anahtarlarına bağlanmaktadır. Her bir edge–aggregation veya aggregation–core bağlantısı, birden fazla paralel fiziksel hatla desteklenmektedir. Bu yapı sayesinde, aynı hedefe birden fazla alternatif yol oluşmaktadır. Bu alternatif yollar ağ tıkanıklıkları önlemektedir. Aynı zamanda bağlantı arızalarına karşı dayanıklılığı artırmaktadır. [31] Şekil 5'te k=2 ile elde edilen Full desenli RUFT PL topolojisi görülmektedir



Şekil 5. $k = 2$ için RUFT-PL Cross Topolojisi

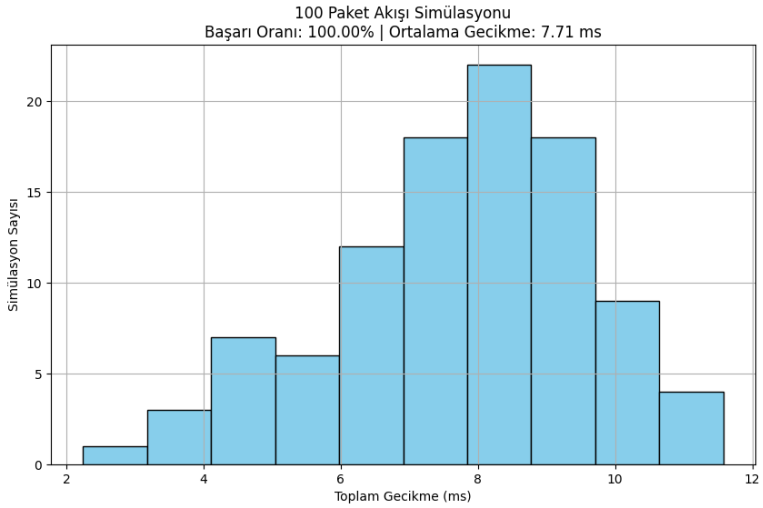
4. Araştırma Bulguları (Research Findings)

Bu çalışmada, Fat Tree, Z-Fat Tree ve RUFT-PL topolojilerinin ağ performansı ve hata toleransı açısından karşılaştırmalı analizi yapılmıştır. Simülasyon süreci Python programlama dili kullanılarak gerçekleştirilmiş ve ağ topolojileri NetworkX [32] kütüphanesi aracılığıyla dinamik olarak modellenmiştir. Her topoloji için belirli bir k=8 parametresi kullanılarak core, aggregation ve edge katmanları ile sunucular oluşturulmuş; bağlantılar ise ilgili topolojinin bağlantı kurallarına göre otomatik olarak atanmıştır. Paket akışı simülasyonu kapsamında, edge katmanından rastgele seçilen bir kaynak ve hedef düğüm arasında en kısa yol belirlenmiş ve bu yol üzerinden iletimin toplam gecikme süresi hesaplanmıştır. Ayrıca, ağ bağlantılarının belirli bir oranı (%0–%60 arası) rastgele devre dışı bırakılarak her topoloji için link arızası senaryoları uygulanmış ve bu durumda paket iletiminin başarı durumu ölçülmüştür. Her senaryo için simülasyon 100 veya 1000 tekrar ile çalıştırılmış, böylece gecikme dağılımı, başarı oranı ve ağ dayanıklılığı istatistiksel olarak değerlendirilmiştir.

Simülasyon sonuçlarına göre, her üç topolojide de paketlerin hedefe başarıyla ulaştığı ve başarı oranının %100 olduğu gözlemlenmiştir. Ancak, bu benzer başarı oranlarına rağmen ağların toplam gecikme süreleri ve dağılım yapıları önemli farklılıklar göstermektedir.

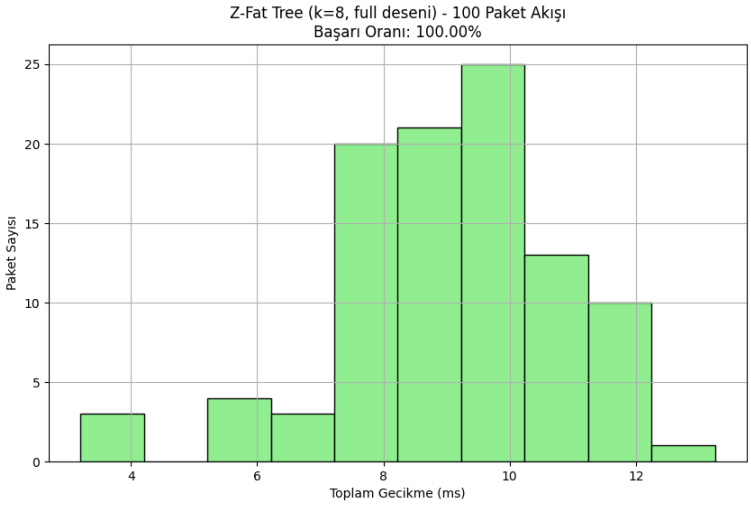
*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696



Şekil 6. Fat-Tree topolojisine göre gecikme dağılımı

Şekil 6'ya göre, Fat Tree topolojisinde, gecikme süreleri oldukça dengeli bir dağılım sergilemiştir. Ortalama gecikme 7.71 ms olarak ölçülmüş, simülasyonların büyük çoğunluğu 6–9 ms aralığında yoğunlaşmıştır. Bu dağılım, Fat Tree'nin düzenli ve eşit sayıda bağlantıya sahip mimarisinin, veri iletiminde tutarlı ve kararlı bir performans sunduğunu göstermektedir.

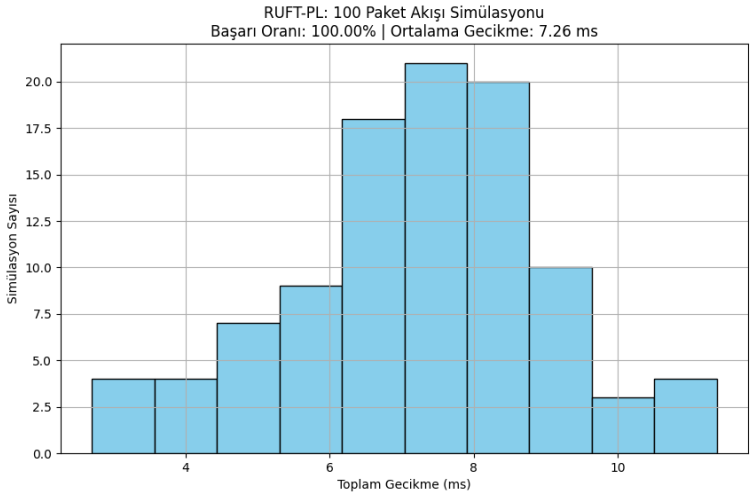


Şekil 7. Z-Fat-Tree topolojisine göre gecikme dağılımı

Şekil 7'ye göre, Z-Fat Tree (full bağlantı desenli) topolojide ise gecikme süreleri biraz daha geniş bir aralıkta dağılmıştır. Başarı oranı yine %100 olmasına rağmen, gecikme sürelerinin sağa doğru kaydığı, yani bazı paketlerin daha yüksek gecikmelere maruz kaldığı görülmektedir. Bu durum, bağlantı sayısının fazla olmasının sağladığı yedekliliğe rağmen, bazı rotaların daha uzun olabileceğini ve bu nedenle gecikme sürelerinin artabileceğini ortaya koymaktadır. Bu yapı, performans açısından esneklik sağlasa da, gecikme açısından daha az öngörülebilir bir karakteristik sergilemektedir.

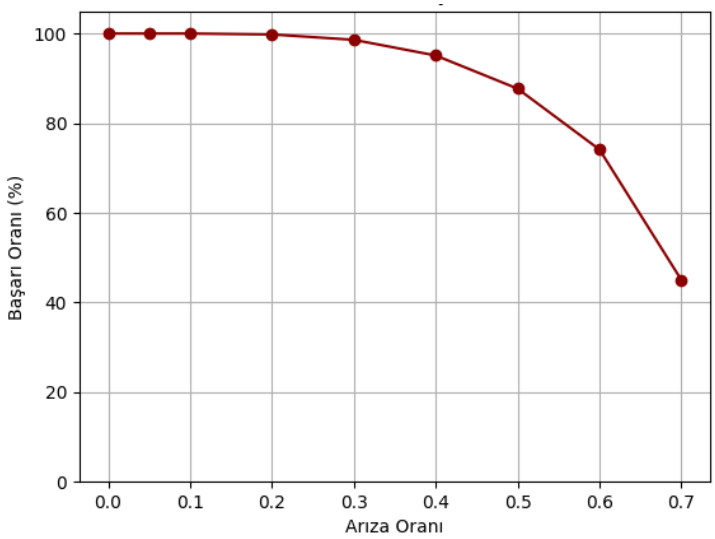
*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696



Şekil 8. RUFT-PL topolojisine göre gecikme dağılımı

Şekil 8'e göre, RUFT-PL topolojisi, en düşük ortalama gecikmeye (7.26 ms) sahip olmasıyla öne çıkmaktadır. Yapısında bulunan paralel bağlantılar, verinin farklı ve daha kısa yollar üzerinden aktarılabilmesine olanak tanımakta ve bu sayede gecikme sürelerinin düşmesini sağlamaktadır. Gecikme dağılımı, Fat Tree'ye benzer şekilde simetrik ve kararlı bir görünüm sunarken, gelişmiş bağlantı yedekliliği sayesinde hem düşük gecikme hem de yüksek güvenilirlik sağlamaktadır.

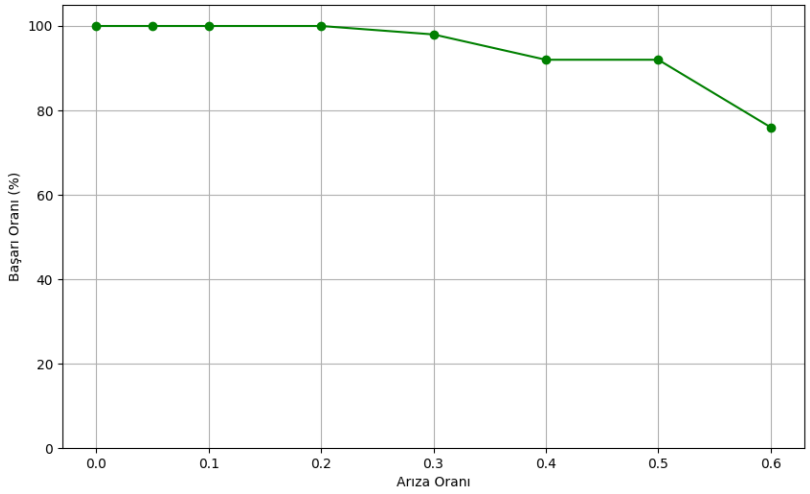


Şekil 9. Fat-Tree topolojisine göre bağlantı arızası ve iletilen paketlerin başarı oranı

Şekil 9'a göre, Fat Tree topolojisi, %0 ile %30 arası arıza oranlarında yüksek başarı oranını korumaktadır. Ancak %40 seviyesinden itibaren başarının hızla düşmeye başladığı, %70 arıza oranında ise başarı oranının %45'in altına indiği görülmektedir. Bu durum, Fat Tree mimarisinin belirli bir yedek bağlantı kapasitesine sahip olduğunu, fakat bu kapasitenin ötesinde kritik bağlantıların kaybıyla iletişimin sekteye uğradığını göstermektedir. Başka bir deyişle, Fat Tree yapısı sınırlı arıza toleransına sahiptir ve yüksek oranda bağlantı kaybı sistemin genel performansını ciddi şekilde etkileyebilmektedir.

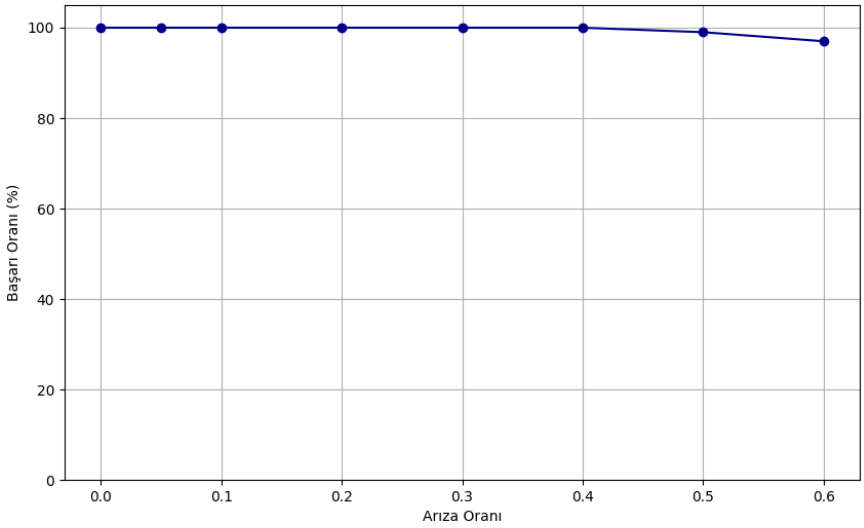
*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696



Şekil 10. Z-Fat-Tree topolojisine göre bağlantı arızası ve iletilen paketlerin başarı oranı

Şekil 10'a göre, Z-Fat Tree topolojisi, full bağlantı deseni kullanılarak tasarlanmış olmasına rağmen, arıza oranı %40'a ulaştığında başarı oranında belirgin bir düşüş göstermektedir. İlk %30'luk arıza oranında sistem oldukça kararlı şekilde çalışmaya devam ederken, %40 ve üzeri oranlarda paket iletiminin sekteye uğradığı anlaşılmaktadır. Z-Fat Tree yapısının daha az bağlantıyla daha hafif bir ağ yapısı sunması, sistem kaynakları açısından avantaj oluştursa da, bu durum aynı zamanda yüksek arıza oranlarında performans kaybına daha açık hale gelmesine neden olmaktadır.



Şekil 11. RUFT-PL topolojisine göre bağlantı arızası ve iletilen paketlerin başarı oranı

Şekil 11'e göre, RUFT-PL topolojisi ise test edilen üç yapı arasında en yüksek dayanıklılığı sergilemiştir. %60 arıza oranına kadar başarı oranı %95'in üzerinde korunmuştur. Bu yapı, edge ve aggregation katmanları ile aggregation ve core katmanları arasında birden fazla paralel bağlantı içerdiğinden, arızalı bağlantılar aktif iletişimi büyük ölçüde etkilememektedir. Bu durum, RUFT-PL topolojisini özellikle yüksek güvenilirlik, yedeklilik ve servis sürekliliği gerektiren sistemler için son derece uygun kılmaktadır. Ağın önemli bir kısmı devre dışı kalsa dahi, alternatif yollar sayesinde iletişim devam edebilmektedir.

5. Tartışma ve Sonuç (Discussion and Conclusion)

Bu çalışmada, üç farklı veri merkezi ağ topolojisi olan Fat Tree, Z-Fat Tree (cross bağlantı desenli) ve RUFT-PL yapıları, paket iletimi başarısı, gecikme performansı ve bağlantı arızalarına karşı dayanıklılık açısından karşılaştırılmıştır. Gerçekleştirilen simülasyonlar sonucunda, tüm topolojilerin düşük arıza oranlarında yüksek başarı oranı sunduğu, ancak arıza oranı yükseldikçe bu başarı oranının topolojinin yapısal özelliklerine bağlı olarak farklılaştığı gözlemlenmiştir. Fat Tree topolojisi düzenli yapısıyla

*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696

dengeli bir performans sunarken, Z-Fat Tree özellikle belirli bağlantı desenleriyle gecikme açısından daha değişken sonuçlar üretmiştir. RUFT-PL topolojisi ise paralel bağlantıların sağladığı yedeklilik sayesinde hem düşük gecikme süreleri hem de yüksek arıza toleransı ile en iyi genel performansı göstermiştir.

Literatüre katkı açısından, bu çalışma üç farklı topolojiyi hem başarı oranı hem de gecikme dağılımı açısından karşılaştırmalı olarak analiz ederek, arıza toleransı perspektifinden farklarını sistematik biçimde ortaya koymaktadır. Önceki çalışmaların çoğunda bu tür karşılaştırmalar yalnızca bant genişliği veya teorik kapasite üzerinden yapılırken, bu çalışma bağlantı arızaları gibi gerçek dünya koşullarını simüle ederek daha kapsamlı bir değerlendirme sunmaktadır.

Elde edilen bulgular, ağ topolojisi seçiminde yalnızca nominal performans değil, aynı zamanda beklenmedik bağlantı kayıplarına karşı sistemin tepkisinin de dikkate alınması gerektiğini ortaya koymaktadır. Kritik uygulamalar ve servis sürekliliği gerektiren sistemlerde, RUFT-PL gibi yüksek esneklik ve hata toleransı sunan yapılara yönelmek, veri iletiminin güvenliğini ve sistemin sürdürülebilirliğini artıracaktır. Gelecek çalışmalarda, farklı anahtar port değerlerinin (örn. 12, 16, 20) performansa etkisi incelenerek ölçeklenebilirlik analizleri yapılabilir. Ayrıca, bu çalışmanın kapsamını genişletmek amacıyla farklı topolojilerin de deneysel analizlere dâhil edilmesi planlanmakta; böylece daha geniş bir ağ mimarisi havuzunda performans, gecikme ve hata toleransı kriterlerinin nasıl değiştiği sistematik olarak değerlendirilebilecektir.

6. Kaynaklar (References)

[1] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, I. Brandic, Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, Future Gener. Comput. Syst. 25 (6) (2009) 599–616.

[2] P. Kumari, P. Kaur, A survey of fault tolerance in cloud computing, J. King Saud Univ. Comput. Inf. Sci. 33 (10) (2021) 1159–1176.

[3] P.M. Mell, T. Grance, The NIST definition of cloud computing, Recommendations of the National Institute of Standards and Technology (2011).

[4] S. Patidar, D. Rane, P. Jain, A survey paper on cloud computing, Proc. 2012 2nd Int. Conf. Adv. Comput. Commun. Technol. (ACCT) (2012) 394–398.

[5] B. Varghese, R. Buyya, Next Generation Cloud Computing: New Trends and Research Directions, arXiv preprint arXiv:1707.07452 (2017). Available from: http://arxiv.org/abs/1707.07452.

[6] M.K. Gokhroo, M.C. Govil, E.S. Pilli, Detecting and Mitigating Faults in Cloud Computing Environment, Proc. 3rd Int. Conf. Comput. Intell. Commun. Technol. (CICT), IEEE (2017).

[7] L.P. Saikia, Y.L. Devi, Fault Tolerance Techniques and Algorithms in Cloud Computing, Int. J. Comput. Sci. Commun. Netw. 4 (1) (2014) 1–8.

[8] Z. Amin, N. Sethi, H. Singh, Review on Fault Tolerance Techniques in Cloud Computing, Int. J. Comput. Appl. 116 (18) (2015).

[9] M. Nazari Cheraghlou, A. Khadem-Zadeh, M. Haghparast, A survey of fault tolerance architecture in cloud computing, J. Netw. Comput. Appl. 61 (2016) 81–92.

[10] Z. Amin, N. Sethi, H. Singh, Review on Fault Tolerance Techniques in Cloud Computing, Int. J. Comput. Appl. 116 (18) (2015).

[11] X. Gu, H. Wang, Online anomaly prediction for robust cluster systems, Proc. Int. Conf. Data Eng. (2009) 1000–1011.

[12] I. Odun-Ayo, R. Goddy-Worlu, V. Samuel, V. Geteloma, Cloud designs and deployment models: A systematic mapping study, BMC Res. Notes 12 (1) (2019).

*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr

ISSN: 2618-6055 / 9, (1), 58 – 69, 2025
DOI: 10.62301/usmtd.1703696

[13] R. Gohil, H. Patel, Comparative Analysis of Cloud Platform: Amazon Web Service, Microsoft Azure, and Google Cloud Provider: A Review, Proc. 15th Int. Conf. Comput. Commun. Netw. Technol. (ICCCNT) (2024).

[14] E.N. Ekwonwune, C.D. Anyiam, O.E. Osuagwu, Modelling Conceptual Framework for Private Cloud Infrastructure Deployment in the ICT Centre of Tertiary Institutions, Commun. Netw. 10 (3) (2018) 117–125.

[15] A. Marinos, G. Briscoe, Community Cloud Computing (2009).

[16] V. Khadilkar, M. Kantarcioglu, B. Thuraisingham, S. Mehrotra, Secure Data Processing in a Hybrid Cloud, arXiv preprint arXiv:1105.1982 (2011). Available from: http://arxiv.org/abs/1105.1982.

[17] S.H.H. Madni, M.S.A. Latiff, Y. Coulibaly, S.M. Abdulhamid, Resource scheduling for infrastructure as a service (IaaS) in cloud computing: Challenges and opportunities, J. Netw. Comput. Appl. 68 (2016) 173–200.

[18] O. Krancher, P. Luther, M. Jost, Key Affordances of Platform-as-a-Service, J. Manag. Inf. Syst. 35 (3) (2018) 776–812.

[19] P.K. Chouhan, F. Yao, S.Y. Yerima, S. Sezer, Software as a Service: Analyzing Security Issues (2014).

[20] M. Howard, Cloud Computing – Everything As A Service, arXiv preprint arXiv:2206.07094 (2022). Available from: http://arxiv.org/abs/2206.07094.

[21] B. Wang, Z. Qi, R. Ma, H. Guan, A.V. Vasilakos, A survey on data center networking for cloud computing, Comput. Netw. 91 (2015) 528–547.

[22] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano, A. Kanagala, J. Provost, J. Simmons, E. Tanda, J. Wanderer, U. Hölzle, S. Stuart, A. Vahdat, Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network, Proc. ACM SIGCOMM (2015) 183–197.

[23] M. Al-Fares, A. Loukissas, A. Vahdat, A scalable commodity data center network architecture, Comput. Commun. Rev., ACM (2008).

[24] X. Li, S. Kumar, RUFT-PL: A Fault-Tolerant Fat-Tree with Parallel Links, Proc. IEEE ICCCN (2013) 1–7.

[25] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, S. Lu, BCube: A high performance, server-centric network architecture for modular data centers, SIGCOMM Comput. Commun. Rev. 39 (4) (2009) 63–74.

[26] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, S. Lu, DCell: A scalable and fault-tolerant network structure for data centers, SIGCOMM Comput. Commun. Rev. 38 (4) (2008) 75–86.

[27] A. Singla, C.-Y. Hong, L. Popa, P.B. Godfrey, Jellyfish: Networking Data Centers Randomly (2012).

[28] B. Lebiednik, A. Mangal, N. Tiwari, A Survey and Evaluation of Data Center Network Topologies, arXiv preprint arXiv:1605.01701 (2016). Available from: http://arxiv.org/abs/1605.01701.

[29] H. Emesowum, A. Paraskelidis, M. Adda, Fault tolerance capability of cloud data center, Proc. ICCP (2017) 495–502. https://doi.org/10.1109/iccp.2017.8117053

[30] H. Emesowum, A. Paraskelidis, M. Adda, Management of fault tolerance and traffic congestion in cloud data center, Proc. ICOIACT (2018) 10–15. https://doi.org/10.1109/icoiact.2018.8350692

[31] D.F. Bermúdez Garzón, C.G. Requena, M.E. Gómez, P. López, J. Duato, A family of fault-tolerant efficient indirect topologies, IEEE Trans. Parallel Distrib. Syst. 27 (4) (2016) 927–940. https://doi.org/10.1109/TPDS.2015.2430863

[32] J. Calle-Cancho, C. Cruz-Carrasco, D. Cortés-Polo, J. Galeano-Brajones, J. Carmona-Murillo, Enhancing Programmability in Next-Generation Networks: An Innovative Simulation Approach, Electron. (Switzerland) 13 (3) (2024).

*Sorumlu Yazar/Corresponding Author: huseyingokmen@sdu.edu.tr