
GAZİ ÜNİVERSİTESİ
Gazi
1926

BULUTTA ETKİN SANAL MAKİNE GÖÇÜ İÇİN YÖNTEMLER GELİŞTİRME

Yasemin DEMİRAL

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HAZİRAN 2019

Yasemin DEMİRAL tarafından hazırlanan "BULUTTA ETKİN SANAL MAKİNE GÖÇÜ İÇİN YÖNTEMLER GELİŞTİRME" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilgisayar Mühendisliği Ana Bilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Dr. Öğr. Üyesi Mehmet DEMİRCİ

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ..……

Başkan: Prof. Dr. Suat ÖZDEMİR

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……...

Üye: Doç. Dr. Ertan ONUR

Bilgisayar Mühendisliği Ana Bilim Dalı, Orta Doğu Teknik Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum**.** ……...

Tez Savunma Tarihi: 25/06/2019

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

…….…….

Prof. Dr. Sena YAŞYERLİ

Fen Bilimleri Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Yasemin DEMİRAL

25/06/2019

BULUTTA ETKİN SANAL MAKİNE GÖÇÜ İÇİN YÖNTEMLER GELİŞTİRME

(Yüksek Lisans Tezi)

Yasemin DEMİRAL

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Haziran 2019

ÖZET

Günümüzde artan teknoloji kullanımı ile bilişim alanında kullanılan fiziksel kaynaklar ve alt yapılar yetersiz kalmaya başlamış ve sanallaştırma kavramı ortaya çıkmıştır. Sanallaştırma sayesinde sunucu ve ağ kaynakları mantıksal parçalara bölünerek daha verimli kullanılabilmektedir. Sanallaştırmadan yoğun olarak faydalanan bulut bilişim, bilgi sistemlerinin daha esnek ve maliyet açısından etkin bir hale gelmesini sağlamıştır. Buluttaki işlerin, üzerinde çalıştırıldığı sanal makinelerin çeşitli nedenlerle bir fiziksel sunucudan diğerine taşınması gerekebilmektedir. Bu taşıma işlemine literatürde sanal makine göçü adı verilmiştir. Sanal makine göçü sırasında hizmet kesintisi, veri kaybı problemleri meydana gelmektedir. Bu tez çalışmasında bulut ortamındaki sanal makinelerin ve sanal ağların etkin göç ettirilmesi üzerine çalışmalar yapılmıştır. Bulut sistemlerinde kullanılan sanal makinelerin ve sanal ağların göç işleminin daha yüksek performanslı olmasına olanak verecek sezgisel algoritma tabanlı yeni bir yöntem geliştirilmiştir. Sanal yapıların sahip olduğu öncelik ve önem katsayısının sezgisel algoritmaya eklenmesi ile sanal makine göçü sırasındaki toplam göç zamanın ve yaşanan kesintilerin en aza indirgenmesi üzerine çalışılmıştır. Aynı zamanda proje öncelikleri göz önünde bulundurularak hizmet seviyesi anlaşmasının ihlali de engellenmiştir. Çalışma kapsamında göç ettirilen sanal makine sayısına ve sistemlerin büyüklüğüne bağlı olarak göç maliyetinde %3-16 aralığında artan iyileştirmeler gözlemlenmiştir.

Bilim Kodu	:	92413
Anahtar Kelimeler	:	Bulut bilişim, sanal makine göçü, sanal ağ göçü
Sayfa Adedi	:	77
Danışman	:	Dr. Öğr. Üyesi Mehmet DEMİRCİ

DEVELOPING METHODS FOR EFFICIENT VIRTUAL MACHINE MIGRATION IN THE CLOUD

(M. Sc. Thesis)

Yasemin DEMİRAL

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

June 2019

ABSTRACT

Today, with the increasing use of technology, physical resources and infrastructures used in information technologies have become inadequate and so the concept of virtualization has emerged. Server and network resources can be divided into logical parts and used more efficiently with virtualization technologies. Cloud computing which benefits from advantages of virtualization has made information systems more flexible and cost-effective. Virtual machines which run workloads in the cloud may need to migrate from one physical server to another for a variety of reasons. This migration process is called virtual machine migration in the literature. During the virtual machine migration, service interruptions and data loss problems can occur. In this thesis study, we work on the efficient migration of virtual machines and virtual networks in the cloud environment. A new heuristic algorithm-based method has been developed which provide the migration of virtual machines and virtual networks used in cloud systems to be more efficient. It has been aimed to minimize the total migration time and disruption during virtual machine migration by adding priority and importance coefficients of virtual structures into the heuristic algorithm. Breaches of the service level agreement have been prevented by taking into account the project priorities. Within the scope of the study, depending on the number of virtual machines migrated and the size of the systems, improvements in the migration cost were observed between 3-16%.

Science Code	:	92413
Key Words	:	Cloud computing, virtual machine migration, virtual network migration
Page Number	:	77
Supervisor	:	Dr. Lecturer Mehmet DEMİRCİ

TEŞEKKÜR

Tüm hayatım boyunca desteklerini eksik etmeyen annem ve babama, fikirleri, yardımları ve destekleri için yakın dostlarıma, çalıştığım kurum olan TÜBİTAK BİLGEM bünyesinde yer alan Bulut Bilişim ve Büyük Veri laboratuvarına ve yönlendirmeleri ile destek olan danışmanım Dr. Öğretim Üyesi Mehmet DEMİRCİ'ye teşekkürlerimi sunarım.

İÇİNDEKİLER

Sayfa

ÖZET iv

ABSTRACT v

TEŞEKKÜR vi

İÇİNDEKİLER vii

ÇİZELGELERİN LİSTESİ x

ŞEKİLLERİN LİSTESİ xi

RESİMLERİN LİSTESİ xii

SİMGELER VE KISALTMALAR xiii

1. GİRİŞ 1

2. ÖN BİLGİ 3

2.1. Sanallaştırma 3

2.2. Hipervizor 3

2.3. Ağ Sanallaştırma 4

2.4. Yazılım Tanımlı Ağlar 5

2.4.1. Yazılım tanımlı ağ mimarisi 7

2.4.2. Yazılım tanımlı ağlar ile çalışan araçlar 8

2.4.3. Yazılım tanımlı ağ kullanım avantajları 10

2.5. Bulut Bilişim ve Veri Merkezleri 11

2.5.1. Bulut bilişim Modelleri 12

2.5.2. Bulut hizmet modelleri 13

2.6. Sanal ortamlarda göç 17

2.6.1. Sanal makine göçü 18

2.6.2. Sanal makine göçü zorlukları 21

Sayfa

2.6.3. Sanal ağ göçü 22

3. LİTERATÜR TARAMASI 29

3.1. Sanallaştırma ve Bulut Bilişim 29

3.2. Sanal Makine Yerleşimi 30

3.3. Sanal Makine Göçü 31

3.4. Etkili Sanal Makine Göçünün Planlaması 33

3.5. Sanal Ağ Göçü 33

3.6. Sunucu Optimizasyonu 34

4. GEREÇ VE YÖNTEM 39

4.1. Problemin Tanımı 39

4.2. Kullanılan Algoritmalar 40

4.2.1. Minimum Spanning Tree 40

4.2.2. Maximal Independet Set 40

4.2.3. Dijkstra Algoritması 40

4.2.4. A* Algoritması 40

4.2.5. Özelleştirilmiş A* 41

4.3. Kullanılan Performans Metrikleri 41

4.4. Maliyet Fonksiyonu 43

4.5. Canlı Ortamda Sanal Makine Göçü 47

5. DENEYSEL DEĞERLENDİRME 51

5.1. Performans Ölçüm Metrikleri 51

5.2. Sanal Ağ Performansında Hipervizör Etkisinin İncelenmesi 51

5.3. Küçük Sanal Yapılarda Göç 52

5.4. Veri Merkezi Kapsamındaki Sanal Yapılarda Göç 54

5.5. Deney Sonuçlarının Karşılaştırılması 57

Sayfa

6. SONUÇ VE ÖNERİLER..... 65

KAYNAKLAR 67

ÖZGEÇMİŞ 77

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

Çizelge 2.1. OpenStack servisleri 16
Çizelge 3.1. Göç örnekleri sınıflandırması 32
Çizelge 4.1. Literatürde kullanılan sanal makine göç metrikleri 42
Çizelge 5.1. Sanal makinelerin özellikleri 51
Çizelge 5.2. Düğüm sayılarına göre öncelik dereceleri 58
Çizelge 5.3. Kullanılan algoritmalara göre test sonuçları 60

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil 2.1. Sanallaştırma katmanları 3
Şekil 2.2. Ağ sanallaştırma mimarisi 5
Şekil 2.3. Yazılım tanımlı ağlar 6
Şekil 2.4. Bulut bilişim modelleri 12
Şekil 2.5. Bulut hizmet modelleri 14
Şekil 2.6. OpenStack ortamında sanal makine ve ağ yapısı 17
Şekil 2.7. Sanal makine göç nedenleri ve sunucu optimizasyonu 18
Şekil 2.8. Sanal makine göçü zorlukları 21
Şekil 2.9. Kullanılan araçların özellikleri 22
Şekil 2.10. Sanal ağ yapısındaki düğümlerin ilk hali 23
Şekil 2.11. Belirlenen düğümde tünelin kurulması 24
Şekil 2.12. Geçici bağlantıların kurulması 25
Şekil 2.13. A düğümünün taşınması 25
Şekil 2.14. Komşu düğümler ile geçici bağlantıların kurulması 26
Şekil 2.15. C tünel bağlantısı ve geçici bağlantılar 26
Şekil 2.16. Komşu düğümler ile kalıcı linklerin oluşturulması 27
Şekil 2.17. Ağın tamamen taşınması 27
Şekil 2.18. Yazılım tanımlı ağlarda sanal ağ göçü ve veri akışı 28
Şekil 4.1. Bulut sistemleri arası göç 39
Şekil 4.2. Sanal ağ göçünde zaman gösterimi 43
Şekil 5.1. Etkin sanal makine göçü için yöntem geliştirilmesi 57
Şekil 5.2. Deney sonuçları 61
Şekil 5.3. Elde edilen iyileştirme grafiği 62
Şekil 6.1. Etkin sanal makine göçü için etkin yöntemler 65

RESİMLERİN LİSTESİ

Resim **Sayfa**

Resim 2.1. sFlow-RT ağ haritasının arayüzü 9
Resim 2.2. OpenDayLight topoloji ekranı 10
Resim 4.1. Projelere ait ağ topolojisi 46
Resim 4.2. Sanal makinelerin dağılımı 47
Resim 4.3. Sanal makine göçü sistem yöneticisi ekranı 48
Resim 4.4. Canlı göç sonucunda sanal makinelerin fiziksel sunucu bilgileri 49
Resim 5.1. Deneysel az düğümlü ağ topolojisi (OpenDayLight) 52
Resim 5.2. Deneysel çok düğümlü ağ topolojisi 53
Resim 5.5. Omurga-uç yapısı 54
Resim 5.6. Deneysel veri merkezi yapısı 55

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
K	Kilobit
M	Megabit
MBytes	Megabytes
s	Saniye

Kısaltmalar	**Açıklamalar**
API	Application Programming Interface
CPU	Central Processing Unit
HSA	Hizmet Seviyesi Anlaşması
KVM	Kernel-based Virtual Machine
MIS	Maximal Independent Set
MST	Minimum Spanning Tree
RAM	Random Access Memory

1. GİRİŞ

Günümüzde teknolojinin yaygınlaşmasıyla bilişim teknolojileri hayatımızda büyük bir öneme sahip olmuştur. Bilişim hizmetlerinin sunulmasında ve kurulmasında en kritik noktalardan birisi alt yapı kaynaklarıdır. Bu kaynaklar; ağ kabloları, ağ cihazları, sunucular, depolama alanları olarak sıralanabilir.

Artan teknoloji kullanımı ile alt yapı kaynakları yetersiz kalmaktadır. Ayrıca performans kayıpları yaşanmakta ve maliyet artmaktadır. Yetersiz kalan donanımlara yeni donanımlar eklemek bir çözüm olabilmektedir. Ancak maliyeti çok yüksek olmakla birlikte istenilen çözümü sunmamakta ve yönetilmesi zor karmaşık yapılar meydana getirmektedir. Bu yüzden fiziksel olarak kaynak eklemek yerine kaynakların sanallaştırılması konusu gündeme gelmiş ve sanallaştırma kavramı ortaya çıkmıştır. Sanallaştırma ile yönetilmesi kolay bir yapı oluşturulmasının yanı sıra fiziksel kaynaklar birden fazla mantıksal kaynaklara bölünerek daha verimli bir sistem oluşturulabilmektedir. Yüksek performanslı, maliyeti düşük ve yönetilmesi kolay bir yapı sunan sanallaştırma teknolojileri; fiziksel sunucularda, konteynerlerde, ağ yapılarında ve depolama alanlarında kullanılabilmektedir.

Alt yapı kaynaklarının sanallaştırılması mevcut sistemlerin yüksek performanslı çalışmasını gerektirmektedir. Ayrıca sürekli gelişen teknoloji ile kullanılan fiziksel kaynaklar güncellenebilmektedir. Fiziksel kaynakların bakımı, onarımı, güncellenmesi, güvenliği, performansı gibi konulardan dolayı yapılan sanallaştırılmış kaynakların göç ettirilmesi gündeme gelmektedir. Göç kavramı; kaynakların bir yerden başka bir yere geçişi olarak tanımlanabilmektedir. Ancak bir yerden başka bir yere göç maliyetli ve göç esnasında kabul edilemez performans düşüşlerine sebep olmakta, güvenlik sorunlarına yol açabilmektedir.

Araştırmanın amacı, sanallaştırılmış yapılardan birisi olan sanal makinelerde ve sanal ağ yapılarında göç sırasında yaşanacak performans kayıplarını en aza indirmek, sistemlere erişilebilirlik oranın yüksek düzeyde tutulmasını sağlamaktır. Ayrıca ortaya çıkacak göç maliyetini en aza indirgemek ve kalite ölçütlerini yerine getirmektedir. Belirlenen performans kriterleri kullanılan ortamlara, teknolojilere ve yazılımlara göre değişiklik göstermektedir.

Bu tez çalışmasında, bulut ortamlarında sanal makine ve sanal ağ göçü gibi sanal kaynak göçlerinden sistemin nasıl etkilendiği ve fiziksel katmanda ortaya çıkan maliyet araştırılmıştır. Bulut sağlayıcılarının karşılaştığı performans sorunları ve ortaya çıkan maliyetler hesaplanmıştır. Sanal makinelerin göç işlemi sırasında sistemin sahip olduğu ağ topolojisi göz önünde bulundurulmuştur. Aynı zamanda bulut sağlayıcılarından hizmet alan kurum, kuruluş, kişi veya projelerin sahip olduğu öncelik derecesi hesaplara katılmıştır.

Tez çalışmasında, en kısa yolu veren Dijkstra algoritmasının yanı sıra sezgisel olarak çalışan ve en kısa yolu veren A* algoritması, Minimum Spanning Tree ve Maximal Independet Set algoritmaları kullanılmıştır. Kullanıcılardan alının bilgiler ve bulut sistem yöneticileri tarafından belirlenen sanal makinelerin öncelik katsayıları dikkate alınarak A* algoritmasının sezgisel fonksiyonu düzenlenmiş olup Python NetworkX kütüphanesi kullanılarak simülasyon ortamında testler gerçekleştirilmiştir.

Testler 5, 10, 25, 50, 75 ve 100 düğüme sahip olan yapılarda gerçekleştirilmiştir. Test sonuçları Dijkstra, MST ve MIS algoritmaları ve sezgisel fonksiyonu makinelerin öncelik katsayısını gözetecek şekilde düzenlenen A* algoritması ile karşılaştırılmıştır. Ortaya çıkan maliyetlerin sistemde göç ettirilecek sanal makine sayısına bağlı olarak sezgisel A* algoritması ile %3-16 arasında daha etkin sonuçlar gözlenmiş görülmüştür.

Tez çalışmasının 1. Bölümünde giriş kısmı yer almaktadır. 2. Bölümünde ön bilgi bulunmakta olup sanallaştırma, hipervizör, ağ sanallaştırma, yazılım tanımlı ağlar, bulut bilişim ve veri merkezleri ayrıca sanal ortamlarda göç konuları yer almaktadır. 3. Bölümde literatürde yapılan çalışmalar bulunmakta olup sanallaştırma ve bulut bilişim, sanal makine yerleşimi, sanal makine göçü, etkili sanal makine göçü planlanması, sanal ağ göçü ve sunucu optimizasyonu başlıkları altında sınıflandırılmıştır. 4. Bölümde tez çalışmasındaki gereç ve yöntemlerden bahsedilerek tezin çerçevesi çizilerek problem tanımı yapılmıştır. Aynı zamanda kullanılan performans metriklerinden ve maliyet fonksiyonundan bahsedilmektedir. 5. Bölümde ise deneysel değerlendirme yer almakta olup kullanılan ölçüm metrikleri, sanal ağ performansında hipervizör etkisinin incelenmesi, küçük sanal ağ yapılarında göç, veri merkezi kapsamındaki sanal yapılarda göç yer almakta olup elde edilen deney sonuçlarının karşılaştırması bulunmaktadır. 6. Bölümde ise sonuç ve öneriler bulunmaktadır.

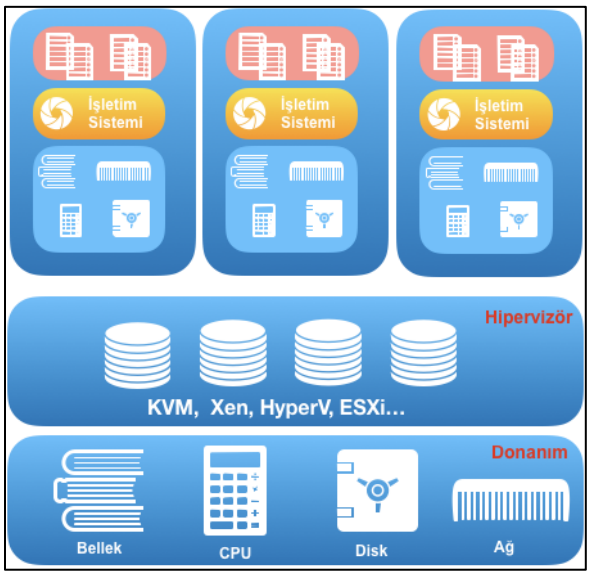
2. ÖN BİLGİ

2.1. Sanallaştırma

Sanallaştırma; fiziksel kaynakların yani CPU, bellek, ağ yapıları ve depolama alanları gibi sistemlerin mantıksal bir katman ile ayrılarak birden fazla sanal alan, düğüme veyahut makine olarak çalışmasına olanak veren, kaynakların daha performanslı, yönetilmesi kolay hale getiren bir teknoloji çözümüdür [1]. Sunucu sanallaştırma olarak değerlendirilen fiziksel sunucularından bir veya daha fazlasının kaynakların mantıksal olarak optimize edilerek kullanılması sunucu sanallaştırmadır. Fiziksel sunucuların özellikleri sanallaştırılarak yönetimi kolay ve birbirinden izole yapılar halinde çok çeşitli kullanım olanakları sağlanabilir [2].

2.2. Hipervizor

Fiziksel katmanların üzerine hipervizör olarak adlandırılan bir yazılımsal bir katman ile sanallaştırma gerçekleştirilmektedir. Sanallaştırılan sistemler üzerine farklı işletim sistemine ait sanal makineler kurulabilmektedir. Ayrıca çoklu işletim sistemlerinin aynı donanım üzerinde çalışmasına olanak sağlayan bir kod parçası olarak tanımlanabilmektedir. Sanal makinelerin üzerinde bulunduğu katmandır [3].



Şekil 2.1. Sanallaştırma katmanları

Şekil 2.1'de gösterildiği gibi fiziksel donanımların üzerinde bulunan ve sanallaştırma katmanı olarak görev yapan hipervizör katmanı olarak farklı yazılımlar kullanılabilmektedir. Bunlardan en sık kullanılanı çekirdek tabanlı sanal makine oluşturmaya olanak veren Linux çekirdeğini kullanarak bir üst sisteme geçiren KVM'dir [4]. Qemu kendi başına bağımsız bir yazılım olmakla birlikte sanal makineleri taklit prensibiyle çalışır. Temel olarak, belirli bir işlemci için yazılmış ikili kodu başka bir taneye dönüştüren özel bir yeniden derleyici ile çalışmaktadır [5].

VMware ESX sunucu sanallaştırma; VMware tarafından geliştirilen Unix temelli bir işletim sistemi olup ortam üzerinde sanal makinelerin ve sanal ağların kurulumuna izin vermektedir [6-7]. Hyper-V; Viridian kod adlı ve daha önce Windows Server Virtualization olarak bilinmekle beraber yerel bir hipervizördür. Windows çalıştırılan x86_64 sistemlerde sanal makineler ve sanal ağ ortamlarının oluşumuna olanak vermektedir [8].

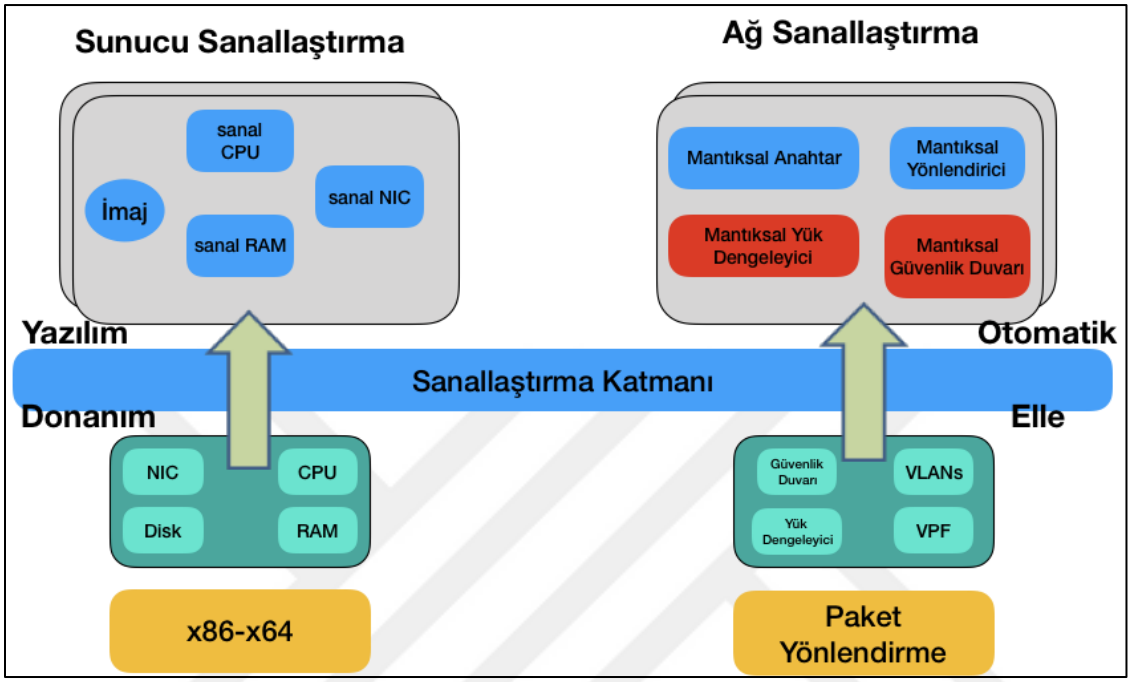
Xen; birçok sanallaştırma teknolojisinde olduğu gibi Xen Project isimli proje de açık kaynaklı olarak geliştirilmektedir. Xen ile sunucu sanallaştırma, hizmet olarak alt yapı tahsisi yapılabilmektedir. Aynı zamanda masaüstü sanallaştırma, güvenlik uygulamaları, gömülü sistemler ve donanım cihazları, otomotiv, havacılık dahil olmak üzere bir dizi ticari ve açık kaynak uygulamasının sanallaştırılmasına olanak vermektedir [9].

Özellikle sunucu sanallaştırma için kullanılan hipervizörlerin sanal makine göçüne etkisi incelenmiş olup Sinyal İşleme ve İletişim Uygulamaları Kurultayı 2018'de "Sanal Ağ Performansında Hipervizor Etkisinin İncelenmesi" [10] başlıklı bildiri ile sunumu gerçekleştirilmiştir. Yapılan çalışan 5. Bölümde ayrıntılı olarak anlatılmaktadır.

2.3. Ağ Sanallaştırma

Ağ sanallaştırmada, ağ bileşenlerinin tıpkı fizikselmiş gibi çalışmasına olanak verecek şekilde yazılımlar kullanılmaktadır. Şekil 2.2'deki gibi var olan donanımların üzerine bir sanallaştırma katmanı eklenerek parçalı ve yönetilmesi kolay sanal yapılar elde edilmektedir. Ağ sanallaştırma ile veri ve ağ paketlerinin aktarım hızları gözlemenebilir, yönetilebilir olmakta ayrıca ölçeklenebilirliğin, güvenilirliğin, esnekliğin ve güvenliğin ağ optimizasyonuna izin vermesi amaçlanmıştır. Ağ yönetim görevleri sanallaştırma ile otomatikleştirilebilmektedir. Ağ sanallaştırma, özellikle büyük, hızlı ve öngörülemeyen

kullanım artışları yaşayan ağlar için yararlıdır. Tek başına çalışan düşük ve yüksek seviyeli ağ fonksiyonlarını yerine getiren donanımların (örneğin güvenlik duvarı, yük dengeleyici) sanal bir ağ fonksiyonu hizmetine dönüştürülmesidir [11].

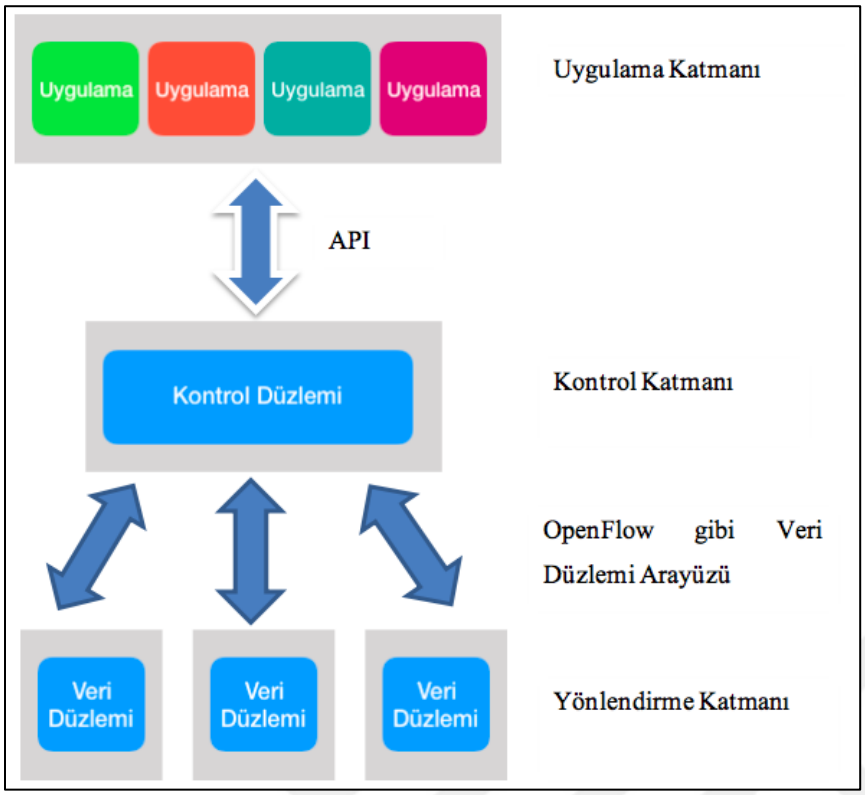


Şekil 2.2. Ağ sanallaştırma mimarisi

2.4. Yazılım Tanımlı Ağlar

Artan kesintisiz internet ihtiyacı ve yeni gelişen bulut bilişim, mobil ağlar gibi teknolojilerle ağ yapıları daha fazla önem kazanmıştır. Ağ yapılarındaki gelişmeler ile kullanılan fiziksel yapıların sayısı artmıştır. Karmaşıklaşan yapıların yanı sıra bir çok şirket ortaya çıkmış ve sağlayıcı bağımlılığı ortaya çıkmıştır. Yönetilmesi çok zor bir hal ağ yapılarında, yazılımsal olarak çözümler sunulmaya başlanmış ve yazılım tanımlı ağlar kavramı ortaya çıkmıştır [12].

Yazılım tanımlı ağlar karşımıza ilk önce akademi de çıkmakla birlikte sanayide de çok fazla kullanılmaya başlamıştır. Geleneksel ağ mimarilerinden farklı olarak Şekil 2.3'deki gibi 2 temel karakteristik özelliğe sahiptir. Bunlardan biri; klasik ağ mimarilerinde bütünleşik olarak var olan kontrol ve veri düzleminin birbirinden ayrılmasıdır. Diğeri ise ağ uygulamasının geliştirilmesi için programlanabilir bir yapı sunması, yazılımsal olarak yönetilmesine izin vermesidir [13].



Şekil 2.3. Yazılım tanımlı ağlar

Open Network Foundation tarafından doğrudan programlanabilen ve ağ kontrolünün iletimden ayrıştırıldığı bir yapıyla ortaya çıkan ağ mimarisi olarak belirtilmektedir. Temel olarak kontrol düzlemini veri düzleminden ayırmak hedeflenmektedir. İstemci ile veri tabanı arasındaki iletişimin en ucuz ve en güvenilir yoldan sağlanması ve yönetilmesi amaçlanmaktadır.

OpenFlow protokolü, bu amaca yönelik olarak geliştirilmiş olup üniversiteler ve araştırma enstitüleri gibi kurumlarda, ağ yönetiminin daha çok araştırmaya imkân sağlanmaktadır. OpenFlow yenilikçi ağ protokollerini ve yeni yöntemlerin kullanımdaki ağlarda geliştirilmesini sağlayan bir açık standart olarak tanımlanmaktadır. OpenFlow, Stanford ve California Berkeley Üniversiteleri tarafından internet ağındaki switch, router ve bazı network cihazlarının donanımsal kısıtlamalarından dolayı zamanla tıkanacağı düşünülen ve artık gerçek zamanlı isteklere arzu edildiği gibi cevap veremeyen küresel internet ağı için tüm bir ağa ve ağ cihazlarına yazılımlar yoluyla daha etkin müdahele edilebilmesi amacıyla ortaya çıkmış bir açık kaynak ağ protokolüdür. OpenFlow protokolü ve POX denetleyicisi kullanarak, orta büyüklükte ağlarda, bu teknolojisinin kullanılabilirliği hakkında niceliksel bir bilgi sağlamak, araştırmacı ve kurumların ağ sistemlerinde kullanabilecekleri bileşenler

açısından önemlidir. OpenFlow protokolü ile birlikte ağ donanımın kontrol bölümünün dışarıda bir sunucuya aktarılması, ağ yöneticilerine büyük esneklik sağlamaktadır. Bu esneklik büyük alanlardaki ağların yönetimlerini, günümüzün bir diğer sorunu olan enerji ihtiyaçları da düşünülerek, donanım ve alt yapıların daha verimli kullanılmasına yardımcı olmaktadır [14]. Akademinin yanı sıra ticari olarak da günümüzde yazılım tanımlı ağlar büyük bir yere sahiptir [15].

2.4.1. Yazılım tanımlı ağ mimarisi

Bir ağ cihazı kontrol katmanı, veri katmanı ve yönetim katmanı olmak üzere 3 bölümden oluşur. Kontrol katmanı; cihazı üzerinde çalışan protokollerin belirlendiği kısımdır. Bir paketin nereye gideceğine karar verir. Veri katmanı; kontrol katmanından alınan bilgiye göre verinin iletildiği kısımdır. Yönetim katmanı ise cihazın yönetimsel işlemlerinin gerçekleştiği kısımdır. Bu katmanlar ağ cihazlarının içerisinde yer almaktadır. Bir paketin iletiminden, nereye gideceğinden, bütün işlemlerden sorumludur. Yazılım tanımlı ağlar ile bu katmanlar birbirinden kolaylıkla ayrılabilmekte, programlanabilmekte ve yönetimi kolay hale getirilmektedir.

Yazılım tanımlı ağ mimarisi katmanlı bir yapıdan oluşmaktadır. Bunlar uygulama katmanı, kontrol ve yönetim katmanı ve altyapı katmanı olarak sıralanabilir. Uygulama katmanı; ağ kontrolcüsünün yönetilmesi, görüntülenmesi, kontrolcüden veri alınması ve bu veriler doğrultusunda istatistik çıkarabilmesini sağlayan bir takım uygulamaların bulunduğu katmandır. Kontrol ve yönetim katmanı; aynı katmanın 2 farklı bileşeni olarak adlandırabilir. Kontrolcü bu katmanda işlem görmektedir. Kontrolcüye komutların verildiği, çalıştırıldığı, programlama yapılan kısımdır. Altyapı katmanı; veri iletimini yapan fiziksel donanımların olduğu katmandır. Veri ve akış yönlendirmeleri bu katmanda yapılır. Bu şekilde ayrılan katmanlar ile kontrolcü ile tek bir merkezden yönetim sağlanabilir ve ağ akışındaki trafik tek bir merkezden görüntülenip yönetilebilir.

Yazılım tanımlı ağların kullanıldığı yapılarda bir sistemin çalışması ve bir paketin gönderilmesi sırasındaki akış aşağıdaki gibi sıralanabilir.

- Ağ yapılarında haberleşmek yada bir bilgi almak için istemci kendisine en yakın cihaz aracılığıyla istekte bulunur.

- Cihaz kendisinde mevcut tabloda herhangi bir aynı istek olup olmadığını kontrol eder. Eğer yoksa, istek alan cihaz direk olarak merkezi kontrolcü ile haberleşir, gelen trafik isteği karşısında nasıl davranmaları gerektiği kontrolcüye sorulur.
- Kontolcü tarafından yönlendirme tablosunda bir kural olup olmadığı kontrol edilir. Eğer mevcut bir akış kuralı var ise o davranış gönderilir. Bir kural yok ise tanımlanan kurallar çerçevesinde akış tablosuna yeni bir kural eklenerek yapılacak davranış kendisine sorgu yapan cihaza cevap olarak gönderilir.
- Belirlenen kurala bağlı olarak gelen paket istedğinin kullanacağı yollarla ilgi bilgi, kullanılacak cihazlara, kontrolcü tarafından iletilir.
- İstemci kontrolcünün yönlendirmesi ile istediği veri tabanına erişebilir, internete çıkabilir.

2.4.2. Yazılım tanımlı ağlar ile çalışan araçlar

Yazılım tanımlı ağ mimarisini kullarak çalışan çeşitli programlama ve görüntüleme araçları bulunmaktadır. Bu tez çalışması kapsamında Mininet, sFlow-RT ve OpenDayLight kullanılmıştır.

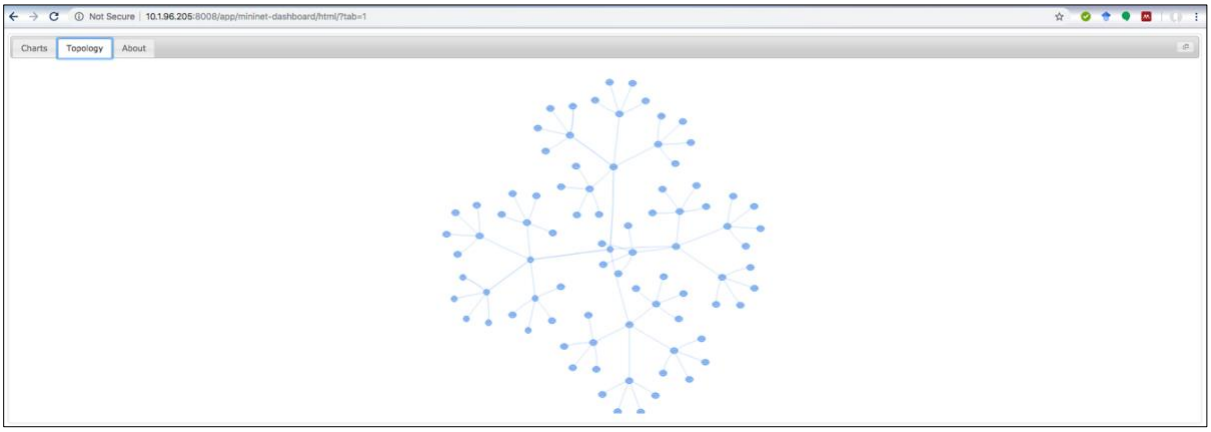
Mininet

Mininet; yazılım tanımlı ağların kurgulanmasına, oluşturulmasına, yönetilmesine ve test edilmesine olanak veren açık kaynak kodlu bir projedir [16]. Mininet ile gerçek bir sanal ağ oluşturularak gerçek bir çekirdek, anahtar ve uygulama kodu bir sanal makine içerisinde kolaylıkla ve hızlı bir şekilde çalıştırılabilmektedir. Mininet ile kontrolcüler, ağ anahtarları ve ağ uç noktaları test edilebilmektedir [17].

sFlow-RT

sFlow-RT ™, yazılım tanımlı ağlara, yönetim yığınlarına, gerçek zamanlı görünürlük sağlayan, yük dengeleme, dağıtık servis dışı bırakma saldırını azaltma ve iş yükü yerleştirme gibi yeni performans tanıyan uygulama sınıfları sağlayarak asekron analitik teknolojisi içeren bir yazılımdır [18].

Ağ görünürlüğü, bulut altyapısında çalışan ölçeklendirme hizmetleri arasındaki etkileşimi anlamak için önemlidir. Ağ, ana bilgisayar ve uygulama izlemeyi entegre bir süreç içerisine alarak sFlow-RT uygulamalardaki ağ trafiği izlenebilmektedir. Oluşturulan topolojinin ve düğümlerin konumu ağ haritası şeklinde Resim 2.1'deki gibi izlenebilmektedir.



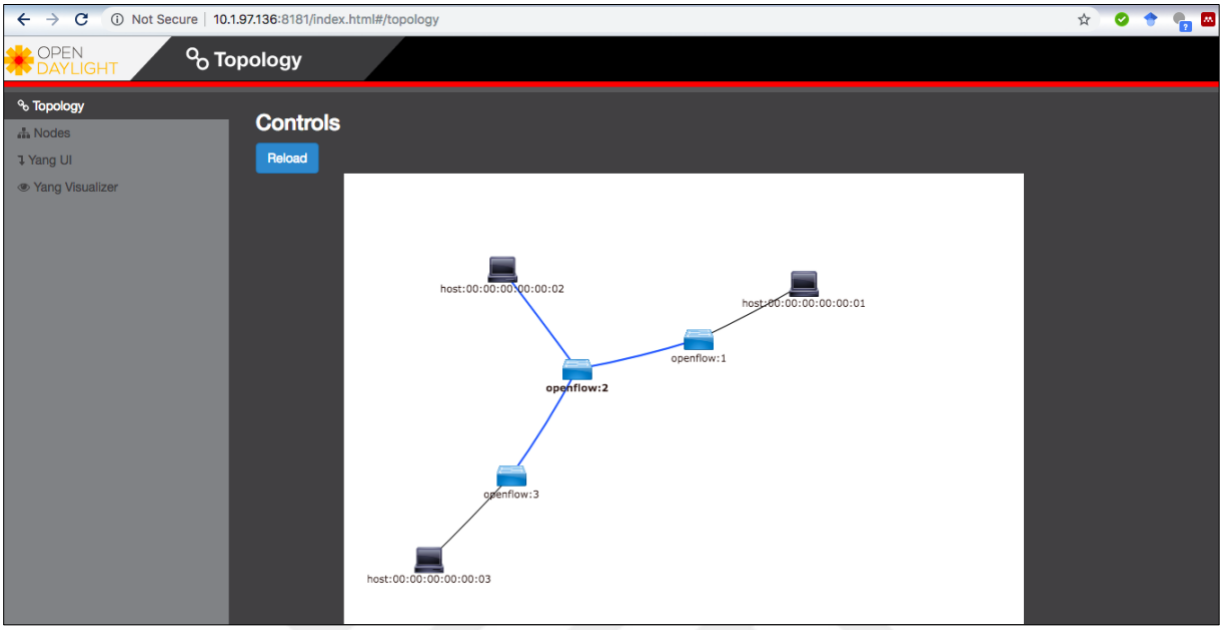
Resim 2.1. sFlow-RT ağ haritasının arayüzü

OpenDaylight

OpenDaylight, herhangi bir boyuttaki ve ölçekteki ağları özelleştirmek ayrıca otomatikleştirmek için modüler bir açık kaynak platformudur. OpenDayLight projesi yazılım tanımlı ağlarla birlikte çıkmış olup ağ programlanabilirliğinin odak noktası olarak geliştirilmiştir. Mevcut ağ ortamlarında çeşitli kullanım durumlarına yönelik ticari çözümler için tasarlanmıştır [19]. Kurgulanan ağ yapısının yönetilmesi, görselleştirilmesi konusunda kullanılmaktadır [20]. Bulut ortamları ile entegre olarak çalışabilen OpenDayLight; ağ yapılarının hızlandırılmasını ve performans artırılmasını sağlamaktadır. Resim 2.3'daki gibi istenilen ağ topolojisi görselleştirilebilmektedir. Topolojinin ve düğümlerin durumu izlenebilmektedir.

OpenDaylight projesi, yazılım tanımlı ağların benimsenmesini hızlandırmayı ve ağ fonksiyonlarının sanallaştırılması için sağlam bir altyapı kurmayı amaçlayan bir projedir. Ortak bir açık kaynak platformu olan OpenDaylight projesi, yenilikçi çalışmaların hız kazanması, risklerin azaltılmasını hedeflemektedir. Yazılım tanımlı ağlar çok daha şeffaf bir yaklaşım oluşturması için teknoloji dünyasının önemli kurumları olan Arista Networks, Big Switch Networks, Brocade, Cisco, Citrix, Ericsson, HP, IBM, Juniper Networks,

Microsoft, NEC, Nuage Networks, PLUMgrid, Red Hat ve VMware tarafından hayata geçirilmiş olup desteklenmeye devam etmektedir [21].



Resim 2.2. OpenDayLight topoloji ekranı

2.4.3. Yazılım tanımlı ağ kullanım avantajları

Yazılım tanımlı ağlar kullanılarak hem sistem yöneticileri, kullanıcılar, geliştiriciler ve hizmet alanlar açısından çeşitli avantajlarını sıralamak mümkündür. Bunlardan birisi daha az maliyet ile ağ yapısındaki cihazların daha verimli kullanılması sağlanır. Mevcut donanımlar daha verimli kullanılabilir, maliyet azaltılabilir. Tek merkezden yönetilebilir bir yapıya sahiptir. Kontrol ve veri düzleminin ayrı olması ile ihtiyaçların artması durumunda veri düzlemindeki yapı kolaylıkla genişletilebilir. Merkezi kontrolcü ile ağ yönetimi sağlanabilir. Dinamik, optimize olma özelliğine sahip bir yapı kolaylıkla oluşturulabilir. Merkezi yönetim olması sayesinde güvenlik duvarları kolaylıkla yönetilebilir, güvenlik kuralları eklenip çıkartılabilir, yetkilendirme ve kısıtlamalar kolaylıkla yönetilebilir. Trafiğin yoğun olduğu bölgeler görüntülenerek, gelen isteklerin trafiğin yoğun olmadığı yollara yönlendirilebilmesine olanak sağlar. Yüksek erişilebilirlik özelliği ile sistemdeki cihazlarda bir sorun olması durumunda, alternatif yollara yönlendirme yapılır, kontrolcü de bir sorun oluştuğundan ise sekronize çalışan ikinci kontrolcü işlemler yönetilmeye devam eder.

2.5. Bulut Bilişim ve Veri Merkezleri

Bulut bilişim, ağ üzerinden istenildiği zaman erişilmesine ve kullanılmasına izin veren kaynakların kullanıldığını kadar ödenmesi prensibine dayanan bir bilişim hizmetidir [22]. Temel olarak uzakta var olan ve hizmet sağlayıcısı tarafından işletilen, kullanıcıların internet üzerinden erişebildikleri ve sadece kullandıkları kadar kaynakların ücretinin ödemesi prensibi ile çalışan bir sistemdir. Bu sistemde kullanıcıların, şirketlerin ve akademisyenlerin projeleri için fiziksel olarak donanım satın almaları yerine maliyeti çok daha düşük bulut hizmeti kullanmaları, projelerini bulut ortamında gerçekleştirmeleri; yüksek performanslı, düşük maliyetli ve çevre dostu bir çözüm sunulmaktadır [23].

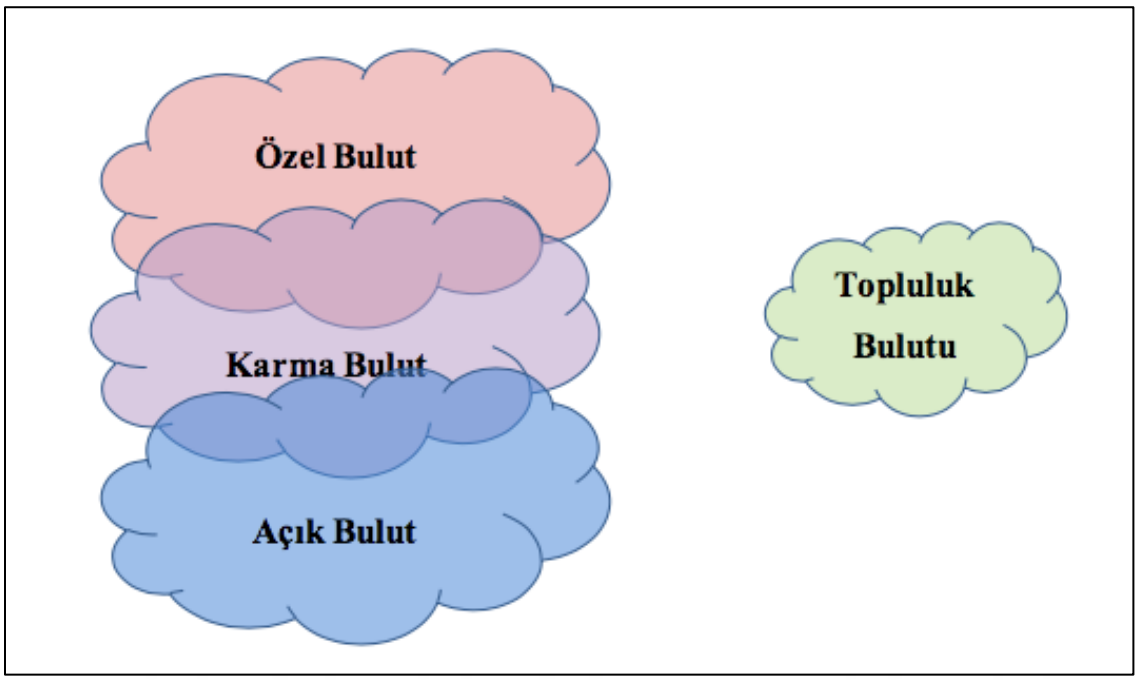
Veri merkezi, bilgisayar sistemleri ile verileri barındıran, sunucuların, ağ ve disk yapılarının bulunduğu yüksek kapasiteli donanım ağırlıklı merkezdir. Mevcut donanımların, yönetilmesi, dağıtılması ve etkin kullanılması veri merkezlerinin varlık amaçlarındandır. Fiziksel kaynakların etkin şekilde kullanılması, maliyeti düşük ve kaliteli hizmet verilmesi için veri merkezi yönetimleri önem taşımaktadır [24]. Bulut hizmetleri için servis sağlayıcıların bir veya daha fazla veri merkezine ihtiyaç olup, veri merkezinin performansı verilen bulut hizmet performansını ve kalitesini doğrudan etkilemektedir [25].

Bulut bilişim, kaynaklarının kullanımı sonucunda, varlık tabanlı fiziksel kaynaklar yerini hizmet tabanlı sanal kaynaklara bırakmaktadır. Gelişmeler sonucunda bulut bilişim kullanımı yaygın bir şekilde artarak devam etmektedir. Performansın önemli olduğu uygulamalar için hesaplama kaynaklarının ve bulut hesaplama alt yapısının verimli ve etkin bir şekilde kullanılarak uygulama çıktısının artırılması ve tepki süresinin azaltılması önem taşımaktadır [26].

Bulut bilişim performansını etkileyen başlıca faktörler; yüksek hacimli istemci-sunucu etkileşimleri, büyük veri yükleri, yetersiz bant genişliği, ağ tıkanıklığı ve sunucu darboğazları olarak sıralanabilir.

2.5.1. Bulut bilişim modelleri

Bulut bilişimde kullanılan veri merkezi yapısına göre ayrışan Şekil 2.4'deki gibi; açık bulut, özel bulut, karma bulut ve topluluk bulutu olmak üzere dört ayrı bulut modeli bulunmaktadır.



Şekil 2.4. Bulut bilişim modelleri

Açık Bulut (Public Cloud)

Açık bulut kamuya açılmış büyük ölçekli altyapıdır. Bulut altyapısı herkese ya da büyük endüstri grubuna açıktır. Bulut servisini veren bulutun sahibidir. Servisi veren yazılım ve depolama gibi kaynakları sağlar ve internet üzerinden kullancıların erişimine açar. Özellikle kişisel kullanımlar için uygun bir bulut bilişim çözümüdür. Örnek olarak; Amazon Elastic Compute Cloud (EC2), IBM BlueCloud, Sun Cloud, Google AppEngine ve Windows Azure Servis Platformu verilebilir.

Özel Bulut (Private Cloud)

Özel bulut kurumun kendi oluşturduğu ya da kiraladığı buluttur. Bulut altyapısı yalnızca firma için çalışır, kurumun kendisi tarafından ya da üçüncü parti bir firma tarafından yönetilir. Bu bulut yapısı daha büyük şirketler ve bilgileri daha önemli olan şirketlerin tercih

ettiği bir bulut teknolojisi tipidir. Buna örnek olarak Microsoft şirketinin sunduğu Hyper-V ve System Center ürünlerini gösterilebilir.

Karma Bulut (Hybrid Cloud)

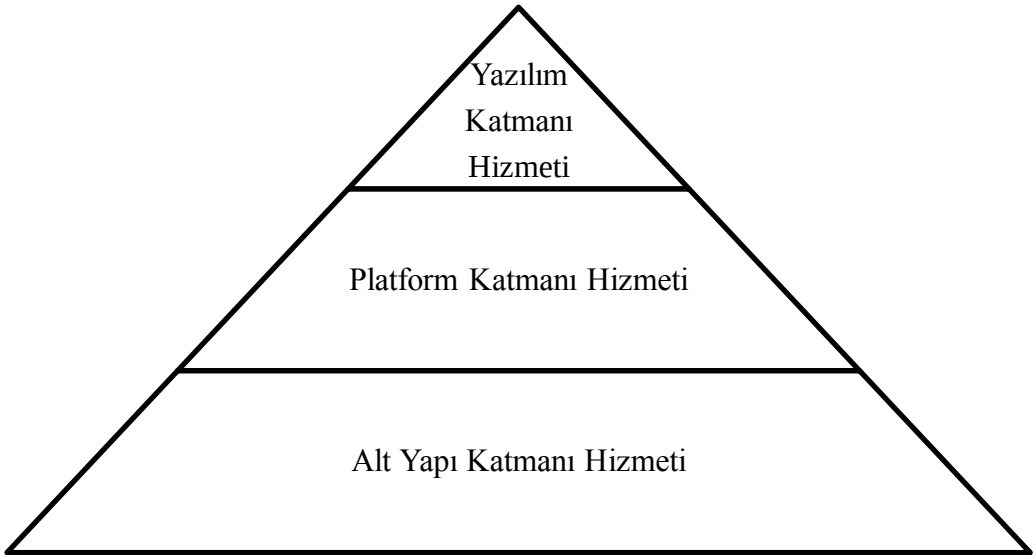
Karma bulut iki veya daha fazla bulut modelinin birlikte kullanılması ile oluşan yapıdır. Bulutlar kendi özelliklerini kaybetmeden yazılımın ve verinin taşınmasına izin verecek şekilde standardize edilmiş ya da özel teknoloji ile bağlanmıştır. Karma bulut; kamusal ve özel bulutun birleşiminden ortaya çıkan yapılardır. Güvenlik ve gizliliğin daha önemli olduğu ve tedbirin yüksek tutulması gereken yerlerde özel bulut teknolojisi kullanılır, güvenlik tedbirlerinin daha düşük düzeyde tutulabileceği alanlarda açık bulut kullanılabilir.

Topluluk Bulutu (Community Cloud)

Belirli bir toplulukla paylaşılan buluttur. Bulut altyapısı birkaç organizasyon ya da firma tarafından paylaşılır, böylece aynı amacı paylaşan, aynı güvenlik gereksinimleri olan, aynı tarzda idare edilen organizasyonlar, firmalar desteklenir. Organizasyon, firma ya da üçüncü parti tarafından yönetilir. Çok fazla tercih edilen bir bulut türü değildir.

2.5.2. Bulut hizmet modelleri

Bulut bilişim hizmet modelleri üç katmanda incelenmektedir. Bunlar; altyapı hizmet katmanı, platform katmanı ve yazılım katmanı olarak Şekil 2.5'deki gibi sıralanabilir.



Şekil 2.5. Bulut hizmet modelleri

Yazılım Katmanı Hizmet Modeli

Yazılım katmanı servisi, hazır paketlenmiş ve bulutta çalışabilir olarak geliştirilmiş bir yazılımın son kullanıcılara bulut üzerinden çalıştırılarak sunulmasıdır. İhtiyaç duyulan yazılım hizmetinin bulut üzerinden sağlandığı ve kullanıcının çeşitli cihazlardan erişim olanağı sunarak yazılımı kullanabildiği hizmet servisidir. Kullanıcılar sadece kendilerine özel yazılımın yapısal ayarlarını sınırlı olarak değiştirebilirler. Farklı bölgelerde faaliyet gösteren firmaların tercih ettiği bir servis hizmetidir. Örnek olarak, Salesforce, Google, Cisco, Symantec, Dropbox verilebilir [27].

Platform Katmanı Hizmet Modeli

Platform hizmet modeli yazılım geliştiricilere hitap etmektedir. Geliştiricilerin sadece yazılımlarını geliştirmesine odaklanarak daha verimli çalışmalarına zemin sunulmaktadır. Yazılımın üzerinde çalışacağı kaynakların ayarlanması, düzenlenmesi ve bakımının yapılması yükü yazılımcı üzerinden alınmaktadır. Ayrıca yazılımın geliştirilmesinden yönetilmesine kadar geçen süreçlerin otomatikleştirilmesi, kesintisiz iletişimin sağlanması hizmetleri bu katmanda verilebilmektedir. Kullanıcılara servis sağlayıcı tarafından donanım ve yazılım katmanları sunularak projelerini geliştirme imkanı sağlanır. Sunulan yazılım dilleri ve araçlarını kullanarak bulut altyapısı üzerinde kendi yazılımlarını geliştirebildiği ve sadece kendi geliştirdiği yazılımlara ve yazılımın barındırılması için gerekli çevre birimleri üzerinde kontrol ve yönetime sahip olduğu servis hizmetidir. Bu hizmet sistem yönetimi,

işletim sistemi, programlama dili ortamı, veri tabanı gibi platformlar sunar. Örnek olarak Amazon, Heroku, Google, Microsoft Azure verilebilir. İhtiyaç duyulan yazılım hizmetinin bulut üzerinden sağlandığı ve kullanıcının çeşitli cihazlardan erişim sağlayarak yazılımı kullanabildiği hizmet servisidir. Kullanıcılar sadece kendilerine özel yazılımın yapısal ayarlarını sınırlı olarak değiştirebilirler. Farklı bölgelerde faaliyet gösteren firmaların tercih ettiği bir servis hizmetidir. Örnek olarak, Salesforce, Google, Cloud Foundry, Kubernetes, Redhat Openshift verilebilir [28].

Altyapı Katmanı Hizmet Modeli

Altyapı hizmet modelinde kullanıcılar servis sağlayıcısından CPU, RAM ve disk alanı gibi özelliklere sahip sanal makineler veyahut sanal disk alanları, büyük veri küme yapıları alıp kullanabilirler. Bulut altyapı hizmeti, bulut bilişimin en temel hizmetidir. IaaS ile proje oluşturup kullanıcılara tahsis edilir. Böylece sanal makine oluşturma, depolama, ağ ve diğer ana bilgisayar kaynaklarına erişmesi ve işletim sistemi dahil yazılımları geliştirip çalıştırabilmesi için olanak sağlanır. Kullanıcının bulut altyapısı üzerinde yine yönetim ve kontrolü yoktur, ama işletim sistemi, depolama, kullanılan yazılımlar üzerinde yönetim ve kontrole sahiptir. Güvenlik duvarı, yük eşleyiciler gibi ağ parçalarını seçme hakkı vardır. Örnek olarak, Amazon EC2, Rackspace, Google, Joyent, HP Cloud, Microsoft Azure verilebilir [29].

Bulut Bilişim Altyapı Servisi: OpenStack

OpenStack, açık kaynak kodlu bir bulut alt yapı hizmetidir. Bütün bulut türlerini desteklemektedir. OpenStack ölçeklenebilir ve yönetim kapasitesi yüksek platformlar oluşturulmasına olanak sağlamaktadır [30].

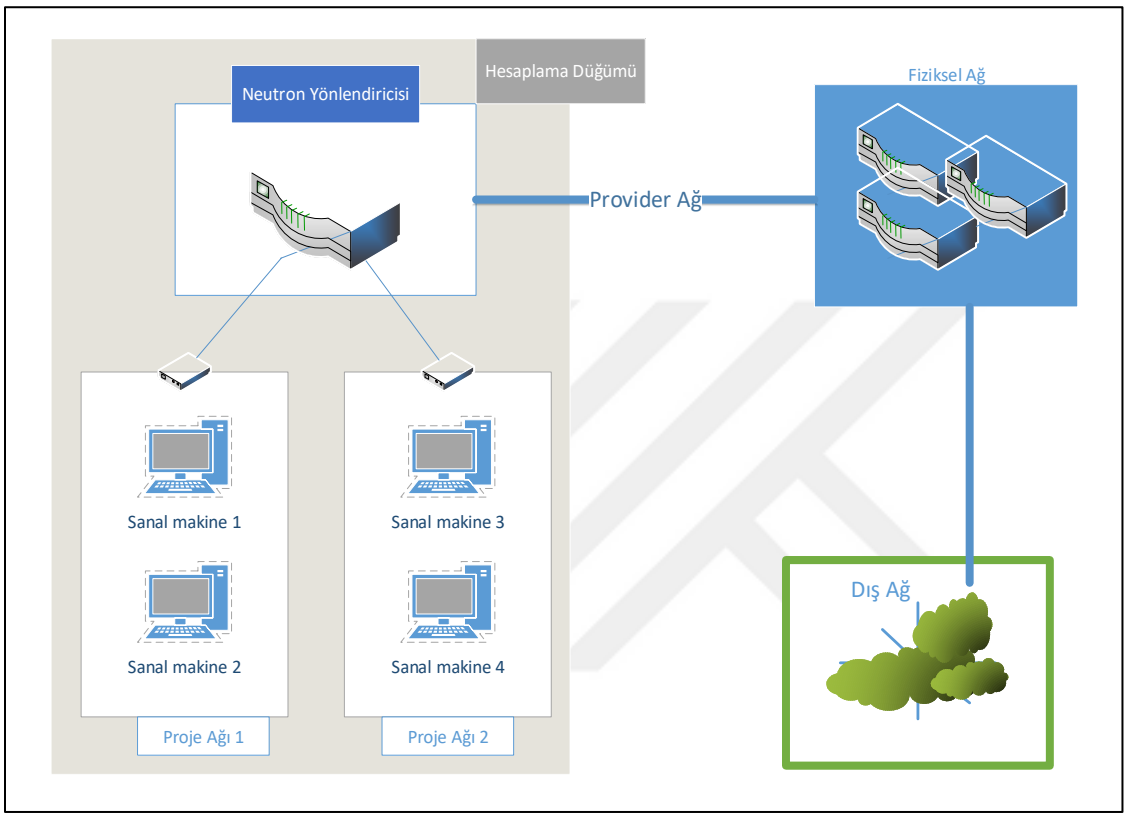
OpenStack birbiriyle ilişkili servisler kullanarak altyapı hizmeti için çözüm sunmaktadır. Her servis birbiriyle entegrasyonu kolaylaştıran (API) uygulama programlama arayüzüne sahiptir. İhtiyaçlara göre servislerin bir kısmı ya da tamamı yüklenebilir. Çizelge 2.1'de her servisin proje ismi ve kısa tanımı yer almaktadır.

Çizelge 2.1. OpenStack servisleri

Servis Adı	Proje Adı	Tanım
Yönetim Paneli	Horizon	Kullanıcı arayüzünü aktif ederek yönetici ve kullanıcılar için OpenStack kaynak ve hizmetlerini yönetmeyi sağlar.
Hesaplama	Nova	OpenStack servislerinde yapılan işlemlerin yaşam döngülerinin hesaplanması, planlaması ve sonlandırılması işlemlerini düzenler.
Ağ	Neutron	Ağ bağlantıları ve diğer OpenStack servislerin bağlantı kurulmasına olanak verir. Ağ sunucuları ve servisleri destekleyen eklenebilir mimariye sahiptir.
Nesne Depolama	Swift	Yüksek erişimli, merkezi bir beyin ve kontrol yapısı olmayan, ölçeklenebilir, dağıtık ve kısmen kararlı bir nesne deposudur. Sanal makina içerisindedir ve makina silindiği zaman bu depolama alanı da silinir.
Blok Depolama	Cinder	Bu depolama alanı sanal makina içerisinde yer almayan, ayrı bir alanda oluşturulup sanal makinaya bağlanan depolama alanıdır. Çalışır durumdaki sanal makinalar ve konteynerlar için blok depolama sağlar.
Kimlik Yönetimi	Keystone	Diğer OpenStack hizmetleri için kimlik doğrulama ve yetkilendirme hizmeti sunar. Tüm OpenStack hizmetleri için uç noktaların (endpoint) bir kataloğunu sunar.
İmaj Yönetimi	Glance	Depo ve sanal makinaların disk görüntülerini alır ve hesaplama bileşeni bu bilgiyi yapılandırma sırasında kullanır.

Bulut sistemlerinde ve OpenStack'de kullanılan sanal makine ve ağ yapısı Şekil 2.6'daki gibi olup dış ağdan alınan internet fiziksel ağ cihazları ile konfigürasyonlarda ve çeşitli kontrollerle sağlayaıcı (provider) ağ olarak OpenStack ağ bileşeni olan Neutron'a yönlendirilir. OpenStack hesaplama düğümlerinden olan ve Nova bileşeni ile kontrol edilen kullanıcıları birbirinden izole etmek için projeler kullanılır. Her bir proje ağı kendine özgü olup, her proje birbirinden izole olarak çalışmaktadır. Aynı proje dışındaki sanal makineler

ortak sağlayıcı (provider) ağını kullanarak internete çıksada, izole yapı sayesinde birbirini özel ağdan göremez. Openstack'in en önemli özelliği çok kiracılı (multi tenancy) sisteme izin vermesidir. Birden fazla kullanıcının birbirini görmeksizin sistemi kullanmasına olanak sağlanmaktadır.



Şekil 2.6. OpenStack ortamında sanal makine ve ağ yapısı

Bu tez kapsamındaki bulut üzerinde yapılan çalışmalar TÜBİTAK BİLGEM bünyesinde yer alan Bulut Bilişim ve Büyük Veri Laboratuvarı'nda bulunan prototip veri merkezindeki Ansible ile kurulumu yapılmış, yüksek erişlebilirlik özelliğine sahip OpensStack Pike üzerinde yapılmıştır.

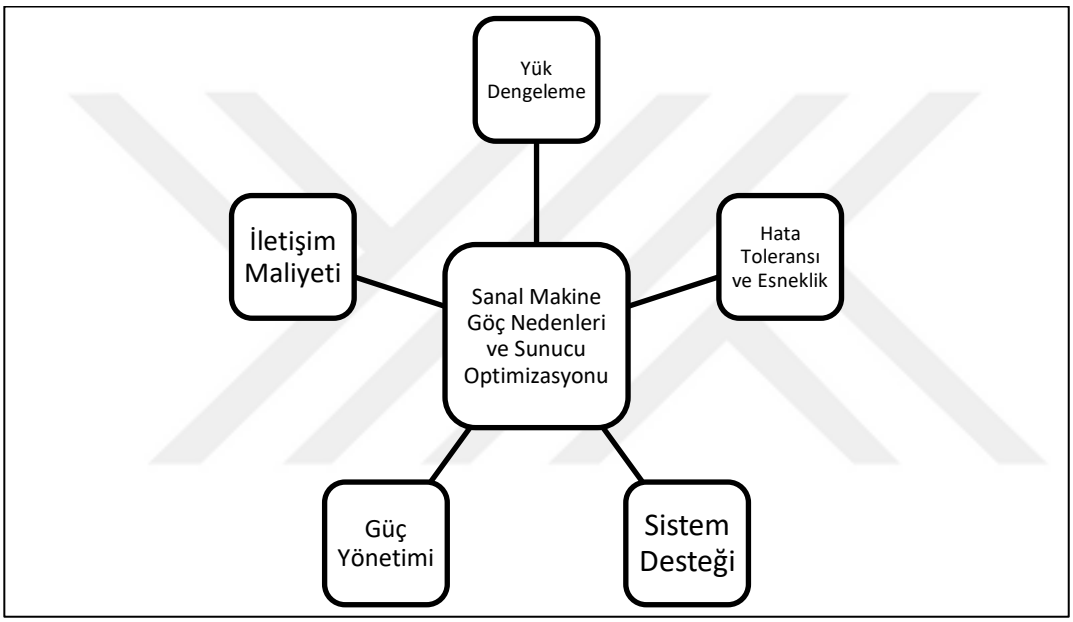
2.6. Sanal ortamlarda göç

Fiziksel olarak bulunan kaynakların mantıksal parçalara bölünmesi ile sanal kaynaklar oluşturulmaktadır. Fiziksel kaynakların bakımı, onarımı, arızası veya değiştirilmesi gibi teknik olaylara bağlı olarak veyahut yönetim açısından daha etkili, performansın artırılması, hizmet servis sözleşmesi gerekliliklerinin yerine getirilmesi, güvenlik kısıtlarının ön planda olması sebebi ile sanal kaynakların taşınması, fiziksel kaynağının değiştirilmesi, sanal

ortamlarda göç olarak adlandırılmaktadır [31]. Sanal ortamda göç temel olarak sanal makine göçü ve sanal ağ göçü olarak ayrılmaktadır.

2.6.1. Sanal makine göçü

Sanal makinanın fiziksel olarak üzerinde çalıştığı makinanın değiştirilmesi durumu sanal makine göçü olarak adlandırılır. Şekil 2.7'de sanal makine göçüne ve sunucu optimizasyonuna neden olan etkenler gösterilmektedir.



Şekil 2.7. Sanal makine göç nedenleri ve sunucu optimizasyonu

Veri merkezlerinde yaşanabilecek olası problemler, soğutma sistemlerinde yaşanabilecek sorunlar, fiziksel makinalardaki depolama alanın yetersizliği, ağ yapısındaki fiziksel yetersizlikler, veri merkezindeki yapısal sorunlar, sanal makine göçüne iten problemler olarak sıralanabilir. Ayrıca, kullanılan fiziksel sunucularda meydana gelebilecek, yazılım güncellemeleri, donanım güncellenmeleri ve çekirdek güncellemeleri gibi durumlar fiziksel makinaların üzerinde çalışan sanal makinaların çalışmasını etkileyebilir. Aynı zamanda veri merkezindeki maliyetin düşürülmek istenmesi, fiziksel makinaların ve sanal makinaların daha dayanıklı şekilde kullanılmak istenmesi gibi amaçlarla sanal makine göçü yapılabilir [32-33]. Aynı zamanda veri merkezinde yapılacak enerji optimizasyon faaliyetleri de fiziksel makinelerin kapatılmasını gündeme getirebilmektedir. Bu yüzden sanal makinelerin taşınamsı gerekmektedir. Ayrıca aşırı yük durumunun getirdiği performans kayıplarının

önlenmesi içinde sanal makinelerin taşınması konusu gündeme gelmektedir. Bu sistemin esnekliğini korumaktadır. Sisteme yeni eklenecek bir düğüme de mevcut yük dağıtılırken sanal makine göçü yapılmaktadır.

Dört farklı sanal makine göç çeşidi vardır. Bunlar temel olarak canlı (live) ve cansız (cold) sanal makine göçü olarak ikiye ayrılmaktadır. Canlı makine göçü ise kullanılan tekniklere göre; blok sanal makine göçü, gerçek sanal makine göçü, okuma izinli disk ile blok sanal makine göçü, okuma izinli disk ile gerçek sanal makine göçü olmak üzere dört başlıkta incelenmektedir [34].

Cansız (Cold) Sanal Makine Göçü

Cansız sanal makine göçü sırasında kullanıcılar makinelerini kullanamaz ve makinelerine erişemez. Sanal makine tamamen kapatılır. Sanal makinenin imajı alınır, o andaki anlık görüntü alma işlemi uygulanır. Daha sonrasında alınan imaj kullanılabilir bir imaj haline getirilir. Eğer var olan bulut sistemi içerisinde başka bir fiziksel makineye taşıma gerçekleştirilecek ise sistemdeki çizelgeleyici tarafından belirlenen hedef adreste oluşturulan sanal makine imajından sanal makine tekrar oluşturulur.

Eğer taşıma işlemi mevcut bulut sisteminden başka bir bulut sistemine veyahut başka bir sanallaştırma ortamına gerçekleştirilecek ise, anlık görüntüsü alınarak oluşturulan sanal makine imajı güvenli bir protokol ile hedef sistemdeki belirlenen adrese gönderilir. Yeni ortamda aktarılan imaj ile ortam özellikleri kullanılarak yeni bir sanal makine oluşturulur. Kullanılan imaj içerisinde sanal makinenin taşıma durumundan hemen önceki bütün özelliklerini barındırmaktadır. Bu süre içerinde sanal makine kullanılamaz durumda olup, içerisinde herhangi bir servis çalıştırılması mümkün değildir. Ancak sistemler arasında taşıma işlemi sırasında en sık tercih edilen taşıma yöntemidir. Mevcut sistem içerisindeki, makineler arasında yapılacak göç işlemleri sırasında ise tercih edilmez [35].

Canlı (Live) Sanal Makine Göçü

Gerçek canlı sanal makine göçünde makinaya ait paylaşılan hafıza alanı veya RAM olarak adlandırılan hafızanın taşınması amaçlanmaktadır. Sanal makine çalışır durumdadır, disk üzerinde tutulan bilgiler ve veriler kolaylıkla başka bir adresten erişilebilir hale getirilebilir.

Bu göç türünde kullanılan RAM içeriğinin taşınması amaçlanmaktadır. Veri kaybının en aza indirgenerek kullanıcılara herhangi bir gecikme hissettirilmez. RAM yoğunluklu bir işlem o anda çalıştırılmıyor ise bir ping süresini geçmeyecek şekilde veri kaybı olmadan göç işlemi gerçekleştirilir.

Okuma izinli canlı sanal makine göçünde, göç işlemi sırasında makine ile bağlantı kesilmez, kullanıcıya disk üzerinde herhangi bir yazma işlemine izin verilmez. RAM üzerindeki bilgilerini okumaya devam edebilir.

Blok canlı makine göçünde ise sanal makine askıya alınmış durumdadır, işlem sırasında sanal makine çalışan işlemler korunur bu yüzden canlı sanal makine göçü türünde olarak değerlendirilir ancak makine askıda olduğu için kullanıcılar taşıma işlemi sırasında erişim sadece diske erişim sağlayabilirler, RAM'e ulaşım sağlayamazlar.

Okuma izinli blok sanal makine göçünde ise askıya alınmış sanal makinenin herhangi bir disk, RAM ya da yerel depolama gibi bir birimine erişim yoktur [36-37].

Canlı Sanal Makine Göçü Aşamaları

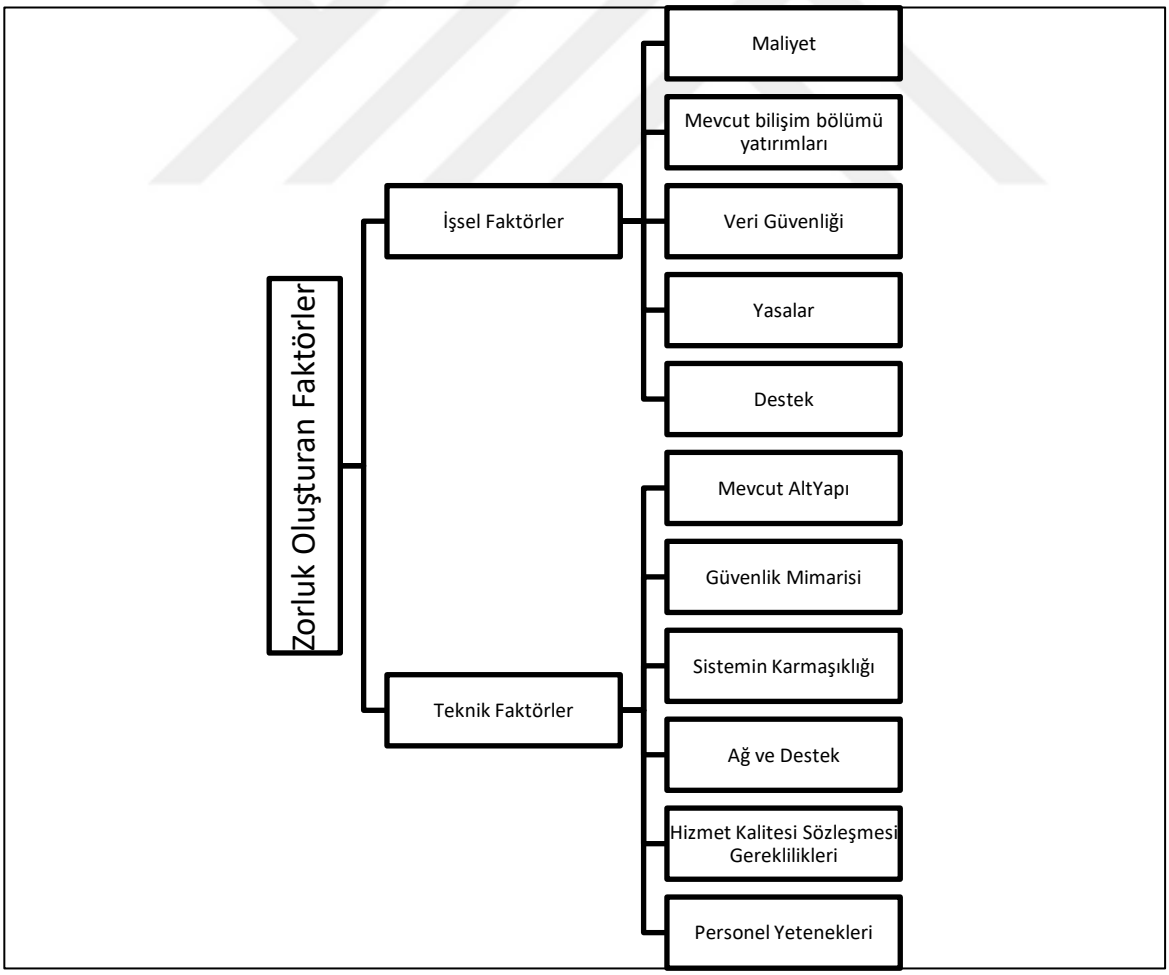
Sanal makine göçü türlerinden en çok tercih edilen gerçek canlı sanal makine göçü olup 5 temel aşamadan oluşmaktadır.

- Ön-Taşıma: İlk aşamada hesaplama düğümlerinde bulunan sanal makine için yeni bir hesaplama düğümü belirlenir. Elle atama yapılabileceği gibi otomatik olarak hesaplama servisinin çizelgeleyici bileşeni (nova-schedular) tarafından da seçilmesi sağlanabilir.
- Rezervasyon: İkinci aşamada hedef seçilen hesaplama düğümü üzerinde bulut sistemi için bir alan rezerve edilir ve yeterli kaynağın yeni düğüm üzerinde bulunduğu doğrulanır.
- Tekrarlamalı (Iterative) Ön Kopya: Üçüncü aşamada hipervizör sanal makinayı yeni düğüme kopyalar. Bu aşamada eski düğümde değişen bellek tekrarlamalı olarak yeni düğüme kopyalanır.
- Durdurma ve Kopyalama: Sanal makine durumu her iki düğümde de yaklaşık olarak eşitlenince hipervizör eski düğümde olan sanal makinayı beklemeye alır ve kalan farklı belleği yeni düğüme kopyalar.

- Tamamlanma: İki sanal makinanın durumları tam olarak eşitlenince hipervizör yeni düğüm üzerindeki sanal makinayı çalıştırır ve eski düğüm üzerindeki sanal makinayı siler. Son aşamada eski düğüm üzerinden sanal makinanın kaldırılmasından önce hata oluşması durumunda her şey geri alınarak sanal makinanın eski düğüm üzerinde çalışması devam eder [38].

2.6.2. Sanal makine göçü zorlukları

Sanal makineler üzerinde göç işlemi gerçekleştirilirken çeşitli zorluklar yaşanmaktadır. Göç işlemi makine üzerinde yapılan ve çalıştırılan işlemleri etkilemektedir. Sanal makine göçüne karar verilirken, taşınma işlemini doğuran olay analiz edilmeli ve yaşanacak problemler göz önünde bulundurulmalıdır. Göç işlemi zorlukları [39] Şekil 2.8'deki gibi temel olarak işsel ve teknik faktörler olmak üzere ikiye ayrılmaktadır.



Şekil 2.8. Sanal makine göçü zorlukları

İşsel faktörler; maliyet, mevcut bilişim bölümü yatırımları, veri güvenliği, yasalar ve destek olarak sıralanabilir. İş planlamasını ve yapılan işlerin bağımlı olduğu faktörlerin sanal makinelerin taşınmasında çeşitli zorluklar oluşturduğunu söylemek mümkündür. Taşıma işleminin sebep olacağı maliyet hesaplanmalı ve iş gücü ve kaynak kullanımı planlamalıdır. Kullanılan sanal makinenin yüksek güvenlikli olması veyahut kullanıcılara teminatı verilen şekilde gizliliğinin korunması kapsamında yasaların getirdiği yükümlülük ve şartların yerine getirilmesi gerekmektedir. Aynı zamanda yönetim tarafından sağlanacak desteklerde sanal makine göç işleminde çeşitli zorluklar oluşturan faktörler içerisinde yer almaktadır.

Teknik faktörler ise mevcut altyapı, kurulmuş olan güvenlik mimarisi, sistemin karmaşıklığı, ağ ve destek yapıları, hizmet kalitesi sözleşmesi ve çalışan personel yetkinlikleri olarak sıralanabilir. Sanal makinelerin göç ettirilmesi sırasında yaşanacak kesintiler ile hizmet kalitesi sözleşmesi ihlal edilmemeli, mevcut güvenlik sistemleri ve güvenlik duvarlarındaki kurallar göç işlemine izin verecek protokolleri barındırmalıdır. Ayrıca taşınacak makineler farklı bir ağa taşınacaksa, ev sahibi sistem ile yeni sistem arasında bir bağlantının kurulması gerekmektedir. Konuyla ilgili çalışan personelin yetkinliğinin bulunması göç işleminin gerçekleştirilmesi ve kesintinin en az şekilde yaşanması için önemlidir.

2.6.3. Sanal ağ göçü

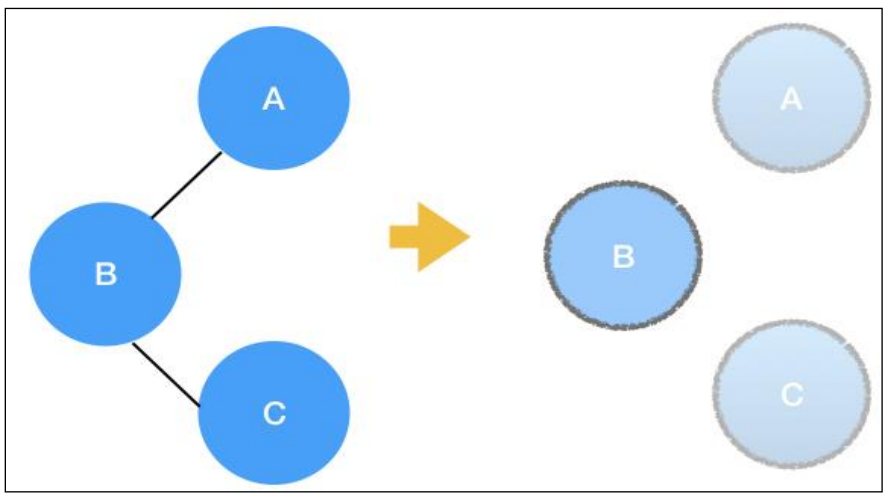
Ağ yapılarında göç işlemi sırasında Şekil 2.9'de yer alan sanal düğüm, göç ettirilen düğüm, ilk ve son bağlantılar, tünel bağlantısı ve geçici bağlantılar şekilleri kullanılmaktadır.



Şekil 2.9. Kullanılan araçların özellikleri

Sanal düğüm; bir sanal makinenin ağ yapısı elemanı olup ona bağlı olduğu varsayılan sanal bir düğümdür. Göç ettirilen sanal düğüm; göç için seçilmiş ve taşınmasına karar verilmiş sanal bir düğümdür. İlk ve son bağlantılar; sanal düğümler arasında bulunan uçtan uca bağlantı sağlayan yollardır. Düğümler, bağlantılar üzerinden haberleşmekte ve veri akışı gerçekleştirilmektedir. Tünel bağlantısı; göç ettirilmeye karar verilen düğümün mevcut bulunduğu düğüm ile göç için oluşturulan ve sonrasında kullanılacak düğüm olan klonlanmış düğüm arasında kurulan bir tüneldir. Mevcut düğümdeki veriler bu tünel aracılığıyla ile yeni düğüme aktarılır. Geçici bağlantılar; taşınan sanal düğüm ile, taşınmamış durumdaki, göç ettirilen düğümün komşu bağlantılarının kaybolmaması için kurulan geçici olarak oluşturulmuş bağlantılardır. Taşıma işlemi tamamlandığında geçici bağlantılar kaldırılarak ilk ve son bağlantılar uçtan uca tekrar kurulur.

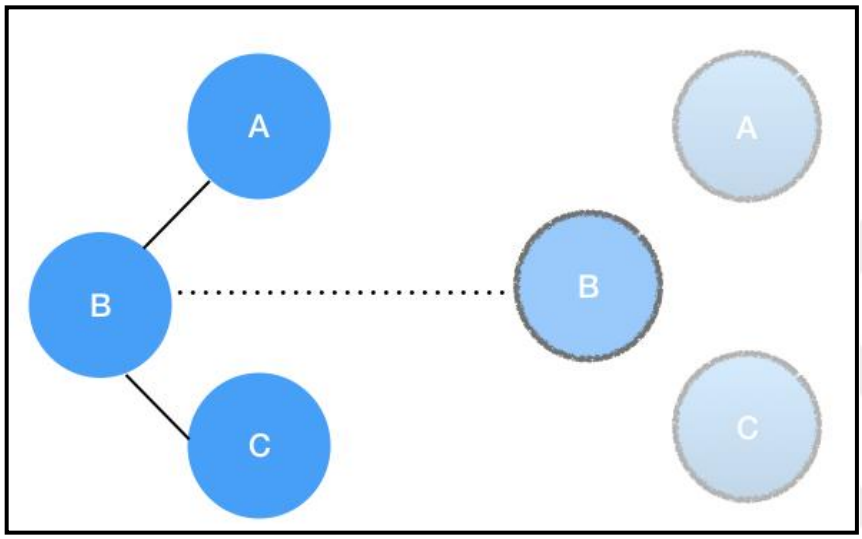
Sanal ağ göçü; Şekil 2.10'daki gibi mevcut ağ topolojisi bir bölgeden başka bir bölgeye taşınmasıdır [40]. Göç işlemi mevcut ağ topolojisinin bozulmadan birebir klonlanarak yerinin değiştirilmesini hedeflenmektedir. Mevcut düğümlerin ve bağlantı yollarının korunması için temelde tek tek taşıma yapılmakta ancak maliyet ve performans açısından düşünüldüğünde grup halinde taşıma konusunda geliştirmeler yapılmakta ve çeşitli algoritmalar uygulanmaktadır.



Şekil 2.10. Sanal ağ yapısındaki düğümlerin ilk hali

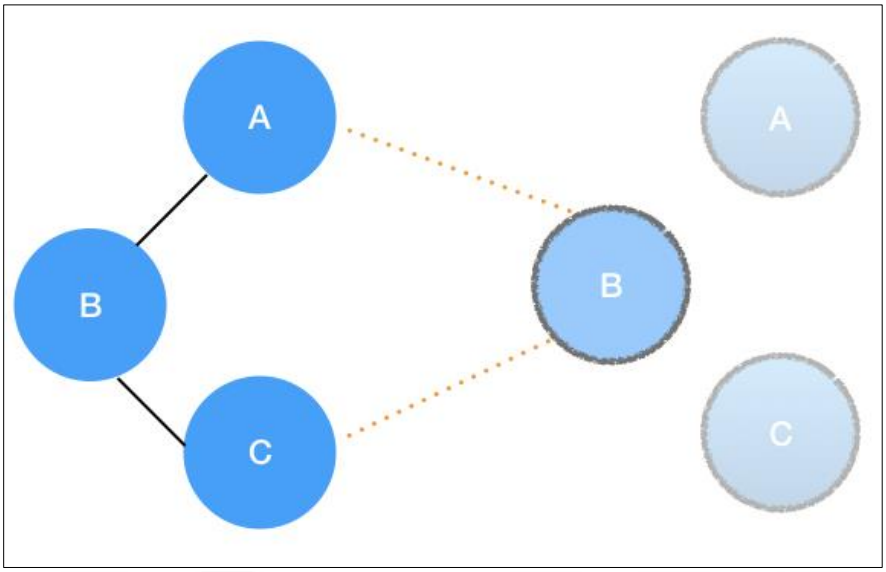
Bu taşıma işlemi sırasında düğümler, sanal düğüm ve taşınacak düğüm olarak belirlenir. Eski konumdan yeni konuma herhangi bir değişiklik yapılmadan taşıma işlemi gerçekleştirilmesi hedeflenir.

Şekil 2.11'daki ağ topolojisinde belirlenen ve taşınmasına karar verilen B düğümünün taşınması için yazılım tanımlı ağlar ile kontrol edilen sistemde kontrolcü katmanında seçilen B düğümü için bir sanal kopya oluşturulur. İki düğüm arasında bir tünel oluşturulur. Oluşturulan tünel ile mevcut B düğümündeki veriler taşıma noktasındaki B düğümüne aktarılır. Tünelin sahip bant genişliği ne kadar yüksek olursa veri akışı o kadar hızlı gerçekleşir. Bu aşamada mevcut B düğümü üzerindeki verilere ulaşılamaz olup bir değişiklik yapılamaz, düğüme bağlantı kurulamaz.



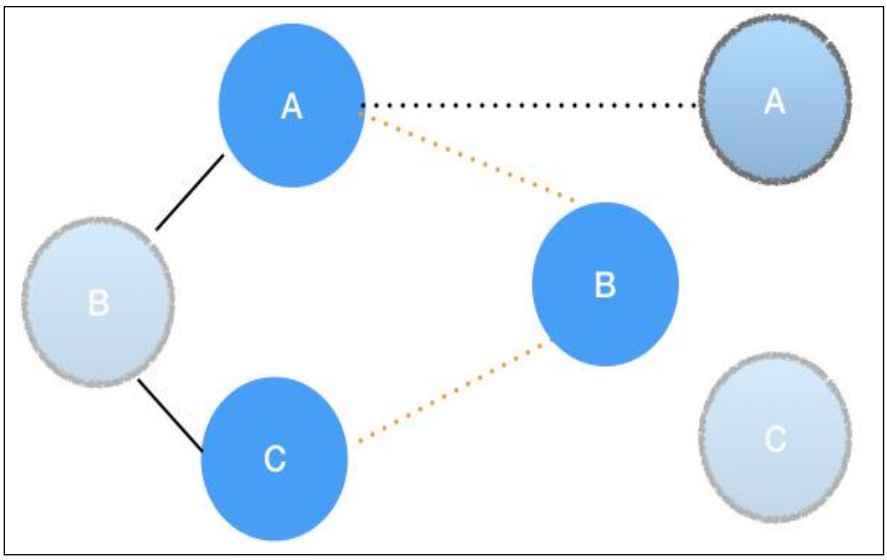
Şekil 2.11. Belirlenen düğümde tünelin kurulması

Tünel bağlantısı ile verilerin aktarım işlemi tamamlandıktan sonra şekil 2.12'deki gibi taşınan B'nin bağlantılı olduğu komşu düğümleri ile yeni kopyalanan B düğümü arasında geçici bağlantılar oluşturulur. Bu şekilde komşu düğümler ile bağlantılar kaybedilmeden kopyalanabilir ve bağlantı yolu yedeklemesi sağlanmış olur.



Şekil 2.12. Geçici bağlantıların kurulması

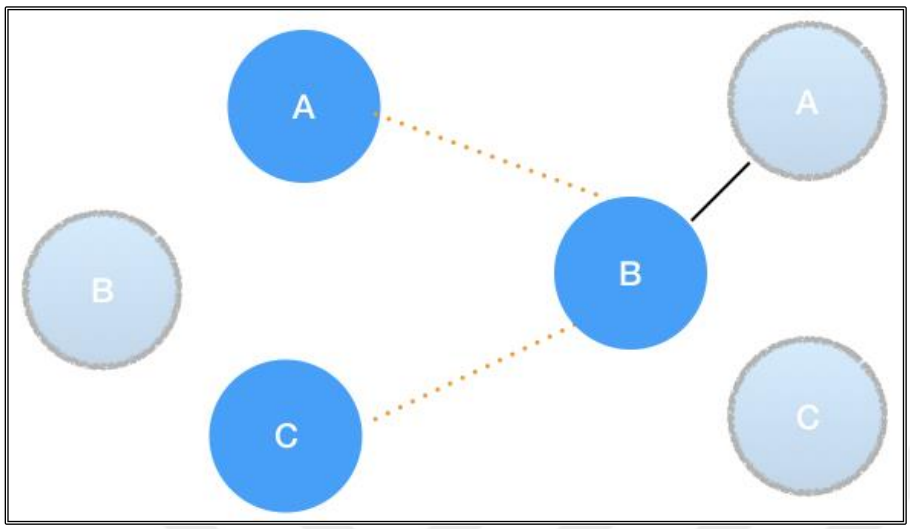
Geçici bağlantıların kurulması ardından B'ye komşu olan düğümlerin kopyalanması işlemine başlanır. B'nin komşularından birisi olan A düğümü kopyalanır. Şekil 2.13'deki gibi mevcut A düğümü ile yeni A düğümü arasında bir tünel bağlantısı oluşturulur. Oluşturulan yeni tünel içerisinden veri aktarımı gerçekleştirilir.



Şekil 2.13. A düğümünün taşınması

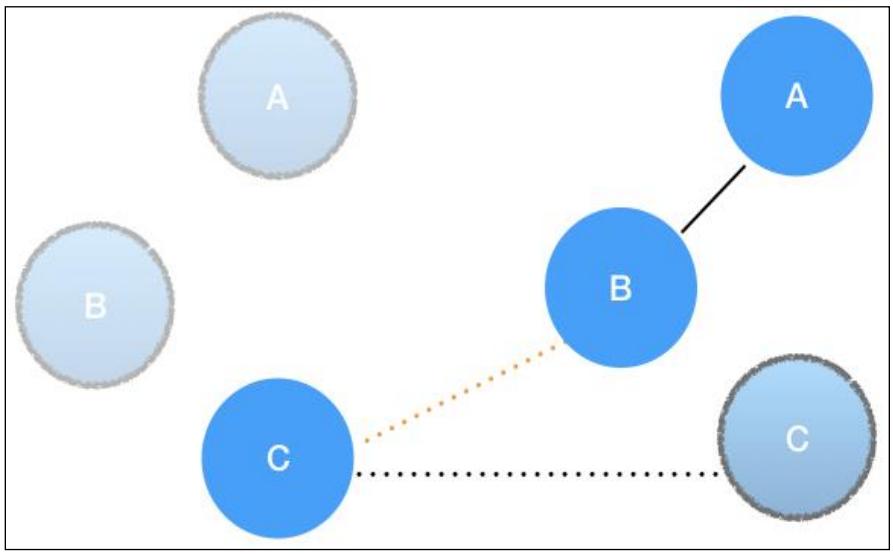
Kopyalanan A düğümü ile ortaya çıkan yeni A düğümü ve daha öncesinde taşıma işlemi tamamlanan B düğümü arasında şekil 2.14'deki gibi uçtan uca kalıcı bir bağlantı kurulur.

Daha sonrasında bağlantı kaybını önlemek için kurulan geçici linklerden olan eski A konumundaki olan düğüm ile aradaki geçici link kaldırılır.



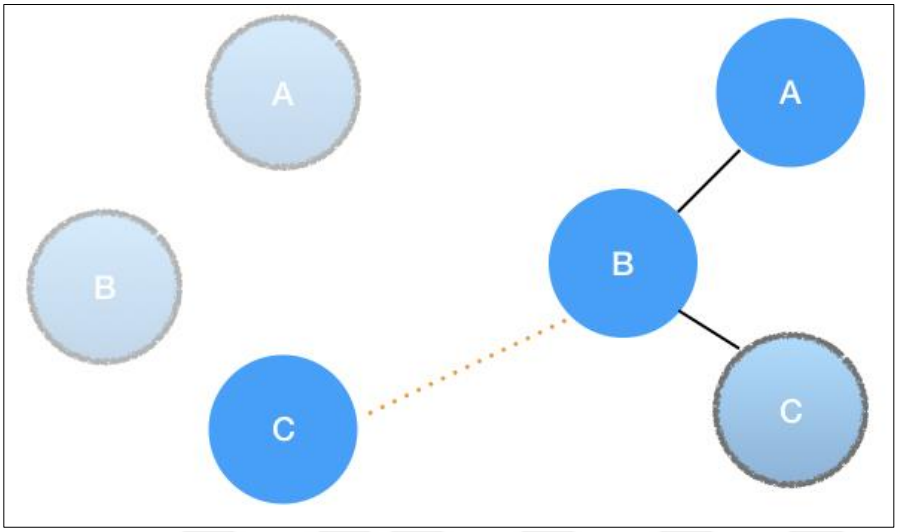
Şekil 2.14. Komşu düğümler ile geçici bağlantıların kurulması

B düğümü için yapılan işlemler benzer şekilde B düğümünün diğer komşusu olan C düğümü içinde Şekil 2.15'deki gibi gerçekleştirilir. C düğümünün kopyalanması için yeni ortamda C'nin klonu olacak şekilde yeni bir sanal düğüm oluşturulur. Ardından yeni oluşturulan C düğümü ile mevcut C düğümü arasında tünel bağlantısı kurulur. C düğümü verileri tünel yardımı ile aktarılır.



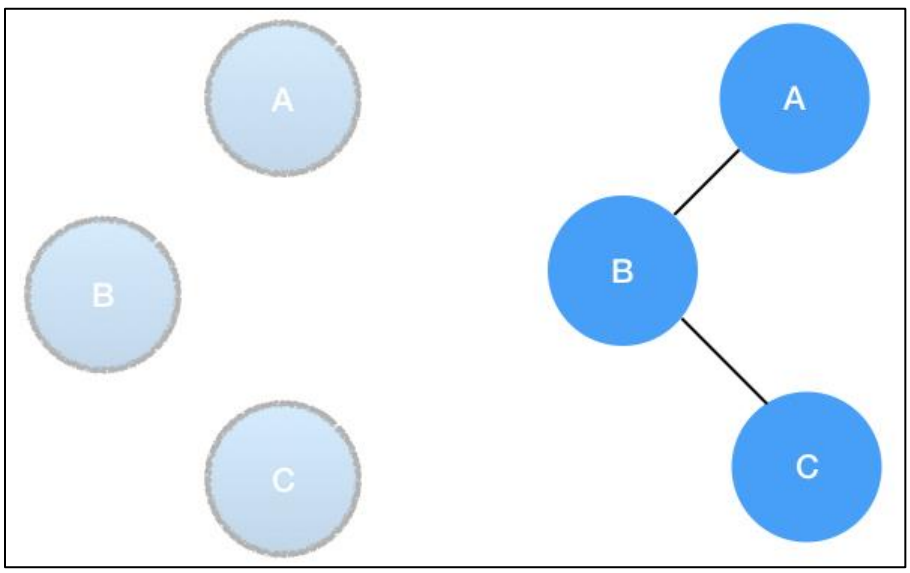
Şekil 2.15. C tünel bağlantısı ve geçici bağlantılar

Veri aktarım işlemi tamamlandıktan sonra Şekil 2.16'daki gibi yeni ortama daha önce taşınmış olan komşu düğüm olan B ile kalıcı ilk ve son bağlantılar oluşturulur.



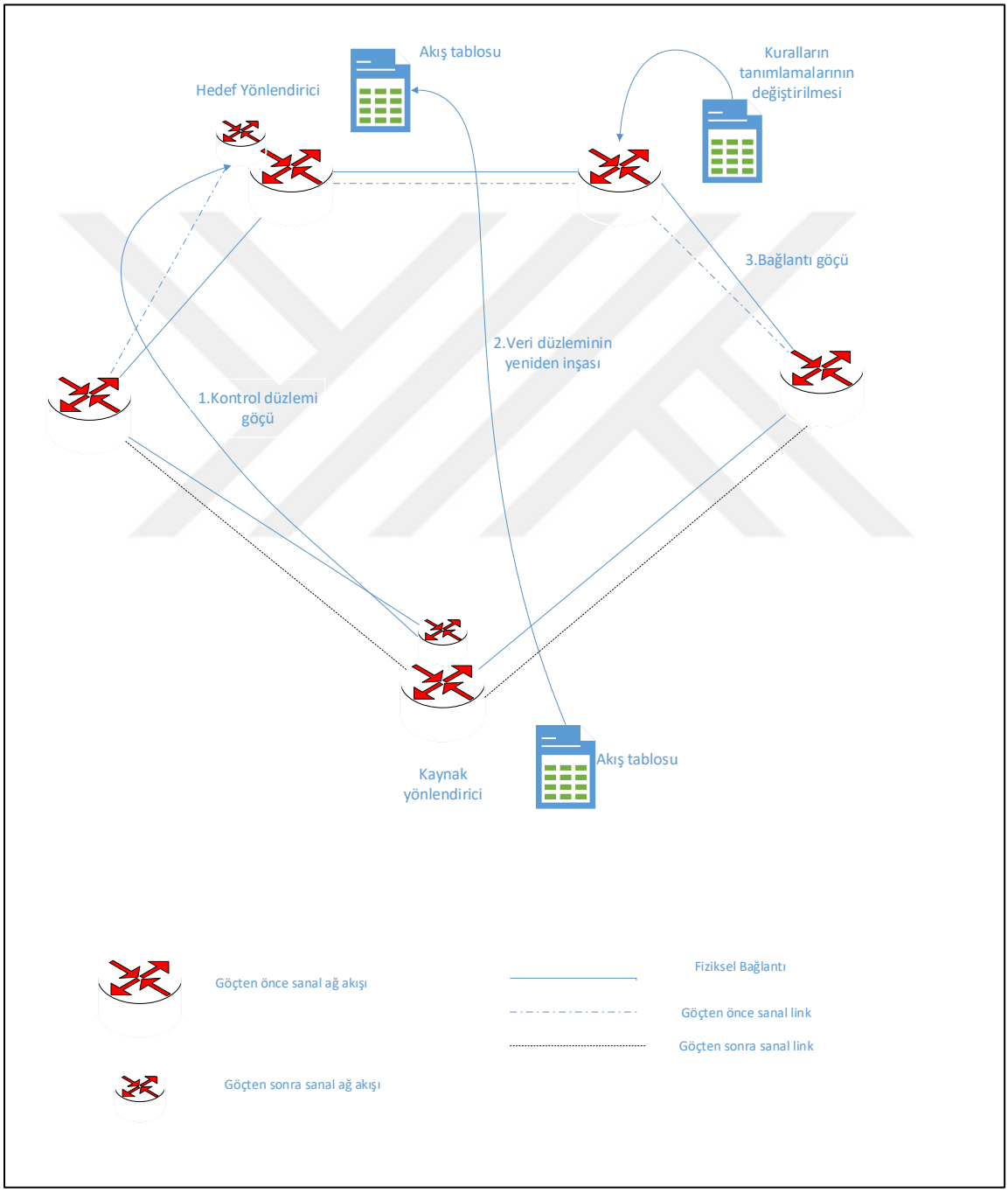
Şekil 2.16. Komşu düğümler ile kalıcı linklerin oluşturulması

Son olarak Şekil 2.17'daki gibi geçici kurulan bağlantılar kaldırılır ve mevcut ağ topolojisi tamamen taşınmış yani göç işlemi tamamlanmış olur [41]. Ağın tamamen taşınması ile yeni ortamda, eski ortamda kullanılan ağ topolojisi yeniden oluşturulur, verilen servisler ve hizmetler kaldığı yerden devam edebilir.



Şekil 2.17. Ağın tamamen taşınması

Şekil 2.18'de yer aldığı gibi yazılım tanımlı ağların kullanıldığı yapılarda sanal ağ göçü ve tablolaların güncellenmesinden bahsedilmektedir. Kontrol düzlemi göç ettirildiğinde veri düzlemi yeniden inşa edilir ve akış tablosunda değişen adresler yeniden güncellenir. Yeni güncellenen akış tablosu ile elemanlar arası göç bağlantıları da taşınarak yeniden inşa edilir ve güncellenir. Veri akışına güncellenen tablolar ve bağlantı yolları üzerinden devam edilir.



Şekil 2.18 Yazılım tanımlı ağlarda sanal ağ göçü ve veri akışı

3. LİTERATÜR TARAMASI

3.1. Sanallaştırma ve Bulut Bilişim

Bulut sistemleri ileveri merkezleri arasında, yüksek performanslı ağlarla birbirine bağlanmış bir dizi heterojen küme seti ve yardımcı programlar, platform ve altyapı gibi bilgisayar kaynaklarının paylaşımı sağlanmıştır [42-43]. Özellikle, bulut bilişim, kesin kaynak gereksinimi veya değişken kullanım şekilleri olmayan işletmeler ve kuruluşlar için iyi bir seçenektir. Aksi takdirde, kurumlar ve şirketler, etkin nokta yükselmelerinden kaçınmak için yoğun trafiğe girene kadar tesislerde geleneksel bilişim teknolojileri kaynaklarını aktif olarak sağlamalıdır [44].

Bulut bilişim, açık rekabet gücü yüksek pazarda farklı sağlayıcılar tarafından sunulan çok çeşitli hizmetleri içerir. Örneğin, Amazon EC2 sanal hizmet, veri aktarımı ve bir hizmet olarak altyapı biçiminde depolama için bant genişliği (altyapı hizmeti) [45], Microsoft Windows Azure gibi tümleşik Geliştirme Ortamı (IDE) araçlarını sağlar. Geliştiricilerin bulut tabanlı yazılımlarını geliştirmeleri ve üretimlerini bulut ortamında yayınlamaları için bir hizmet (platform servisi) [46] ve Salesforce, farklı endüstriler için bir hizmet olarak yazılım şeklinde müşteri ilişkileri yönetimi (CRM) yazılımı sağlamaktadır [47]. Verimsiz kaynak tahsisi, daha yüksek elektrik tüketimine neden olur. Sanallaştırılmış ortamdaki bulut veri merkezlerinde güç yönetimi için sunucu konsolidasyonu, maksimum sayıda sanal makineyi minimum fiziksel makine sayısına eşlemek için sanal makine geçişlerini tetikler; daha sonra gereksiz fiziksel sunucuları kapatır [48]. Bununla birlikte, canlı ve canlı olmayan sanal makine göçleri, yöneticilerin sistem bakımı, yük dengeleme, kullanılabilirlik gibi sebepler için veri merkezlerinin içinde ve genelinde gerçekleştirilmesine olanak tanır. Üstelik, sanal makine geçişi LAN / WAN olarak, zamansız veya zorunlu sanal makine göçlerinde gerçekleşir. Sanal makine göçü sırasında performans ve ağ bant genişliğini çok fazla etkilediği için, ağa duyarlı diğer uygulamalar performans düşüşlerinden etkilenir [49].

3.2. Sanal Makine Yerleşimi

Sanallaştırma, aynı fiziksel sunucu, fiziksel makine üzerinde farklı işletim sistemleri örneklerinin barındırılmasına izin veren bir tekniktir [50]. Hipervizör, sanallaştırma ortamının ana bileşenidir; her işletim sistemi örneği için ayarlanan sanal makine monitörlerine donanım kaynakları sağlar. Kullanıcı başvurusunun fiziksel makine üzerindeki sanal makine tahsisi sanal makine yerleşimi olarak adlandırılmaktadır [51]. Sanal makine yerleşimi, yer tahsisinde çakışan birkaç hedef nedeniyle Np-Hard problemine aittir. 2 boyutlu sanallaştırma platformunu gösterir. Xen, VMWare, KVM, Hyper-V gibi hipervizörler bulut veri merkezlerinde tipik sanallaştırıcılardır [52]. Sanallaştırma, kaynak kullanımı ve taşınabilirliğide maksimize etmek için en iyi seçenektir. [7, 53-54] gibi literatürlerde, bir sanal makine yerleşimi, değişken kutu boyutları ve minimum miktarda paketlenmesi gereken fiyatlar ile birlikte bir kutu doldurma problemi olarak formüle edilmiştir [55]. Bunun nedeni, kullanıcı gereksinimlerinin genellikle mümkün olan en düşük işletme maliyeti olarak karşılanacak hizmet seviyesi anlaşması (HSA) biçiminde belirtilmesi, sanal makine tahsisatının zorlu bir iş olmasıdır.

Bulut veri merkezlerinde sanal makine yerleşimi iki durumda ortaya çıkar. Birincisi, kullanıcı isteğini sorduğunda, dağıtıcı (aracı) modülü, kullanıcının hizmet kalitesi gereksinimlerini karşılamak için en azından kaynaklarının bulunduğu en yakın veri merkezinde bir sanal makine başlatır, bu yüzden sanal makine yerleştirme, yanıt süresini garanti etmek için mümkün olduğunca hızlı olmalıdır [56]. İkincisi, dinamik sanal makine yerleştirme planları, kullanıcılar ve sağlayıcılar arasındaki çelişkili gereksinimi dengelemek için sunucu birleştirme teknikleriyle sanal makine geçişini tetikler. İkinci sanal makine yerleşimi türü dinamiktir ve doğrudan sistem kontrol programlarına bağlıdır. Sanal makine yerleşimi hem canlı hem de canlı olmayan sanal makine geçişinde gerçekleştirilir, oysa hedef makine belirlenmeli ve gereken kaynak göçten önce korunmalıdır. Genel olarak, sanal makinelerin sanal makinelerinin haritalanması problemi büyük problemler için NP zorlu problemlere aittir, ndeterministik yaklaşımlar kabul edilemezdir, dolayısıyla sanal makine yerleştirme problemini ortaya çıkarmak için literatürde farklı sezgisel algoritmalar ortaya çıkmıştır [51, 57-58].

3.3. Sanal Makine Göçü

Sanallaştırma, mobil bilgisayar kullanımı, taşınabilirlik, bağımsız platform ve kaynak kullanımını en üst düzeye çıkarmak için modern veri merkezlerinde yaygın olarak uygulanır. Ayrıca, güç yönetimi, yük dengeleme, kullanılabilirlik, hizmet ihlali oranının azaltılması amacıyla sanal makine göç işlemleri uygulanabilir. Bunun için, sanal makine geçiş işlemi, CPU içeriği ve kaynakların ana bilgisayarlara aktarılması gibi kaynak durumlarını bir araya getirir ve bunları ana bilgisayarlar arasında aktarır [59]. Göç süreci, canlı ve canlı olmayan modlarda, hipervizör tarafından gerçekleştirilir. Canlı sanal makine geçişi, sanal sayfaları aktarılırken kullanıcıya kesintisiz servisler sağlar. Canlı olmayan sanal makine geçişinde kullanıcının uygulama yürütmesi sanal görüntüleri aktarılmadan önce kesilir. Hedef makinesine aktarılan sanal makine sayfaları tamamlandığında, yürütmeye devam edilir [60]. Bununla birlikte, canlı olmayan sanal makine geçişi etkileşimli internet (web) uygulamaları için uygun değildir. Aktif uygulamalar için performansın düşmesine neden olur. Canlı sanal makine geçişi, kopyalama öncesi ve kopyalama sonrası iki modelde sınıflandırılmıştır. Canlı sanal makine göçünün temel amaçları uygulama performansını artırmak, genişlik kullanımını en üst seviyeye çıkarmak ve sunucu duruş süresini en aza indirmektir [61].

Ayrıca, literatürdeki aksama süresi kavramı, kaynak sanal makinenin askıya alınma süresi ile hedef makine üzerinde sanal makinenin yeniden başlatılmasından sonraki zamana kadar olan süreyi ve toplam göç zamanının, toplam makine göçünün hedef makinede tamamlanmasına kadar geçen başlangıçtaki başlangıç süresi arasındaki süredir. Her iki geçiş modelinde de sanal makine sayfalarının kaynak makineden hedef makinenin hafızasına yinelenerek kopyalanması gerekir. İlk model, ön kopyalama yaklaşımı, hedef makineye devam etmeden önce toplam sanal makine belleğini kaynak makinesinden aktarır; Altı adımda bulunur: (1) Göç öncesi: hedef düğümü seçmek için, (2) Rezervasyon: hedef düğümde gerekli kaynağı ayırmak için, (3) Yinelemeli Ön kopya: sanal makine sayfaları sınırlı çevrelerde aktarılır, (4) Durdur ve Kopyala: kaynak sanal makineyi kaynak düğümü üzerinde askıya alır ve kalan trafiği hedef düğümün içine yönlendirir ve koordine eder, böylece ARP oluşturur, (5) Taahhüt: kaynak koddan serbest sanal makine ayarlar ve (6) Etkinleştirme: sanal makine normalde hedef düğümde başlatılır [31].

Literatürdeki çalışmalar eserler üzerinde çalıştıktan Çizelge 3.1’deki gibi göç örneği sınıflandırılması yapılabilir.

Çizelge 3.1. Göç örnekleri sınıflandırması

Çalışma	Ana Sebep	Hipervizor	Metot	Değerlendirme	Bant genişliği optimizasyonu
Diallo ve ark (2017) [62]	Erişilebilirlik	Xen	Naïve Bayesian makine öğrenmesi	Deneyler kümesi	Sayfa önceliklendirme
Liu ve ark (2013b) [63]	Güç yönetimi / yük dengeleme	Xen	Linear regression	Gerçek iş yükü/ SPEC benchmark	Sanal makine belleğini yeniden boyutlandırma
Al-Kiswany ve ark (2011) [64]	Erişilebilirlik / ölçeklenebilirlik / mobil bilgi işlem	QEMU/ KVM	Profil ve tahmin	Prototip	Tekilleştirme
Wood ve ark (2015) [65]	Ölçeklenebilirlik	Xen	Super fast hash (Hsieh, 2018)	Prototip/ SPEC benchmark	Sıkıştırma
Jin ve ark (2014) [60]	Hata toleransı / güç yönetimi / bakım	Xen	LZO sıkıştırma	Varity workload/ benckmark	Compression/light weight page
Hines ve ark (2009) [66]	Bakım	Xen	Dynamic self ballonong (Waldspurger, 2002)	Benchmark/ stress test	Bubbling
Hu ve ark (2013) [67]	Bakım / yük dengeleme	Xen	XOR Binary RLE (Pountain, 1987)	Test Case	Delta compression
Forsman ve ark (2015) [52]	Yük dengeleme	Xen/ KVM/ VMWare	Binary exponent based on algorithm	Simulation (Omnet ++)	Sanal makine belleğini yeniden boyutlandırma
Deshpande ve ark (2012) [68]	Bakım / yük dengeleme / güç yönetimi	QEMU/ KVM	SHA hash	Prototyping/ Benchmark	Tekilleştirme
Deshpande ve ark (2011) [69]	Bakım / yük dengeleme / güç yönetimi	QEMU/ KVM	Super fast hash (Hsieh, 2018)	Mixed workload/ database benchmark (MySQL, 2017)	Tekilleştirme/light weight page

Çizelge 3.1. (devamı) Göç örnekleri sınıflandırması

Riteau ve ark (2011) [70]	Yük dengeleme / güç yönetimi	KVM	SHA-1 (National Institute of Standarts and Technology)	Prototip	Tekilleştirme

3.4. Etkili Sanal Makine Göçünün Planlaması

Yeni bir model geliştirilerek göç maliyeti ve işlem sırasında yaşanan gecikmeler en aza indirmek hedeflenmektedir. Bu yüzden düğümlerin tek tek göç ettirilmesi üzerine çalışma yapılmış olup gruplama şeklinde taşıma ile yeni bir yaklaşım önerilerek sanal ağ göç çizelgeleyicisi problemine çözüm sağlamak amaçlanmıştır [71]. Tasarlanan yeni çizelgeleyici gerçek bir ortam imkânı sunan planetlab üzerinde gerçekleştirilmiştir. Temel olarak ağ göçünde gerçekleştirilmesi gereken aşamalar hedefleyen çalışma, tasarlanan algoritma ve çizelgeyicilerin performans ölçümleri için simülasyonun Planetlab'da gerçeklenmesini konu almış ve uzun soluklu çalışmalarda gerçek ortam olarak planetlab kullanılmasını önermektedir. Ayrıca göç çizelgeleyici ile göç sırasında yaşana kesinti ve bölüntülerin en aza indirilmesi küçük ölçekli deneylerle gerçeklenmiştir. Var olan alternatif göç yolları araştırılmıştır. Ortalama maliyet düşürülmüştür [72].

Yazılım tanımlı ağlar ile kurulmuş klasik ağ yapılarının yanı sıra geniş alanlar üzerine kurulmuş ağ yapılarında etkin göç üzerinde değerlendirmeler yapılmıştır. GENI adapte edilerek hatalı paketlerin onarımı, enerji kazancı ve saldırılardan korunma hedeflenmiştir. Kullanılan algoritmanın bazı kısıtları ile karşılaşılmıştır [73]. Göç sırasında ortaya çıkan maliyeti düşürmek hedeflenmiş ve markov decision process kullanılmıştır. Göç sırasında etkili bir çözüm için çeşitli politikalar belirlenmesini öneren çalışma en kötü durumda %20 hata payı ile çalışmaktadır [74]. FTPA algoritması ile %10 varyasyon ile downtime süresinin %40 ve %20 azalma gözlemlenmiştir [75].

3.5. Sanal Ağ Göçü

Ağ yapılarındaki artan ihtiyaçların karşılanabilmesi için sanallaştırılmaya başvurulmaktadır [76]. Ağ yapılarının sanallaştırılması ile ağ trafiği daha kolay izlenebilir ve yönetilebilir bir

hale gelmiştir. Sanallaştırma teknolojileri temel olarak veri merkezlerindeki bulut yapılarında kullanıldığı gibi bulut sistemleri sanal ağ yapıları ile bütünleşmiş şekilde çalışmaktadır [77]. Bulut sistemlerinde kullanılan sanal makinaların performans iyileştirmeleri ve bakım onarım için sanal makine göçü işlemine başvurulmaktadır [78]. Aynı zamanda veri merkezinde meydana gelebilecek bir felaket anında ağ bağlantılarının kesilmemesi için gerçekleştirilen ağların taşınması işlemi sırasında performans kayıpları yaşanabilmektedir [77]. Ağdaki verimsiz çalışma veri merkezinde hizmet sağlayıcısı sözleşmesinde beyan edilen hizmet kalitesi ihlaline sebep olabilmektedir [79-81]. Sanal makine göç performansını iyileştirmek için ağ yapılarındaki göç işleminin de iyileştirilmesi gerekmektedir. Yaşanabilecek performans kayıplarının önüne geçmek için ağ yapılarının göç ettirilmesi sırasında en etkili şekilde ağ düğümlerinin taşınması gerekmektedir [82]. Bu zamanlayıcının tasarlanmasında öncelikle taşındığı sırada en büyük bağımsız alt ağ yapısındaki en az maliyete sahip düğümün taşınmasına öncelik verilmiştir [40].

Etkili taşıma işlemleri için zamanlayıcı tasarım çalışmaları yapılmıştır [41]. Bu zamanlayıcılar ile verimli bir taşıma ortamı gerçekleştirilebilmekte ve ağ göçü kolaylıkla yönetilebilmektedir. Aynı zamanda zamanlayıcının kontrol edilmesinde yeni bir ara katman geliştirilmesi sisteme bir ara katman daha eklenmesi çalışmaları yapılmıştır [83]. Bu kontrolcünün yazılım tanımlı ağlarla tasarlanmış dağıtık sistemlerde kullanılması mümkün olduğu gibi geniş alan ağı olarak tanımlanan (wide area network) sistemlerde de kullanılması mümkündür. Verimli ve performans kaybı yaşanmadan ağ göçü önemli olduğu gibi ortaya çıkan maliyetin de düşürülmesi önem arz etmektedir. Veri merkezlerinin harcadığı toplam güç göz önünde bulundurulduğunda, sanal makine göçü, toplam enerji kullanımını arttırmaktadır [84]. Bulut sistemlerinde sanal makina göçü sırasında fiziksel makinaların harcadığı enerjinin daha azaltılması konusunda zamanlayıcı çalışmaları yapılmıştır [85]. Ağ topolojilerinin taşınması sırasında da ortaya çıkacak enerji maliyetinin azaltılması veri merkezinin harcadığımı enerjiyi de azaltacak, doğa dostu bir taşıma işlemi gerçekleştirilir [86].

3.6. Sunucu Optimizasyonu

Sunucu optimizasyonu, veri merkezlerin verimli kullanımı sağlayan ve sürekli arge faaliyeti barındıran bir konudur. Verimli kullanım aşamalarında öncelikle bir kuruluşun ihtiyaç duyduğu fiziksel sunucu sayısının azaltılmasına başvurulur [87]. Sunucuların çok dağınık

kullanılması, üzerinde uygun olmayan veyahut makineleri yoracak iş yüklerinin olması, iş yüküne göre uygun bulunmayan yüksek işlemci kabiliyetini ve güç tüketimini kullandığı için yetersiz kaynak tahsisi nedeniyle veri merkezleri maliyeti artabilir. Enerji tüketimine yönelik en iyi iyileştirme, bulut ortamındaki uygulama optimizasyonu ile sağlanmaktadır [56]. Kaynak kullanımı, konsolide iş yükü performansı ve sistem güç tüketimi arasında sıkı ilişkiler var. Örneğin, işletilmeyen sunucu güç tüketimi, ancak açık, tam yüklü sunucunun yaklaşık % 70'idir [88]. Bulut veri merkezlerinde, sanallaştırma teknikleri kullanılarak, sanal makine sayısının etkin bir şekilde sanal sunucuya eşlenmesi; sunuculara yüksek kullanım sağlar ve çalışma süresi güç tüketiminde potansiyel olarak azaltılır. Sunucu optimizasyonunda sanal makine geçişi uygulasada, birkaç çalışma, sunucu birleştirmenin, kaynak kullanımı, özel ilgi alanlarına sahip sanal makineleri bir araya getirme yoluyla güç yönetimi, trafiğe açık iletişimde ortak iletişim kurmak için yüksek trafiği bir araya getirme yoluyla güç yönetimini en üst düzeye çıkarmak için özel olarak yararlanıldığını göstermektedir.

Aynı fiziksel makinelerin yöneticiler tarafından sistem bakımı için sanal makinelerin optimize edilmesi gerekmektedir [89]. Sanal makine geçişi, yük dengeleme, hata toleransı ve kullanılabilirlik gibi çeşitli nedenlerle gerçekleşir, ancak sunucu konsolidasyonu sanal makine geçişinden ayrılır. Bununla birlikte, konsolidasyon programı, ilgili fiziksel sunucuların kullanım modellerini aynı fiziksel sunucuda birlikte barındırmak için doğru bir şekilde ele alınmalıdır. Hizmet seviyesi anlaşması ihlali oranında performansın düşmesine neden yol açar. Bu bağlamda, geçiş süresi ve belirli kaynaklar üzerinde uygulama ilgisi göz önüne alınmadan, kullanılan çeşitli optimizasyon algoritmaları, sunucu konsolidasyon tekniğinin yararını geçersiz kılacaktır. Kullanıcı işlem tutarı ve hafıza ihtiyaçları açısından gereksinimleri tanımlayan kaynak aracısına bir talepte bulunur. İstenilen kaynaklar yeterliyse, kullanıcının işi kabul edilir ve birincil yerleşim gerçekleşir. Daha sonra, kullanım izleme çalışmaları düzenli aralıklarla kaynak kullanımını kontrol ediyor ve gerekliyse, sunucu birleştirmeyi tetiklemektedir [57]. Aslına bakılırsa, literatürdeki çerçevelerin çoğu, az ve çok kullanılmış sunucular için üst ve alt eşik değerlerini sırasıyla belirler [51].

Enerji verimli bir sunucu optimizasyonu şeması için güç tüketimini ve etkin kaynak tahsisini yönetmek bulut veri merkezleri için önem taşımaktadır. Matematiksel formülasyon ile güç maliyetinin düşürülmesi amaçlanmaktadır. Verimli sunucu birleştirmeden yararlanma temel alınmaktadır. Ayrıca veri merkezi sorunları dikkate alınmıştır. gelen ve başarısız sanal

makinelerin haritalandırılması, toplamın azaltılması gibi sanal makine geçişlerinin sayısı ve çalışan sunucunun birleştirilmesi ve iş yükleri olarak sıralanabilir. Kaynak tahsisi kısmı CPU, RAM ve ağ alır. Çok amaçlı problemlerin çözümü için sezgisel algoritma geliştirdiler. Sonuç üstünlüğünü en fazla tanınmış Best Fit Consolidation (BFC) yaklaşımına karşı gösterilmiştir [90]. Sunucu konsolidasyon alanındaki araştırmaların çoğu verimsiz kaynak tahsisi, kaynak kullanımı iyileştirme ve yeşil veri merkezi endişeleri, ancak zorunlu özel olarak sistem bakımı, yakınlığı ve çekişmeli olan özellikle çok katmanlı web uygulamalarında sanal makinelerin kaynakları zorunlu sunucu optimizasyonu programları yer almaktadır.

Literatürde önerilen optimizasyon planları kaynakların farkında olunması, göçün kontrollünün sağlanması, göçmenlik süresinin azaltılmasıolarak sıralanabilir. Ayrıca kaynakların yeniden tahsis edilmesine dair literatürde çalışmalar yapılmıştır. Fiziksel makinelerde yeni eklenen sanal makineler için yeni yer tahsis edilmesi üzerinde çalışmaktadır [87], [91]. Yoğun kaynak tüketen gereksiz sanal makine geçişinin önlenmesi hedeflenmektedir. Ayrıca sunucularda çalışan voltaj-frekans çiftlerini ayarlayabilir iş yükü tahminine göre zamansız ya da agresif bir şekilde takılma sunucu konsolidasyon şeması veya çalışan voltaj frekansını artırabilir için gereksiz sanal makine geçişlerini engellemek için hizmet seviyesi anlaşması ihlalini önlemek hedeflenmektedir. Çizelge 3.2‘de yapılan çalışmalar, sebep ve optimizasyonun nasıl tetiklendini özetleyen çizelge yer almaktadır.

Çizelge 3.2. Sunucu optimizasyonu şemaları ve sanal makine göçü

Çalışma	Sebep	Görüntüleme	Optimizasyon tetikleme
Mazumdar ve Pranzo (2017) [90]	CPU / Bellek / Bant Genişliği Kullanımı	Var	Sezgisel
Jian-ping ve ark. (2014) [56]	Güç Yönetimi / CPU Kullanımı	Var	Sezgisel
Rao ve Thilagam (2015) [91]	CPU / Bellek / Bant Genişliği Kullanımı	Var	Sezgisel
Wang ve Wang (2014) [92]	CPU / Bellek / Bant Genişliği Kullanımı	Var	Veri geçmişi

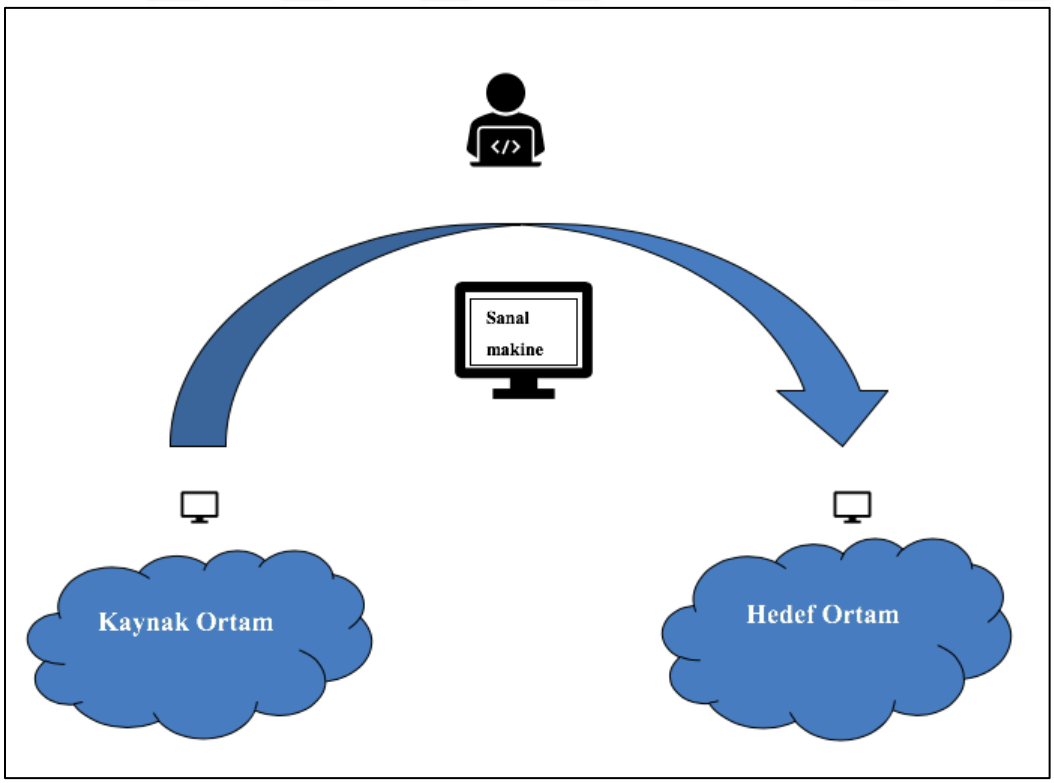
Çizelge 3.2. (devamı) Sunucu optimizasyonu şemaları ve sanal makine göçü

Sudevalayam ve Kulkarni (2013) [93]	Kaynak benzeşimi	Var	Sezgisel
Perumal ve Subbiah (2014) [94]	CPU / Bellek / Bant Genişliği Kullanımı	Var	Sezgisel
Ferreto ve ark (2011) [87]	Güç Yönetimi / Bellek Kullanımı	Yok	Sezgisel
Al-Azzoni (2015) [95]	Kaynak Kullanımı / Güç Yönetimi	-	Sezgisel
Yang ve ark (2014) [60]	Güç Yönetimi / Kaynak Kullanımı	Var	Veri geçmişi
Li ve ark (2016) [96]	İletişim Maliyeti (Trafik içi maliyet) / Kaynak Kullanımı	Var	Veri geçmişi
Sedaghat ve ark. (2016) [97]	Sistem bakımı	Var	Sezgisel / Veri geçmişi

4. GEREÇ VE YÖNTEM

4.1. Problemin Tanımı

Veri merkezlerinde yaşanabilecek elektrik kesintileri, bakım-onarım işlemleri, kullanıclan donanımların değişimi gibi sistemsel olaylar üzerinde bulunan sanal makinelerin ve sanal kaynakların göç ettirilmesini gerektirmektedir. Göç işlemi sırasında ağdaki yoğunluğun artması, fiziksel sunucuların ve ağ elemanlarının yüksek performans ve CPU yoğunluklu çalışması ile artan enerji tüketimi sanal makine ve ağ taşınması sırasında problemler oluşturmaktadır. Sanal makinelerin taşınması sırasında; sanal makinenin erişilemez olması, toplam göç süresi ve sanal makinenin büyüklüğü canlı göç işleminin bir maliyet oluşturmasına sebep olmaktadır. Aynı zamanda sistemde hizmet verilen kişiler, kurumlar ya da projeler ise çeşitli önceliklendirmelere sahip olup hizmet seviyesi sözleşmesi ile göç sırasındaki kesinti sürelerinin uzunluğu kadar daha fazla maliyete sebep olmaktadır. Şekil 4.1'de bu tez kapsamında temel alınan bir bulut sisteminden diğer bulut sistemine göç gösterilmektedir.



Şekil 4.1. Bulut sistemleri arası göç

Mevcut tanımlanan probleme sezgisel yaklaşımlı bir algoritma ile çözüm yolu önerisi yapılmıştır. Problem; göç sırasında, göç ettirilmesi gereken sanal makinelerin listesi üzerinden olası yollar tahmin edilerek en az maliyet ve en iyi performans ile işlemin gerçekleştirilmesi hedeflenmektedir.

Problemin çözümünde; probleme özgü çözüm sunan sezgisel algoritmalar kullanılacaktır. Sezgisel yöntem ile karmaşık ve değişken sayısı fazla olan probleme kesin doğru olmamakla birlikte en optimum ve tatmin edici sonuçların elde edilmesi hedeflenmektedir.

4.2. Kullanılan Algoritmalar

4.2.1. Minimum Spanning Tree

Mevcut bir topolojinin taşınmasında en yakın bölgenin belirlenmesi [105] "Minimum Spanning Tree (MST)" [106-107] algoritması ile gerçekleştirilebilmektedir. Test sırasında MST algoritmalarından Kruskal Algoritması kullanılmıştır [108-109].

4.2.2. Maximal Independet Set

Mevcut bir topolojiye bağlı alt topolojilerden en bağımsız setin seçilmesi [103-104] "Maximal Independent Set (MIS)" [83] ile gerçekleştirilebilir.

4.2.3. Dijkstra Algoritması

Dijkstra'nın algoritması, kaynaktan minimum mesafeye sahip bir dizi düğüm oluşturarak, tek bir kaynak düğümden en kısa yol ağacını bulur [116].

4.2.4. A* Algoritması

Basit bir sezgisel olarak da adlandırılan bir sezgisel işlev, hangi dalın izleneceğine karar vermek için mevcut bilgilere dayanarak her dallanma aşamasında arama algoritmalarındaki alternatifleri sıralayan bir işlevdir. A* algoritması yapı olarak sezgisel bir algoritmadır. Algoritma hesaplanırken mesafe hesaplamada Eş. 4.1'teki gibi kullanılmaktadır.

$$f(n) = g(n) + h(n) \quad (4.1)$$

f(n); hesaplama yapan sezgisel (heuristic) fonksiyondur.

g(n); Başlangıç düğümünden mevcut düğüme kadar gelmenin maliyeti

h(n) = Mevcut düğümden hedef düğüme varmak için tahmin edilen mesafe

Denklem 3'de yer alan f(n) fonksiyonunun sezgisel olma sebebi, bu fonksiyon içerisinde bulunan ve tahmine dayalı olan h(n) sezgisel fonksiyonudur.

Algoritma yukarıdaki toplama işlemini kullanan oldukça basit bir yapıya sahiptir. Veri yapısı olarak bir öncelik sırası kullanan algoritmada en öncelikli olan düğüm f(n) değeri en az olan düğümdür. Algoritma her adımda en önemli düğümü ve düğümü sıradan çıkarır. Gidilen bu düğüme göre komşu olan bütün düğümlerin değerleri güncellenir ve düğüme ulaşmanın bir maliyeti vardır ve f(n) fonksiyonu içerisinde bu değer yer almaktadır.

4.2.5. Özelleştirilmiş A*

Standart A* algoritması içerisinde bulunan h(n) sezgisel fonksiyonu 'None' olarak değerlendirildiğinde en kısa yolu bulan Dijkstra algoritması gibi çalışır. Ancak istenildiği gibi özelleştirilmesi mümkündür. Tez kapsamında özelleştirilmiş A* algoritmasında projelerin sahip oldukları öncelikler göz onunda bulundurulmuştur. Projenin önceliği HSA'daki yaptırımlardan dolayı maliyeti artırmaktadır. Proje bazında yapılan taşıma işleminde, projeye ait sanal makinelerin üzerinde bulunduğu fiziksel makinelerde bir sorun yaşanması durumunda projenin sanal kaynaklarının en kısa ve en sağlıklı şekilde çalışan fiziksel sunuculara aktarılması gerekmektedir. Böylelikle toplam göç süresi azaltılması hedeflenmekte olup, aynı zamanda göç süresi içerisinde yaşanan makinenin erişilemez durumda olduğu kesinti süresinin en aza indirgenmesi hedeflenmektedir. Öncelikli makinelerin bir döngü içerisinde kontrol edilerek öncelikli olarak mevcut katsayılarına göre taşınmaları, projelerin rasgele ya da sırayla taşınmasına göre maliyeti düşürmektedir.

4.3. Kullanılan Performans Metrikleri

Sanal makine göçü sırasında en çok sık rastlanan problem taşınan sanal makineye erişimin kesilmesidir. Erişimi kesilen sanal makineden hizmet verilememekte ve işlemler gerçekleştirilemez olmaktadır. Sanal makineye olan erişilemez durumun en aza indirgenmesi göç işleminde performans metriği olarak değerlendirilmektedir. Aynı zamanda

hizmet seviyesi sözleşmesinde beyan edilen erişilememezlik süresini geçmemesi gerekmektedir.

Bu göç tekniklerinin performansını değerlendirmek için kullanılır. Değerlendirmede kullanılan toplam göç süresi, kapalı kalma süresi ve aktarılan toplam veri olmak üzere üç ana ölçüt bulunmaktadır.

- Toplam göç süresi: Geçiş işlemine başlama ile sanal makinenin artık kullanılabilir olmadığı zaman arasındaki süredir. Göç teknikleri bu süreyi mümkün olduğunca azaltmayı amaçlamaktadır.
- Kapalı Kalma Süresi: Sanal makinenin yürütmesini askıya alma nedeniyle sanal makinenin yanıt vermediği süre olarak tanımlanır. Göç araştırmaları ayrıca, kesinti süresini kısaltarak göç sürecini daha şeffaf hale getirerek göçün performansını artırmaya çalışır.
- Aktarılan toplam veri: Göç süresince aktarılan toplam veri olarak tanımlanabilir. Veri azaldıkça ağ tüketimi azalır.

Aynı zamanda Çizelge 4.1'de yer aldığı gibi sanal makine ve ağ göç işlemlerinde performans değerlendirmesinde kullanılan metrikler literatürde genel geçer olarak kabul edilmektedir [98].

Çizelge 4.1. Literatürde kullanılan sanal makine göç metrikleri

Metrik	Özellik tanımı	Ölçüm yöntemleri
Kesinti	Geçiş yapan sanal makinenin yürütmesinin durdurulduğu süre	Sanal makine, bir taşıma sırasında aksama süresi paketlerin kaybolduğu zamana eşittir.
Toplam göç zamanı	Baştan sona göç sırasında geçen süre	Bir geçiş komutunun süresidir.

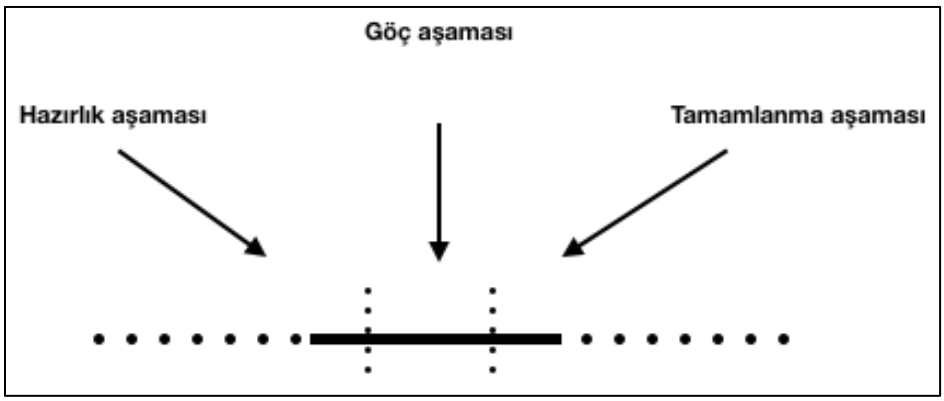
Çizelge 4.1. (devamı) Literatürde kullanılan sanal makine göç metrikleri

Taşınan veri miktarı	Tüm göç döneminde aktarılan toplam taşınan veri miktarı	Bir taşıma için ayrılmış TCP bağlantı noktasından aktarılan toplam veri miktarıdır.
Göç yükü	Geçiş yapmak için kullanılan fazladan makine kaynakları	Bir iş yükünün göç durumundaki performansını göç dışı durum ile karşılaştırılır veri yükü, reddedilen oranı ile temsil edilmektedir.

Sezgisel algoritmalarının baskınlık, tutarlılık ve kabul edilebilir özellikleri ile önerilen yöntem ile performanslı bir çözüm sağlanması hedeflenmektedir.

Göç işlemi tamamlandıktan sonra taşınan makinenin yeni olduğu yerde de sorunsuz bir şekilde çalışması önem taşımaktadır. Sezgisel yöntem ile taşıma işleminin performanslı ve hızlı bir şekilde gerçekleştirilmesinin yanı sıra sanal makinenin taşıma sonrasında sorunsuz ve performanslı bir şekilde taşınması beklenmektedir.

4.4. Maliyet Fonksiyonu



Şekil 4.2. Sanal ağ göçünde zaman gösterimi

Sanal bir ağın taşınması Şekil 4.2'deki gibi hazırlık aşaması, göç aşaması ve tamamlanma aşaması olmak üzere üç kısımdan oluşmaktadır. Hazırlık aşamasında taşınacak sanal makinelerin belirlenmesi tünel linklerin ve geçici bağlantıların oluşturulması gerçekleştirilir.

Göç aşaması ise kurulan bağlantılar üzerinden seçilen düğümlerdeki verilerin kopyalanması işlemleri yer almaktadır.

Tamamlanma aşamasında ise sanal makineye ait son düzenlemeler yapılarak bir taşıma işleminde tek bir düğüm ve düğümün komşuları ile bağlantılarını sağlayan kenarlıkları taşınmaktadır.

İlk hazırlık evresi 3.2 saniye sürerken sanal kaynakların tamamen taşınması için her taşıma döngüsünde için sistemin tekrar tekrar analiz edilmesi gerekmektedir. Her bir analiz döngüsü yaklaşık 3,5 saniye sürmektedir [41,82]. Aynı zamanda tamamlanma aşaması da sanal makine linklerinin ve floating ip linklerinin güncellenmesini içermekte olup 4.2 saniye sürmektedir.

Her bir düğümün taşınması durumu "i" olarak değerlendirilmektedir. S toplam taşınacak sanal makine sayısını göstermektedir. g() fonksiyonu göç ettirilecek sanal makineyi göstermektedir. Her durum için t^d olarak belirlenen zaman çarpanı (3,5 saniye) taşınan sanal makinenin iteratif bir şekilde taşınan RAM büyüklüğne bağlı olarak değişen taşıma döngüsü g(i) ile ifade edilmekte olup buna bağlı geçen süre; göç aşamasındaki göç süresi, Eş. 4.2'de hesaplanmaktadır.

$$t^d \sum_{i=0}^{S-1} g(i) \tag{4.2}$$

Ağdaki mevcut düğümlerin taşınması işlemi sırasında ilk başta yapılan hazırlık aşamasında geçen süre t^h olarak değerlendirilmekte ve göç aşamasındaki toplam geçen süre olan t^g'ye eklenmekte ve t^t ise tamamlanma süresini göstermektedir. Analiz edilen sistemin göç ettirilmesinde geçen toplam süre Eş. 4.3'de hesaplanmaktadır.

$$\sum_{i=0}^{S-1}(t^h(i) + t^g(i) + t^t(i)) \tag{4.3}$$

Ayrıca göç aşaması sırasında oluşan sanal makineye erişim kesinti süresi t^k büyük önem taşımaktadır. Çünkü maliyet fonksiyou ile toplam göç süresinin ve yaşanan kesinti miktarının en aza indirgenmesi hedeflenmektedir. Toplam kesinti süresi Eş. 4.4'de gösterilmektedir.

$$\sum_{i=0}^{S-1}(t^k(i)) \quad (4.4)$$

Zamanın yanı sıra maliyet fonksiyonun hesaplanmasında sanal makinelerin bulunduğu projelerin öncelikleri göz önünde bulundurulmuştur [99]. Taşıma işleminde düğümün belirlenmesinde kullanılan algoritmaya göre ağırlıklarında hesaba katıldığı maliyet fonksiyonu değişiklik göstermektedir.

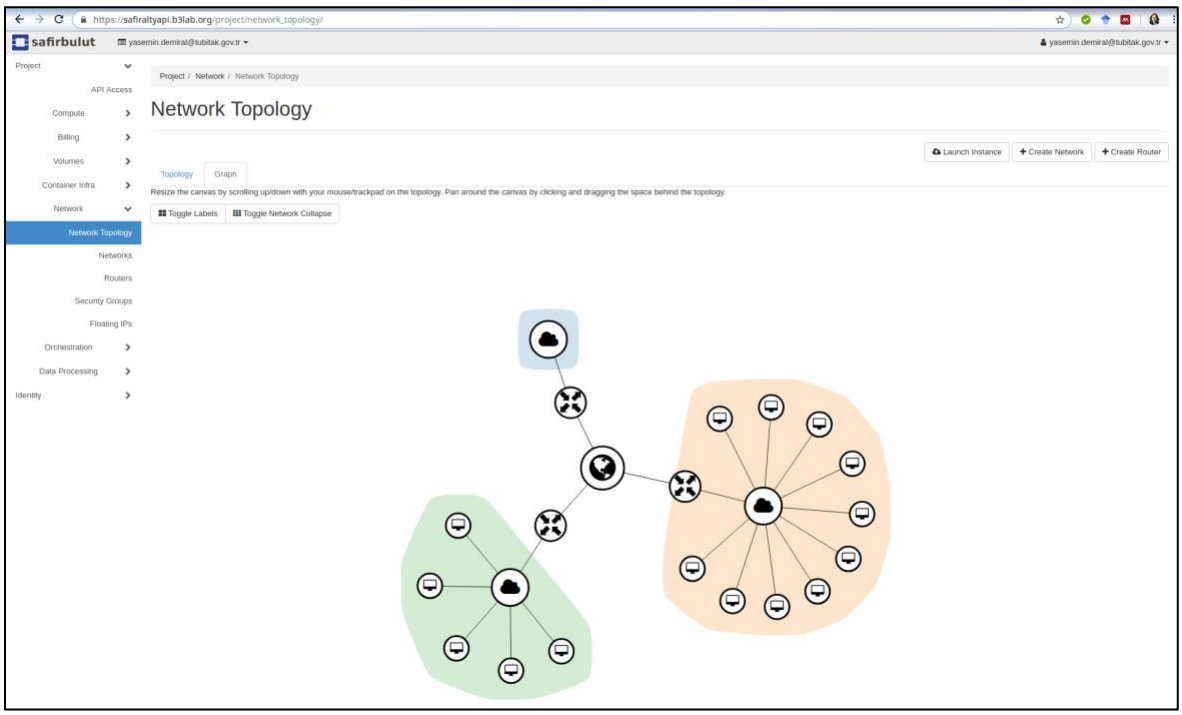
Taşınacak olan düğümün belirlenmesinde taşınacak makinenin öncelik sırasında maliyetin hesaplanmasında önemli bir rol oynamaktadır. Google Cloud [100] gibi büyük sistemlerde ise sanal makine ilk oluşturulurken olası göç işlemlerinde sanal makinenin durumu ile ilgili sorular cevaplandırılması istenmektedir. Sanal makine kritik ise taşınmayacak fiziksel makinelerde oluşturulması veyahut izole sadece tek bir fiziksel makine kullanılması durumları kullanıcılardan alınan bilgilere göre optimize edilmektedir. Prototip veri merkezlerinde [101] sanal makine taşıma işlemi gerçekleştirilmeden önce kullanıcılara bir bilgilendirme yapılmaktadır. Kullanıcıların mevcut hesaplarında bulunan sanal makineleri ile ilgili taşıma işleminde nasıl bir aksiyon alınması istediklerine dair geri dönüş beklenmektedir. Kullanıcılar makinelerini kullanmıyor ise silinmesini talep edilebilir, yeni oluşturulacak ortama kendileri taşıyabilir veyahut kendileri verileri yedekleyip otomatik olarak taşınmasını isteyebilir. Her bir seçeneğin veri merkezine ve yöneticilere belirli getirdiği sorumlulukları ve maliyetleri bulunmaktadır.

Kullanıcılardan gelen taleplere göre bir önceliklendirme sırası ile veri merkezi tarafından verilen hizmetin kalitesi kapsamında yapılan anlaşmanın gereklilikleri yerine getirilmektedir. Hizmet seviyesi anlaşması ile göç maliyeti hesaplamaları gerçekleştirilmektedir. Kullanılan test ortamlarındaki sanal makinelerin öncelik sınıflandırması aşağıdaki Çizelge 4.2'de yer almaktadır.

Çizelge 4.2. Sanal makinelerin öncelik derecelendirmesi

Önem	Öncelik
Kritik	1
Çok Fazla	2
Fazla	3
Az fazla	4
Standart	5

Sanal makineler üzerlerinde bulundukları fiziksel makineye özel bir ağ ile bağlı olup, OpenStack'te her bir proje kendi özel ağına sahiptir. Dış ağa çıkmadan bir proje içerinde tanımlanmış özel ağa ait sanal makineler kendi aralarında haberleşebilmektedirler. Bir projeye ait ağ topolojisi Resim 4.1'de yer almaktadır.



Resim 4.1. Projelere ait ağ topolojisi

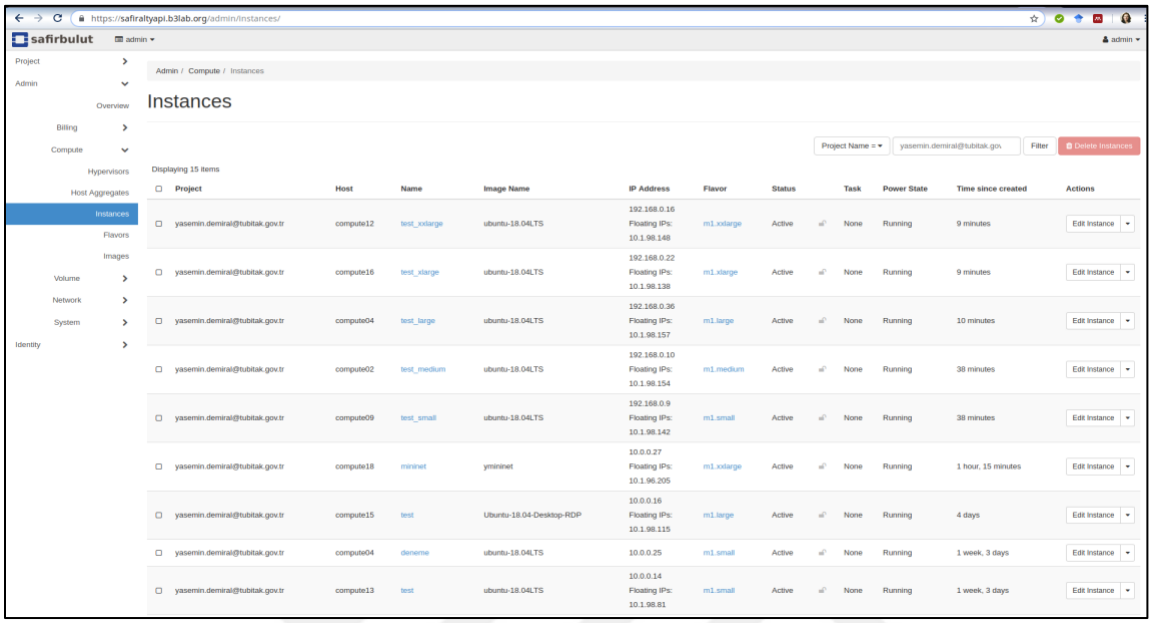
Test için kullanılan sanal makinelerin büyüklükleri birbirinden farklı olup, en sık kullanılan flavorları içermektedir. Flavorlara ait sanal makinelerin sahip oldukları sanal kaynak büyüklükleri Çizelge 4.3'deki gibidir.

Çizelge 4.3. Sanal makinelerin sanal kaynak bilgileri

Flavor Adı	vCPU	RAM (GB)	Disk (GB)
m1.small	1	2	20
m1.medium	2	4	40
m1.large	4	8	80
m1.xlarge	8	16	80
m1.xxlarge	16	32	160

Sanal makinenin göç ettirilmesi sırasında bulunduğu fiziksel makinenin değişimi söz konusudur. Bir projeye ait sanal makineler sistemde var olan fiziksel sunucular üzerinde

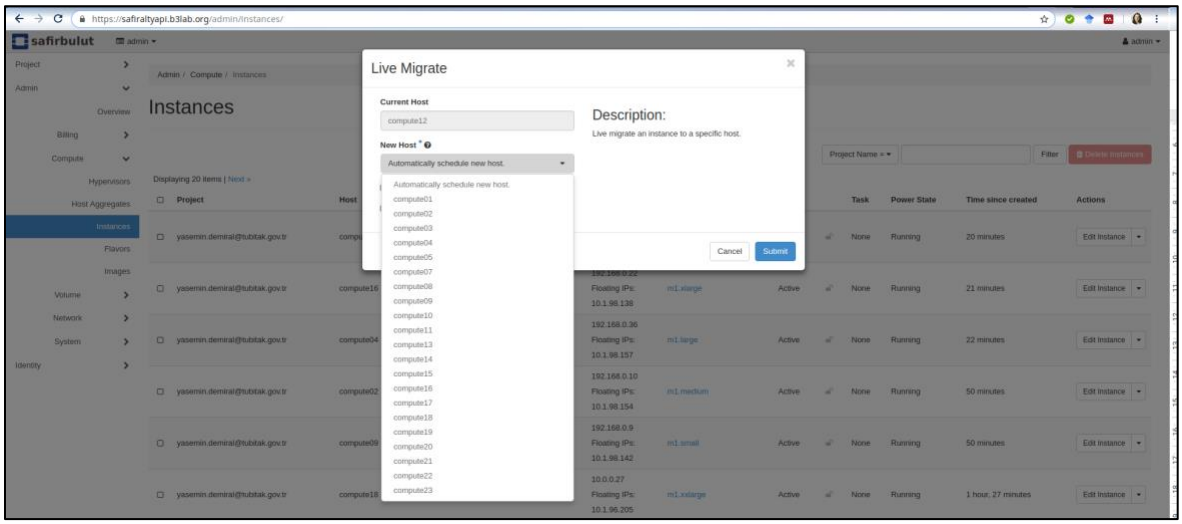
dağılmış bir şekilde bulunmaktadır. Sanal makinelerin bulunduğu fiziksel sunucular sistemde bulunan Admin hesabı ile Resim 4.2'deki gibi görüntülenebilmektedir.



Resim 4.2. Sanal makinelerin dağılımı

4.5. Canlı Ortamda Sanal Makine Göçü

Sanal makine göç işlemi sistem yöneticisi tarafından tek tek el ile veyahut bir program aracılığıyla gerçekleştirilebilir. Göç işlemi proje bazlı yapılabildiği gibi fiziksel makine bazlı da gerçekleştirilebilir. Bu tez çalışmasında proje bazlı göç işlemi projelerin öncelik katsayısı gözetilerek yapılmaktadır. Sanal makine göç işleminde göç edilecek hedef kaynak otomatik belirlenebildiği gibi, istenilen fiziksel sunucuda Resim 4.3'deki gibi seçilebilmektedir.



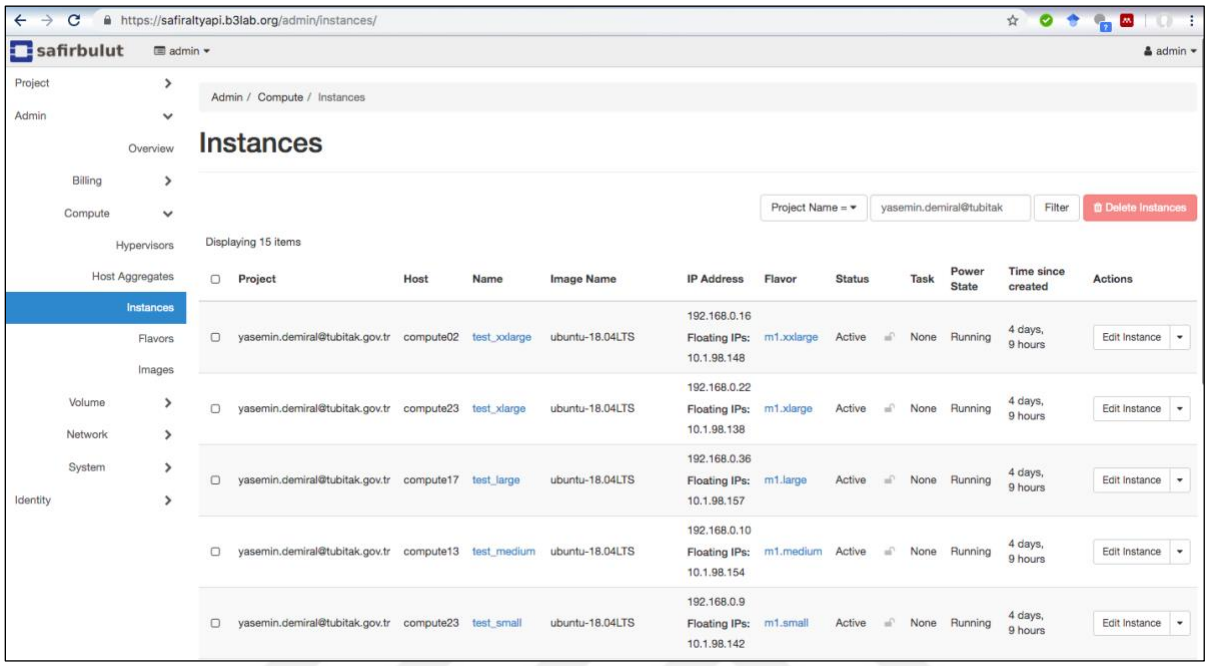
Resim 4.3. Sanal makine göçü sistem yöneticisi ekranı

Sanal makinelerin içerisinde %60 oranında RAM performanı kullanan iş yüklerinin olduğu varsayılmaktadır. Bütün makinelerde standart iş yükü kullanılarak bu parametrenin sabit tutulması hedeflenmektedir. Sanal makinelerin büyüklüğününe göre elde edilen gerçek toplam göç süreleri ve göç sırasındaki kesinti süreleri veri merkezinde yapılan testlerle ilgili elde edilen süre bilgileri Çizelge 4.4'de yer almaktadır.

Çizelge 4.4. Sanal makinelerin göç süreleri

Sanal Makine Adı	Sanal Makine RAM büyüklüğü (GB)	Taşınacak RAM büyüklüğü (%60) (MB)	Toplam Göç Süresi (s)	Göç hazırlık için geçen süre (s)	Göç sırasındaki kesinti süresi (s)	Göç sonu hazırlık süresi (s)
test_small	2	1228	30	3,2	3	4,2
test_medium	4	2458	48	3,2	4,2	4,2
test_large	8	4916	90	3,2	7,4	4,2
test_xlarge	16	9830	160	3,2	12,6	4,2
test_xxlarge	32	18432	300	3,2	20,4	4,2

Resim 4.4'de yer aldığı gibi sanal makinelerin bulunduğu fiziksel sonuçlar canlı sanal makine göçü ile değiştirilmiştir.



Resim 4.4. Canlı göç sonucunda sanal makinelerin fiziksel sunucu bilgileri

Göç süresi hesaplanırken toplam göç süresinden; hazırlık ve tamamlanma aşamaları çıkartılarak göç işlemindeki veri aktarma süresi maliyette etkin olarak kullanılır. Elde edilen süre bilgileri ile toplam aktarılan veri miktarı (v(i)), göç süresi ve kesinti süresi kullanılarak maliyet Eş. 4.5'deki gibi hesaplanmaktadır.

$$\sum_{i=0}^{S-1}(v(i)\ t^{g}(i)t^{k}(i)) \tag{4.5}$$

Çizelge 4.5. Gerçek ortamda elde edilen maliyet sonuçları

Sanal makine adı	Veri miktarı (MB)	Göç Süresi (s)	Kesinti Süresi (s)	Maliyet (MBs^2)
test_small	1228	22,6	3	83,258
test_medium	2458	40,6	4,2	419,138
test_large	4916	82,6	7,4	3004,8
test_xlarge	9830	152,6	12,6	18900,73
test_xxlarge	18432	292,6	20,4	110021,34
Toplam	36864	628	47,6	132429,824

Yapılan canlı ortamda testlerde 5 farklı boyutta sanal makine kullanılmıştır. Sanal makinenin boyutları değişmesi ile değişen göç süreleri ve maliyet Çizelge 4.5'de yer almaktadır. Elde edilen sonuçların toplamı daha sonrasında simülasyon ortamında yapılacak testler için temel alınacaktır. Bu şekilde 5 makine için yapılan test ve sonucunda hesaplanan maliyette hiçbir kısa yol ve bağımlılık algoritması kullanılmadan yapılan göç işlemi maliyeti hesaplanmaktadır.

Bir projede en az 5 farklı büyüklükte sanal makine olduğu varsayılmıştır. Aynı zamanda genişletilen test ortamında makine büyüklüklerinin dengeli bir dağılım gösterdiği varsayılmaktadır. Simülasyon ortamında oluşturulan yapılarda gerçek ortamdaki maliyetler elde edildikten sonra genişleterek algoritmalar kullanılmış ve özelleştirmeler gerçekleştirilmiştir. En kısa yol ve mevcut bağımlılıklar göz onunda bulundurularak projelerin öncelik katsayıları işleme katılmış, ortaya çıkan maliyetin düşürülmesi hedeflenmiştir.

5. DENEYSEL DEĞERLENDİRME

5.1. Performans Ölçüm Metrikleri

Sanal makine göçü deneylerinde en önemli kullanılan metrikler CPU, RAM, ağda yaşanan gecikme olarak sıralanabilir. CPU (Central Processing Unit); merkezi işlem birimi diğer bilgisayar aygıtlarından gelen verileri matematiksel işlemler yardımı ile işler, hesaplamaları yapar sonuca ulaşır ve sonucu gerekli yerlere gönderir, bilgisayarın içindeki tüm bilgi akışını kontrol eder ve yönetir [110]. RAM; rasgele erişilebilir bellek, bilgisayar üzerinde çalışan programların açıldığı kısımdır. Bir uygulama başlatıldığında bilgisayar üzerindeki bellekte herhangi bir yere yerleştirilerek çalıştırılır. Veriler düzensiz ve rasgele yazılır [111]. Ağ gecikmesi (network delay); bir bilgisayar ağının veya telekomünikasyon ağının önemli bir tasarım ve performans özelliğidir. Bir ağın gecikmesi, ağ üzerinde bir düğümden veya uç noktadan diğerine bir veri miktarının ne kadar süreceğini belirler. Genellikle, katların veya saniye kesirleriyle ölçülür [112].

5.2. Sanal Ağ Performansında Hipervizör Etkisinin İncelenmesi

Göç öncesi kaynakların durumunu incelemek için hipervizör etkisinin gözlemlenmesi bir çalışma yapılmıştır. Ayrıca iş yükünün performansa olan etkisinin gözlenmiştir. Test ortamının oluşturulması bulut platformu olarak OpenStack kullanılmıştır. Tercih edilen hipervizörler KVM, Qemu, Hyper-V ve VMWARE ESXi'dir. Her hipervizörün üzerinde aynı özelliklere sahip sanal makineler oluşturulmuş, aynı sanal ağ bilgileri eklenmiştir. Kullanılan sanal makinelerin bilgileri Çizelge 5.1'de yer almaktadır.

Çizelge 5.1. Sanal makinelerin özellikleri

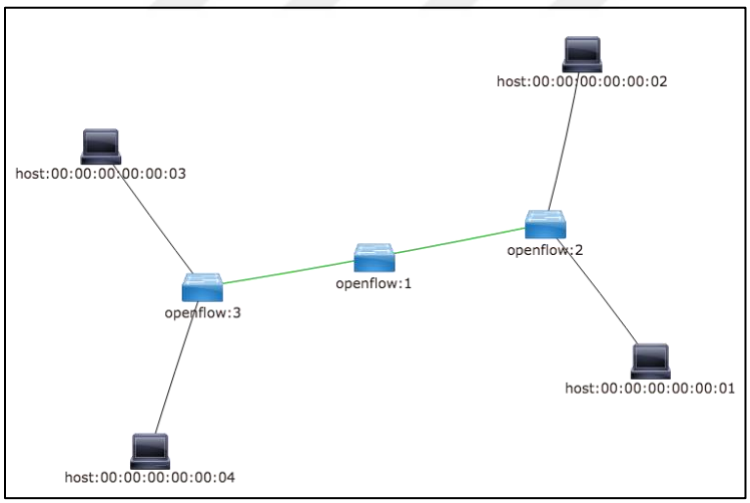
Özellik	Çeşidi
vCPU	4GB
RAM	8GB
Disk	80GB
İşletim Sistemi	Ubuntu 16.04 Server LTS

Ağ performans testlerinin gerçekleştirilmesi için "iperf" kullanılmıştır. Iperf iki bilgisayar arasındaki ağ hızını ve bant genişliğini tespit etmek için kullanılmaktadır. Sunucu-istemci mantığı ile çalışmaktadır [113].

Sanallaştırma bulunmayan bir masaüstü Ubuntu işletim sistemine sahip bir makine ile KVM sanallaştırma uygulanan bir Ubuntu işletim sistemine sahip sanal makinede, Openstack'te oluşturulan ve sunucu olarak belirlenen sanal makine ile aralarında test gerçekleştirilmiştir. 3 farklı kategoride gerçekleştirilen testlerde kullanılan "-p", "-l" ve "–w" değişkenlerindeki değerlerin ağ performansında gözle görünür düzeyde değişikliklere sebep olmadığı gözlemlenmiştir. Değişen büyüklüklerin masaüstü bilgisayar ya da daha büyük sunucu olması kullanılan fiziksel kaynakların performanslarının sanal kaynaklara da yansıdığı gözlenmiştir. Farklı hipervizorlerin kullanılmasının sanal makine ve sanal ağ göçü maliyetini göz önünde bulundurulması gereken düzeyde etkilememektedir.

5.3. Küçük Sanal Yapılarda Göç

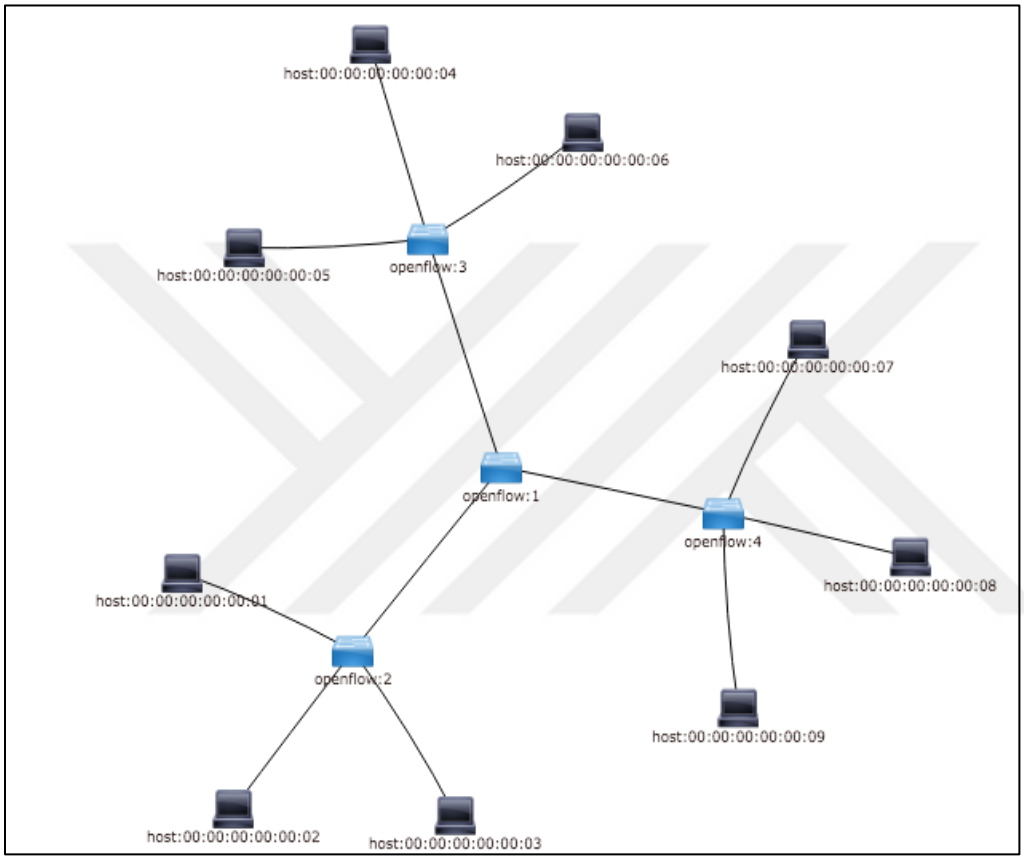
Sanal ağlarda göç deneyleri öncelikle literatürde geçen küçük sanal ağ topolojileri ile tanımlanan Mice akışa sahip bir topoloji kullanılmıştır [114]. Test ortamı için Python kütüphanelerinden birisi olan NetworkX [115] tercih edilmiştir. Test için farklı ağ topolojileri oluşturulmuştur.



Resim 5.1. Deneysel az düğümlü ağ topolojisi (OpenDayLight)

Ağ topolojileri temel olarak çok düğümlü topoloji ve az düğümlü topoloji olmak üzere ikiye ayrılmıştır. Her iki grupta ise düşük ağırlıklı, yüksek ağırlıklı ve ağırlıksız olarak üç sınıf bulunmaktadır. Çok düğümlü topoloji 10 düğümden, az düğümlü topoloji ise 4 düğümden oluşmaktadır.

Düşük ağırlıklı düğüm sınıfındaki ağırlıklar 0.3-2.8 aralığındadır. Az düğümlü ve düşük ağırlıklı olarak değerlendirilen topoloji Resim 5.1'de yer almaktadır. Topolojide yer alan düğüm sayısı 5'ın altında ve düşük ağırlıklıdır. Şekil 5.2'de Mininet ile oluşturulmuş ve görselleştirilmesi sFlow-RT ye yönlendirilmiş deneysel test ortamında kullanılan ve az ağ yüküne sahip ağ topolojisi yer almaktadır.



Resim 5.2. Deneysel çok düğümlü ağ topolojisi

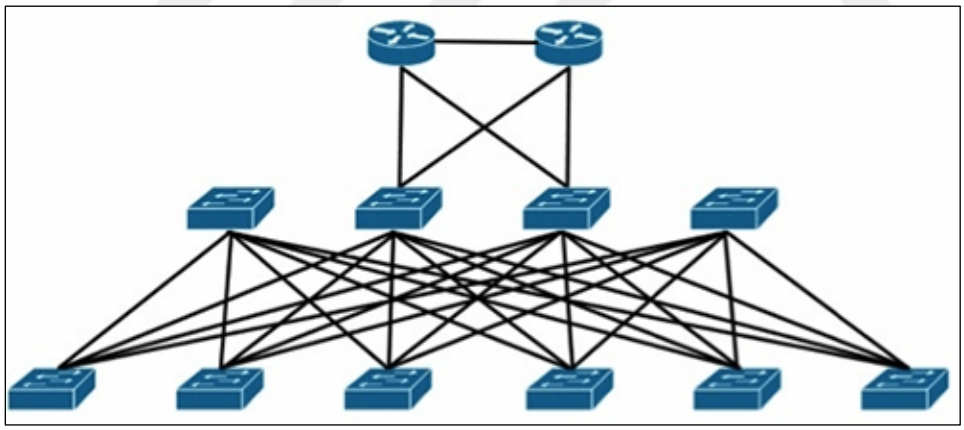
Çok düğümlü ve düşük ağırlıklı olarak değerlendirilen topoloji Resim 5.2'de yer almaktadır. Topolojide yer alan düğüm sayısı 5'ın üzerinde ve düşük ağırlıklıdır. Yüksek ağırlıklı sınıfı ise 2.9-5.8 aralığındadır. Ağırlıksız sınıfı ise NetworkX tarafından belirlendiği gibi varsayılan ağırlıkları olan 1 şeklinde hesaplanmıştır [41].

Yapılan çalışmada düşük ağırlıklı, yüksek ağırlıklı ve ağırlıksız olmak üzere 3 farklı ağ topolojisi üzerinde deneyler yapılmıştır. MIS algoritmasında en bağımsız setin seçilmesinde aynı seviyede bağımsız topolojiler elde edilirse ilk topolojisi rasgele seçilmektedir. Bu yüzden 10 kere aynı test çalıştırılmış ve ortalaması alınmıştır. MST'de ise benzer bir durumla karşılaşılmamış ancak testin doğrulanması adına testler birden fazla çalıştırılmıştır.

“Minimum Spanning Tree” ve “Maximum Independent Set” algoritmalarının kullanıldığı sanal ağ topolojisinin göç ettirilme deneyinde maliyet hesaplanmış olup, sistemdeki ağ topolojisinin analizi yapılarak belirli kurallar çerçevesinde ağ göçünün gerçekleştirilmesi halinde daha etkili sonuçlar alınabileceği sonucuna ulaşılmıştır. Deneysel ortamda yapılan testler ve sonuçları veri merkezi topolojisinde yapılacak ve sanal makinelerin göç ettirilmesinde etkili yöntemlerin geliştirilmesinde temel oluşturacak sonuçlar elde edilmiştir.

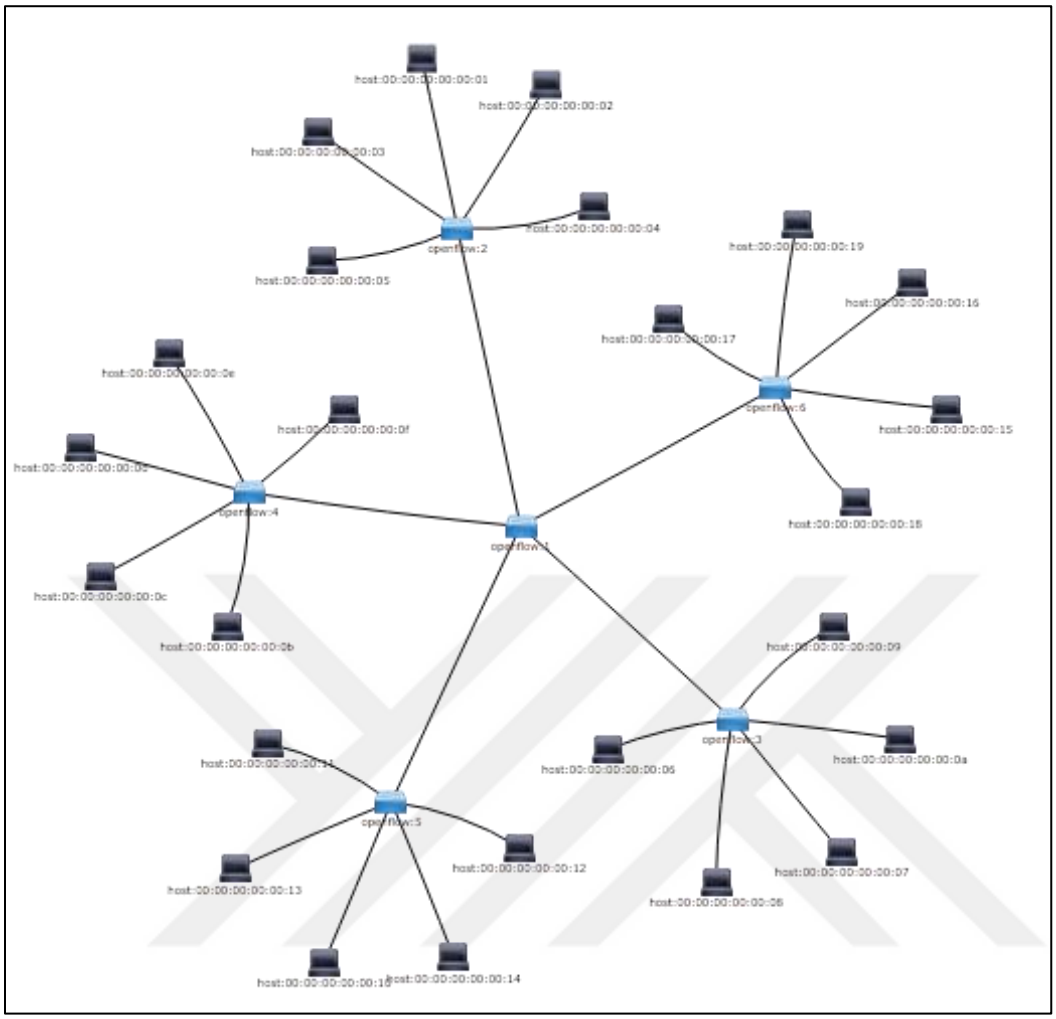
5.4. Veri Merkezi Kapsamındaki Sanal Yapılarda Göç

TÜBİTAK BİLGEM Bulut Bilişim ve Büyük Veri Merkezi Laboratuvarı [101] bünyesinde bulunan prototip veri merkezinde bulunan OpenStack Pike sürümünde olan bir bulut bilişim ortamı kullanılmaktadır. Prototip veri merkezinde yapılan deneyler ile daha büyük ve hizmet veren veri merkezleri için bir yöntem önerisi geliştirilmiştir. Prototip veri merkezinde kullanılan bulut sisteminde 23 tane hesaplama düğümü, 3 adet kontrolcü, 1 adet log sunucusu ve 1 adet depolama düğümü yer almaktadır. Omurga-uç mimarisine sahip olan veri merkezi yapısı Resim 5.3’de yer almaktadır.



Resim 5.3. Omurga-uç yapısı

Resim 5.4’da ise Mininet üzerinde yapılmış veri merkezi topolojisine yakın bir simülasyon topolojisi yer almaktadır.



Resim 5.4. Deneysel veri merkezi yapısı

Aynı benzer yapı NetworkX ortamında oluşturulmuştur. Ağ akışının düzenli ve kontrol edilebildiği varsayılmaktadır. Sanal makinenin taşınmasının oluşturduğu maliyetin hesaplanmasında çok yönlü ve değişkenli problemin tatmin edici çözümü için sezgisel yaklaşımlı taşıma yöntemleri tercih edilmektedir.

Ağ topolojilerinde ve çeşitli ilişkisel yollara sahip yapılarda kısa yol bulmak için kullanılan çeşitli algoritmalar kullanılmaktadır. Bir düğümden hedef düğüme en kısa hangi düğümler kullanılarak gidilebileceğini hesaplayan algoritma "en iyi yerleştirme (best fit)" algoritmasıdır.

Algoritma çalışma adımları takip edilerek hedefe varana kadar yani hedef düğümü öncelik sırasında en öne gelene kadar veya sırada düğüm kalmayana kadar tekrarlanmaktadır [102].

Taşınacak olan düğümün ilk olarak belirlenmesi ve sonrasında taşıma işleminin nasıl devam edeceği göç işleminin performans kaybı yaşanmadan gerçekleştirilmesi için büyük bir öneme sahiptir. Taşıma işlemi sırasında ağ yapısındaki trafiğin göz ardı edilebilecek miktarda etkilenmesi ve yaşanacak gecikmelerin asgari düzeye olması beklenmektedir.

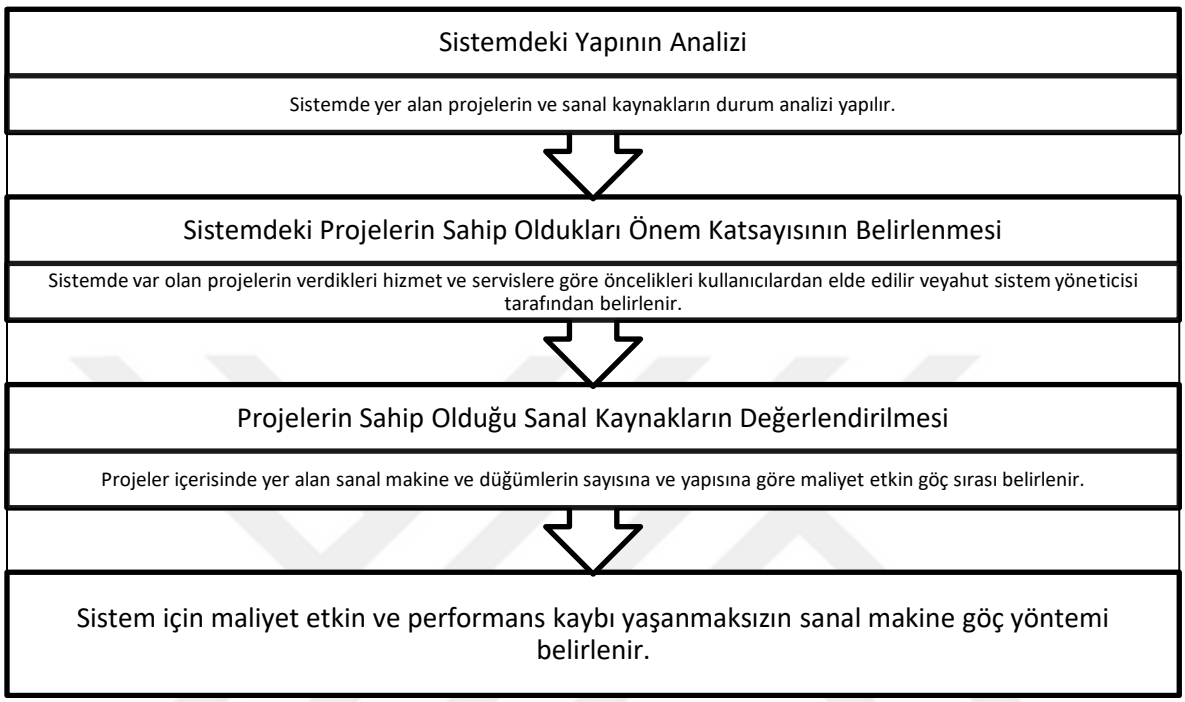
Çalışmada sezgisel A* algoritması ile önceliği belirlenen projeye ait sanal makineler kendi içerisinde gruplanarak kümeleme işlemi gerçekleştirilir. Alt kümelere ayrılan sanal makineler ise maliyetlerine ve sahip olunan en kısa yollara göre tekrar alt gruplara ayrılır. MIS ile en fazla bağımsız setin oluşturulabildiği alt gruplar seçilmektedir. Birden fazla aynı sayıda set halinde grup var ise rasgele grup seçimi yapılmaktadır. Seçilen alt gruplar bağımsız olmasından dolayı hep beraber bir set halinde taşınmaktadır. Düğümlerin aynı anda taşınması topolojisinin daha kısa sürede taşınmasını sağlamakta ve toplam maliyeti etkilemektedir.

MST algoritmasında ise ağırlıklı bir ağda her bir düğümü birbirine bağlayan kenarların maliyetini hesaplayarak bütün ağ dolaşıldığında o düğüme ait en yakın yolu vermektedir. Ayrıca en yakın en az ağırlığa sahip durumlarına göre taşıma işlemleri gerçekleştirilmektedir. Düğümler tek tek taşınmakta ve maliyet hesaplaması değiştirmektedir.

A* algoritmasının kullanılması sırasında maliyet taşınacak düğümün öncelik derecesine göre değişmektedir. Düğümlerin öncelik katsayısı kullanıcılardan alınan önem derecesi raporu ve sistem yöneticisinin işaretlemesine göre belirlenmektedir. Aynı zamanda hizmet seviyesi sözleşmesinde yer alan her kullanıcı ve kurum bazında değişiklik gösterebilecek önem bildirimine göre gruplama yapılabilmektedir. Önemi yüksek olan sanal makinelerin taşınması sırasında öncelik katsayısı da maliyet fonksiyonuna eklenmektedir. Aynı zamanda sezgisel yaklaşım ile hesaplanan maliyet fonksiyonunda müşteri isterlerine göre değişiklik yapılabileceği gibi HSA kısıtlarına göre dinamik olarak değişkenlik gösterebilir.

Hibrit olarak kullanılan algoritmalarda; öncelikle projelerin öncelik dereceleri belirlenerek projelerin taşınma sırasının en optimum şekilde belirlenmesi sağlanmaktadır. Sonrasında göç sırası belirlenen projenin sahip olduğu sanal makine sayısı ve düğüm sayısı göz önünde bulundurularak Dijkstra, özelleştirilmiş A*, MIS ve MST algoritmaları ile Python NetworkX kütüphanesi kullanılarak oluşturulan simülasyon ortamında testler gerçekleştirilmiştir.

Şekil 5.1’ deki gibi bu tez kapsamında sezgisel olarak çalışan fonksiyon sanal yapıların sahip olduğu öncelik ve öneme göre optimize edilmiş hibrit bir ve yapı ile etkin göç yöntemi oluşturulmuştur.



Şekil 5.1 Etkin sanal makine göçü için yöntem geliştirilmesi

5.5. Deney Sonuçlarının Karşılaştırılması

Gerçekleştirilen çalışmalar ile sezgisel algoritma olan A* algoritması kullanılarak gerçekleştirilen ve öncelikli sanal makine kümesinin hesaplamalara katılarak yapılan göç çalışmalarında, en kısa yolu bulan Dijkstra Algoritması ile karşılaştırılmıştır. A* algoritmasında da sezgisel metot ‘None’ olarak kullanıldığında Dijkstra ile aynı şekilde çalışmaktadır. Ancak sezgisel fonksiyon olarak düğümlerin sahip olduğu öncelik katsayı olarak maliyet hesaplaması tekrar yapıldığında ortaya çıkan maliyette iyileştirmeler görülmektedir. Artan düğüm sayısı ile öncelik katsayısına sahip olan düğümlerde değişmektedir. Çizelge 5.2’de yer aldığı gibi genel sistemde öncelik katsayılarının içerisinde dengeli bir şeklilde dağıldığı varsayılmaktadır. Bu proje önceliği katsayıları sistemde hizmet seviyesi sözleşmesindeki yaptırımları kapsamakta olup projenin önceliğine göre hizmet verilen projede yaşanacak kesintinin maliyetinin daha fazla olduğunu göstermektedir.

Çizelge 5.2. Düğüm sayılarına göre öncelik dereceleri

Öncelik Katsayısı ⇒ / Düğüm Sayısı ⇓	1	2	3	4	5
100	1, 6, 11, 16, 21, 26, 31, 36, 41, 46, 51, 56, 61, 66, 71, 76, 81, 86, 91, 96	2, 7, 12, 17, 22, 27, 32, 37, 42, 47, 52, 57, 62, 67, 72, 77, 82, 87, 92, 97	3, 8, 13, 18, 23, 28, 33, 38, 43, 48, 53, 58, 63, 68, 73, 78, 83, 88, 93, 98	4, 9, 14, 19, 24, 29, 34, 39, 44, 49, 54, 59, 64, 69, 74, 79, 84, 89, 94, 99	5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100
75	1, 6, 11, 16, 21, 26, 31, 36, 41, 46, 51, 56, 61, 66, 71	2, 7, 12, 17, 22, 27, 32, 37, 42, 47, 52, 57, 62, 67, 72	3, 8, 13, 18, 23, 28, 33, 38, 43, 48, 53, 58, 63, 68, 73	4, 9, 14, 19, 24, 29, 34, 39, 44, 49, 54, 59, 64, 69, 74	5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75
50	1, 6, 11, 16, 21, 26, 31, 36, 41, 46	2, 7, 12, 17, 22, 27, 32, 37, 42, 47	3, 8, 13, 18, 23, 28, 33, 38, 43, 48	4, 9, 14, 19, 24, 29, 34, 39, 44, 49	5, 10, 15, 20, 25, 30, 35, 40, 45, 50
25	1, 6, 11, 16, 21	2, 7, 12, 17, 22	3, 8, 13, 18, 23	4, 9, 14, 19, 24	5, 10, 15, 20, 25
10	1, 6	2, 7	3, 8	4, 9	5, 10
5	1	2	3	4	5

Deney sırasında 5, 10, 25, 50, 75 ve 100 düğüme ait simülasyon ortamları hazırlanmıştır. Hazırlanan topolojilerde her bir düğümün ağırlığı 1 olarak kabul edilmiştir. Her düğüm sayısına ait sanal makinelerin göç mekanizması 10 kere çalıştırılarak sonuç kısmında ortalaması alınmıştır. Aynı zamanda canlı ortam testlerinden elde edilen sonuçlar ile artan makine sayısı ile elde edilen maliyetlerin orantılı olarak artacağı varsayılmaktadır.

Taşınacak sanal makinenin hanigi fiziksel makine üzerine taşınabileceği sistem yöneticisi tarafından belirlenebilmektedir ya da otomatik olarak seçilen bir fiziksel makineye taşıma işlemi yapılabilmektedir. Sanal makinenin taşınabileceği her bir fiziksel makine değerlendirilerek fiziksel makinenin arasındaki en kısa yol tercih edilerek değerlendirilmiş sanal makine göçünün optimum düzeyde tutulması hedeflenmiştir.

Öncelikli olarak Dijkstra algoritması kullanılarak deney gerçekleştirilmiş ardından ise sezgisel olarak çalışan A* algoritması kullanılmıştır. A* algoritması sezgisel fonksiyon düğümlerin öncelik katsayılarına göre optimize edilmiştir. Böylelikle önem derecesi yüksek olan ve hizmet seviyesi sözlşemesinde yaptırımları yüksek maliyetlere yol açan projeler öncelikli ve olabildiğince en büyük gruplarla taşınarak maliyet düşürülmeye çalışılmıştır. Artan düğüm sayısına önceliklere sahip olan düğümlerin sayısı ve miktarı değişmiş olup bunların sistem yöneticisi tarafından atandığı varsayılmaktadır.

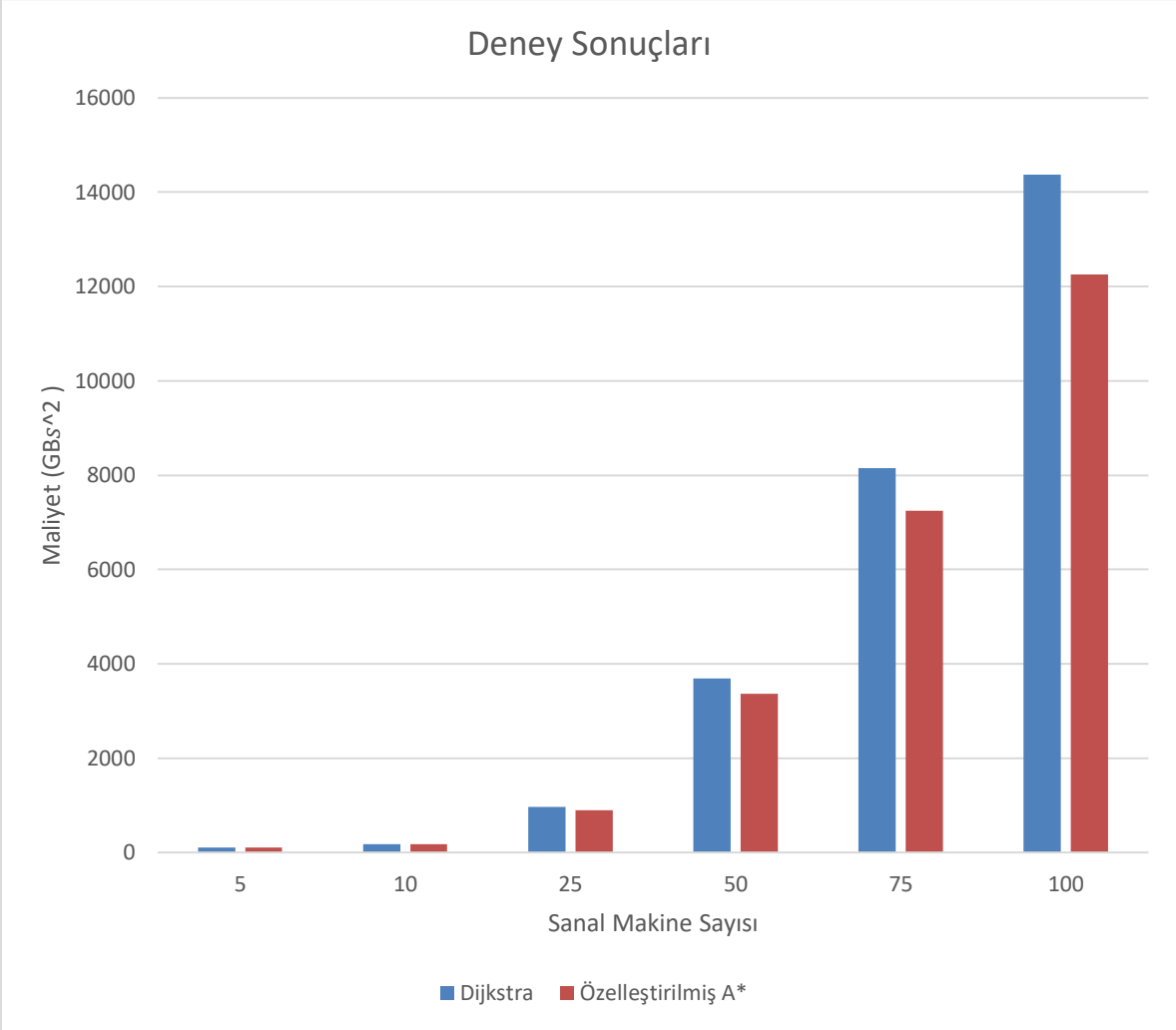
Deney sonuçları kullanılan algoritmalara göre Çizelge 5.3'daki gibidir. Değişen sanal makineleri içerisindeki sanal makine makine boyutlarının dengeli dağılım gösterdiği varsayılmaktadır. Testler sırasında her bir sanal makine sayısı ile ilgili test durumu 10 kere çalıştırılarak maliyetin ortalaması alınmaktadır. Aynı zamanda göç süreleri de 10 kere çalıştırma sonucunda eldilen toplam göç sürelerinin ortaması olarak hesaplanmaktadır. Testlerde Çizelge 5.3'te yer alan toplam göç sürelerinden hazırlık ve tamamlanma süreleri çıkarılarak ham göç süreleri hesaplanarak kesinti yaşanan sürede göz önünde bulundurularak hesaplama yapılmaktadır. Öncelikli olan projelerin öncelikli olarak en kısa yoldan fiziksel makinelere taşınarak HSA kısıtlarından doğan maliyetlerin engellenmesi sağlanmaktadır.

Çizelge 5.3. Kullanılan algoritmalara göre test sonuçları

Sanal Makine Sayısı	Toplam RAM Büyüklüğü (%60) (GB)	Dijkstra			Özelleştirilmiş A*			MST			MIS		
		Toplam Göç Süresi (s)	Göç sırasındaki kesinti süresi (s)	Maliyet (GBs^2)	Toplam Göç Süresi (s)	Göç sırasındaki kesinti süresi (s)	Maliyet (GBs^2)	Toplam Göç Süresi (s)	Göç sırasındaki kesinti süresi (s)	Maliyet (GBs^2)	Toplam Göç Süresi (s)	Göç sırasındaki kesinti süresi (s)	Maliyet (GBs^2)
5	36	58,4	1,3	107	57,8	1,1	103,77	43	4	128,4	44	6,1	149,42
10	72	113,8	2,3	178,94	38,9	1,89	169,67	83	8,9	253,95	85	9,1	255,2
25	180	281,3	4,4	968,32	276,4	3,4	902,21	208	11,3	2803	212	15,1	2285,8
50	360	550	5,8	3685,82	548	4,15	3264,18	493	45	22094	495	49	26663
75	540	823,4	30,6	8151,32	821	28,4	7242,26	833	65	74192	838	74	96449
100	720	1093	41,3	14368,82	1072	39,8	12258,38	1236	84	175534	1243	97	232798

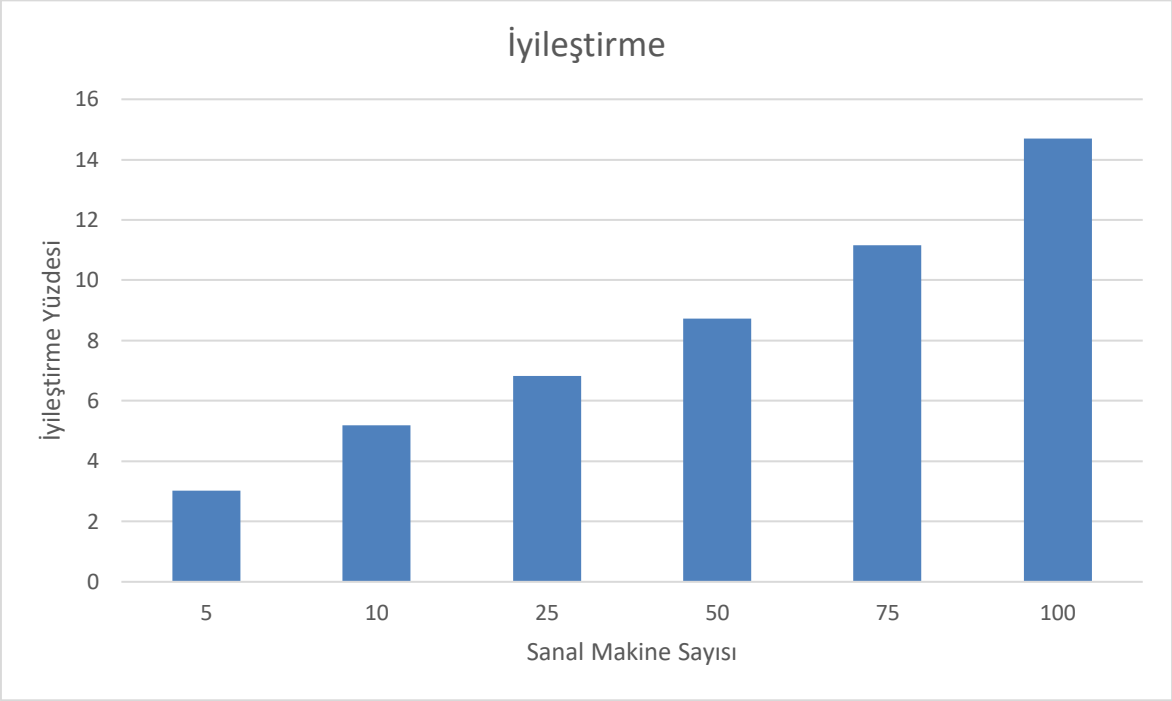
Deney sırasında sanal makineler taşınırken sanal makinenin büyüklüğü göz önünde bulundurularak sistemin büyüklüğünü ölçmek için geçen süre kullanılmıştır. Aynı zamanda göç işlemi sırasında taşınan veri miktarıda dikkate alınarak GBs^2 biriminde sonuçlar elde edilmiştir. Ayrıca maliyet fonksiyonuna eklenen öncelik çarpanı eklenmiş olup bir katsayı şeklindedir ve birimsiz olarak değerlendirilmiştir.

Elde edilen deney sonuçları grafiksel olarak Şekil 5.2'deki gibidir. Düğüm sayısı arttıkça mevcut sanal makine göç maliyetinin artışı görülmektedir. En kısa yolun hesaplanarak göç işlemi gerçekleştirilmesinin yanı sıra A* sezgisel algoritması ile artan düğüm sayısına göre artan iyileştirme oranları gözlemlenmektedir.



Şekil 5.2 Deney sonuçları

Deney sırasında A* algoritması en kısa yolu hesaplayan bir algoritmadır ve içerisinde yer alan sezgisel fonksiyonu 'None' olarak kabul edilirse Dijkstra algoritması ile aynı sonuç elde edilmektedir. Sanal makinelerin ve bulut ortamındaki proje hesaplarının öncelik derecelerine göre optimize edilen ve sezgisel fonksiyonu bu şekilde düzenlenen A* algoritması ile düğüm sayılarına bağlı olarak iyileştirilmelerin sağlandığı gözlemlenmektedir. İyileştirmelere ait grafik Şekil 5.3'deki gibidir.



Şekil 5.3 Elde edilen iyileştirme grafiği

Sanal makine veya sanal ağ sayısı az iken ortaya çıkan maliyet, Dijkstra ile özelleştirilmiş A* algoritması karşılaştırıldığında sanal makine sayısına göre artan bir grafik göstermektedir. Taşınan sistemde sanal makine sayısı arttıkça hizmet seviyesi anlaşmasının göz önünde bulundurulmasının maliyetin daha da düşürülmesine olanak sağladığı sonucuna ulaşılmıştır. HSA kısıtları ile fiziksel makine üzerindeki karmaşık şekilde bulunan farklı projelere ait sanal makinelerin taşınma sırası düzenlenmektedir. Proje bazlı taşıma da öncelikli olan makinelerin taşınması sırasında fiziksel makine üzerinde hangi sıra ile sanal makinenin taşınacağına kolaylıkla karar verilebilmekte, projeye bağlı olan ve birbiri ile bağımlılığı olan makineler öncelikli olarak taşınmaktadır.

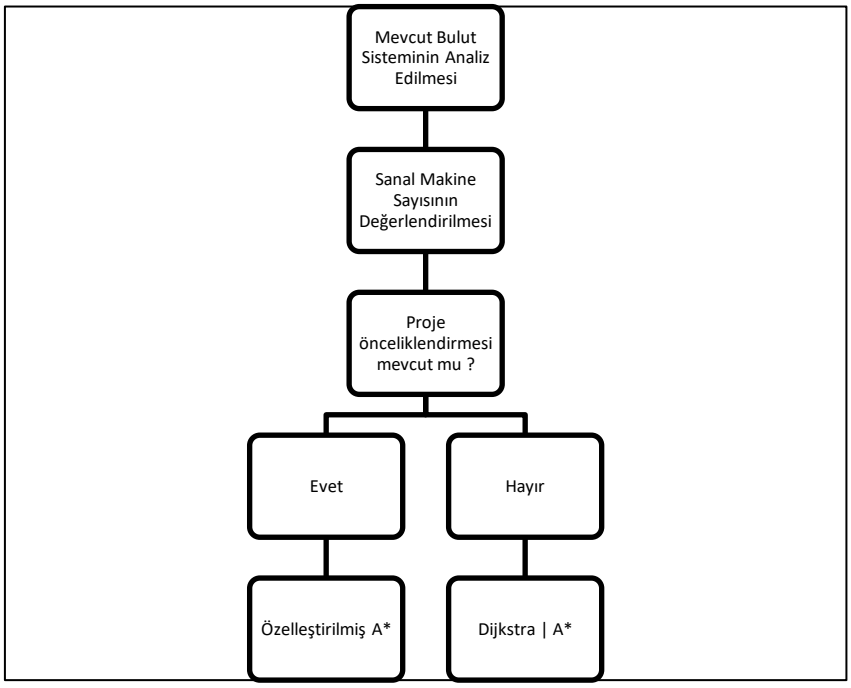
Dijkstra yöntemi ile taşındığında projelerin seçimi rasgele olurken, HSA kısıtlarında yüksek öneme sahip proje bekletilmekte olabileceği için kesinti süresinin maliyeti daha fazla olmaktadır.

Özelleştirilmiş A* algoritmasına ile yapılan göç işleminde Dijkstra algoritmasına göre maliyette %3-%16 aralığında iyileştirme sağladığı sonucuna ulaşılmıştır. Sanal makine sayısını artışına göre artan iyileştirme oranın yapılan tez çalışmasının ilerletilerek daha büyük veri merkezlerinde kullanılabilecek bir temel oluşturmaktadır.

6. SONUÇ VE ÖNERİLER

Günümüzde artan teknoloji kullanımı ile hayatımızda büyük yer edinen sanallaştırma sistemleri sektörde hizmet ve yönetenler açısından önemli bir yere sahiptir. Bulut bilişim hizmeti veren sistemlere sahip kurum kuruluşlarda sistemin kurulumu, yönetilmesi ve idamesi sistem sağlığı açısından önemlidir. Kullanılan fiziksel kaynakların bakımı, onarımı veya yazılımsal olarak sistemlerin güncellenmesi gibi konular bulut hizmetleri kapsamında hizmet veya servis verilen sanal yapıların taşınması yer değiştirmesi veyahut göç ettirilmesini konusunu doğurmaktadır.

Bu çalışmada 5, 10, 25, 50, 75 ve 100 düğüm sayılarına sahip sanal yapılarında göç maliyetinin en kısa yol hesaplanmasının yanı sıra sezgisel olarak çalışan A* algoritmasının sezgisel fonksiyonunun bulut sistem yöneticileri tarafından atanmış sanal yapıların öncelik katsayısı gözetilerek optimize edilmesi yani hizmet seviyesi anlaşmasının da göç maliyetinde göz önünde bulundurulması ile ortaya çıkan göç maliyeti değerlendirilmiştir. Elde edilen sonuçlarda sanal makine sayısına göre ortaya çıkan maliyetin değiştiği gözlenmekte olup en uygun yöntem araştırılmış olup Şekil 6.1'deki gibi sonuca ulaşılmıştır.



Şekil 6.1 Etkin sanal makine göçü için etkin yöntemler

Sistemde taşınacak sanal makineler değerlendirilmiş olup sistemde değişken HSA kısıtları varlığı kontrol edilir. Sistemde herhangi bir önceliklendirme yok ise Dijkstra veyahut sezgisel fonksiyonu tanımlanmamış en kısa yolu veren algoritmalardan birisi kullanılarak göç işlemi gerçekleştirilebilir.

Sistemde bir önceliklendirme söz konusu ise sezgisel fonksiyonu özelleştirilmiş A* algoritması kullanılarak sistemdeki sanal makine sayısına bağlı olarak Dijkstra algoritmasının kullanıldığı durumlara göre %3 ila %16 arasında sanal makine göç işleminde maliyeti iyileştirilmiştir.

Bulut sistemlerinde yer alan sanal yapıların göç ettirilmesi, sistem sahibine hem kaynak hem de personel maliyeti oluşturmaktadır. Ortaya çıkan maliyetin düşürülmesine ve sistemde yaşanması muhtemel performans sorunlarını en aza indirmek için bu yüksek lisans tez çalışması kapsamında 'Bulutta Etkin Sanal Makine Göçü için Yöntemler' çalışması gerçekleştirilmiştir.

Literatüre bulut sistemlerindeki göç maliyetlerinin hesaplanmasında sanal yapıların öncelik ve önem gibi kriterlerinde göz önünde bulundurulmasına dair katkıda bulunulmuştur. Aynı zamanda kullanılan maliyet fonksiyonu müşteri isterleri göz önünde bulunarak optimize edilebilir, müşterilerin önceliklerine yeniden etkin maliyet hesaplanabilir. Bu kapsamda yapılan çalışmaların gelecekte sanallaştırma yapılarında yeni bir yer edinen konteyner teknolojileri üzerinde gerçekleştirilmesi hedeflenmektedir.

KAYNAKLAR

1. Popek, G.J., Goldberg, R.P. (1974). Formal requirements for virtualizable third generation architectures, *Communications of Association for Computing Machinery*, 17(7), 412–421.

2. İnternet: Sanallaştırma (Virtualization) Nedir? Türleri Nelerdir? URL: https://radore.com/blog/sanallastirma-virtualization-nedir-turleri-nelerdir.html, Son Erişim Tarihi: 05.04.2019.

3. Hwang, J., Zeng, S., Wood, T. (2013). A component-based performance comparison of four hypervisors, *International Symposium on Integrated Network Management*, 14(4), 269–276.

4. Kivity A, Lublin U, Liguori A, Kamay Y, Laor D. (2007). *Kvm: The Linux virtual machine monitor*, Linux Symposium., Melbourne, 12(2), 225–230.

5. İnternet: QEMU URL: https://www.qemu.org/. Son Erişim Tarihi: 18.03.2019.

6. Waldspurger, C.A. (2002). Memory resource management in VMware ESX server. *Association for Computing Machinery Operation Systems*, IEEE, 181-185

7. Elsayed, A., Abdelbaki, N. (2013). *Performance evaluation and comparison of the top market virtualization hypervisors*, 8th International Conference on Computer Engineering and Systems, IEEE, 45–50.

8. İnternet: Microsoft Virtualization with Hyper-V. URL: https://dl.acm.org/citation.cfm?id=1593830. Son Erişim Tarihi: 18.03.2019.

9. İnternet: About Us - Xen Project. URL: https://xenproject.org/about-us/. Son Erişim Tarihi: 18.03.2019.

10. Demiral, Y., Demirci, M. (2018). *An investigation of hypervisor effect on virtual networks performance.* 2018 26th Signal Processing and Communications Applications Conference (SIU), İzmir, 1–4.

11. İnternet: Software-Defined Multi-Cloud Computing: A Vision, Architectural Elements, and Future Directions. URL: https://arxiv.org/pdf/1805.10780.pdf. Son Erişim Tarihi: 22.01.2019.

12. Akyildiz, I.F., Lee, A., Wang, P., Luo, M., Chou, W. (2014). A roadmap for traffic engineering in SDN-OpenFlow networks. *Computer Networks*, 71(1), 1–30.

13. İnternet: Yazılım Tanımlı Ağlar (Software-Defined Networks). URL: https://www.researchgate.net/publication/323999508_Yazilim_Tanimli_Aglar_Software-Defined_Networks. Son Erişim Tarihi: 18.03.2019.

14. McKeown, N., Anderson, T. Balakrishnan, H. (2008). OpenFlow, *Association for Computing Machinery Computer Communation*, 38(2), 69-78.

15. Tkachova, O., Salim, M.J., Yahya, A.R. (2015). *An analysis of SDN-OpenStack integration*, 2015 2nd International Scientific-Practical Conference Problems of Infocommunications Science and Technology, IEEE, 60–62.

16. İnternet: Mininet: An Instant Virtual Network on your Laptop Mininet. URL: http://mininet.org/. Son Erişim Tarihi: 17.03.2019.

17. İnternet: Mininet as Software Defined Networking Testing Platform. URL: https://www.researchgate.net/publication/287216738_Mininet_as_Software_Defined_Networking_Testing_Platform. Son Erişim Tarihi: 24.01.2019.

18. İnternet: sFlow-RT. URL: https://sflow-rt.com/. Son Erişim Tarihi: 17.03.2019.

19. İnternet: Home - OpenDaylight. URL: https://www.opendaylight.org/. Son Erişim Tarihi: 28.03.2019.

20. Singh, P., Manickam, S. (2015). *Design and deployment of OpenStack-SDN based test-bed for EDoS.*, 4th International Conference on Reliability, Roma, 1-4.

21. Nikbazm, R., Dashtbani, M. Ahmadi, M. (2015). *Enabling SDN on a special deployment of OpenStack*, 5th International Conference on Computer and Knowledge Engineering, Mashhad, 337–342.

22. Mao, M., Li, J., Humphrey, M. (2010). *Cloud auto-scaling with deadline and budget constraints*, 11th Association for Computing Machinery International Conference on Grid Computing, Dubai, 41–48.

23. İnternet: The NIST, Definition of Cloud Computing. URL: https://csrc.nist.gov/publications/detail/sp/800-145/final. Son Erişim Tarihi: 24.03.2019

24. İnternet: Veri Merkezi Nedir? URL: https://www.konzek.com/data-center-veri-merkezi-nedir-standartlari-nelerdir--s1187.html. Son Erişim Tarihi: 25.03.2019.

25. Sesver, D., Tuna, S., Aktaş, M.S. (2018). *Dağıtık Büyük Veri İşleme Platformları Üzerinde Birliktelik Kuralları Çıkarma Algoritmalarının Gerçeklenmesi,* International Conference on Computer Science and Engineering, Bosna,1-4

26. İnternet: A View of Cloud Computing. URL: https://dl.acm.org/citation.cfm?id=1721672. Son Erişim Tarihi: 24.03.2019

27. Serhani, M.A., Atif, Y. Benharref, A. (2014). *Towards an adaptive QoS-driven monitoring of cloud SaaS*, 3th International Conference on Reliability, Barcelona, 1-4.

28. Li, Z., Zhang, Y., Liu, Y. (2017). Towards a full-stack devops environment for cloud-hosted applications, *Tsinghua Science Technology Journal*, 22(1), 1-9.

29. Qevani, E., Panagopoulou, M., Stampoltas, C., Tsitsipas, A., Kyriazis, D. (2014). What can *OpenStack adopt from a Ganeti-based open-source IaaS?,* IEEE International Conference on Cloud Computing, CLOUD, San Francisco, 833–840.

30. İnternet: Openstack. URL: https://www.openstack.org/assets/survey/Public-User-Survey-Report. Son Erişim Tarihi: 24.04.2018

31. Clark, C., Fraser, K., Hand, S. (2017). *Live Migration of Virtual Machines,* Association for Computing Machinery Cloud Computing Conference, Santa Clara, 1-4.

32. Callegati, F., Cerroni, W. Contoli, C., Santandrea, G. (2014). *Performance of Network Virtualization in cloud computing infrastructures: The OpenStack* case, 2014 IEEE 3rd International Conference on Cloud Networking, Luxembourg, 87-91.

33. Hu, B., Yu, H. (2013). *Research of scheduling strategy on openstack*, Conference on Cloud Computing and Big Data, Fuzhou, 1-4.

34. Mansour, I., Cooper, K., Bouchachia, H. (2014). *Effective live cloud migration,* 2016 IEEE 4th Internet Things Cloud, FiCloud, Vienna, 334–339.

35. Eramo, V., Ammar, M., Lavacca, F.G. (2018). *Migration energy aware reconfigurations of virtual network function instances in NFV architectures* Association for Computing Machinery Cloud Computing Conference, Carlsbad, 1-4.

36. Alicherry, M., Lakshman, T. (2012). *Network aware resource allocation in distributed clouds*, IEEE International Conference on Computer Communications, Orlando 89-91.

37. İnternet: Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers. URL: http://www.cloudbus.org/papers/Optimal-VM-Consolidation-CCPE2012.pdf Son Erişim Tarihi: 17.07.2018

38. İnternet: Dive Into VM Live Migration OpenStack Liberty Summit URL: https://01.org/sites/default/files/dive_into_vm_live_migration_2.pdf. Son Erişim Tarihi: 22.03.2019.

39. Rawat, S., Tyagi, R., Kumar, P. (2016). *An investigative study on challenges of live migration*, 2016 5th International Conference on Reliability, Porto, 129–134.

40. Keller, E., Ghorbani, S., Caesar, M., Rexford, J. (2012). *Live migration of an entire network (and its hosts)*, 11th ACM Work Hot Top Networks, Redmond, 109–114.

41. Lo, S., Ammar, M., Zegura, E. (2013). *Design and analysis of schedules for virtual network migration.* IFIP Network Conference, New York, 1–9.

42. İnternet: Addressing the Benefits of Infrastructure in the Cloud, URL: http://archive.gtra.org/component/attachments/download/453 Son Erişim Tarihi: 24.04.2019

43. Zhang, Q., Cheng, L., Boutaba, R. (2010). Cloud Computing: state-of-the-art and research challenges. *Journal of Internet Services and Applications,* 1(1), 7-18.

44. İnternet: The Cloud Adoption Toolkit: Supporting cloud adoption decisions in the enterprise. URL: https://arxiv.org/pdf/1008.1900.pdf. Son Erişim Tarihi: 23.03.2018

45. İnternet: Amazon EC2. URL: https://aws.amazon.com/ec2/?sc_channel=PS&sc_campaign=acquisition_TR&sc_publisher=google&sc_medium=ACQ-P%7CPS-GO%7CBrand%7CDesktop%7CSU%7CCompute%7CEC2%7CTR%7CEN%7CText&sc_content=ec2_e&sc_detail=amazon ec2&sc_category=Compute&sc_segment=293607107934&sc. Son Erişim Tarihi: 09.04.2019.

46. İnternet: Microsoft Azure Bulut Bilgi İşlem Platformu ve Hizmetleri. URL: https://azure.microsoft.com/tr-tr/. Son Erişim Tarihi: 09.04.2019.

47. İnternet: CRM Software Cloud Computing Solutions - Salesforce EMEA. URL: https://www.salesforce.com/eu/?ir=1. Son Erişim Tarihi: 09.04.2019.

48. Leite, A.F., Boukerche, A., Magalhaes, A.C., Eisenbeis, C., Tadonki, C., Ralha, C.G. (2016). Power-aware server consolidation for federated clouds, *Concurrency and Computation*, 28(12), 3427-3444.

49. Voorsluys, W., Broberg, J., Venugopal, S., Buyya, R. (2009). Cost of virtual machine live migration in clouds: A performance evaluation, *Computer Science and Artifical Intelligence*, 5391(1), 254–265.

50. Barham, P., Dragovic, B., Fraser, K. (2003). *Xen and the art of virtualization,* Symposium on Operating systems principles, 64-68.

51. Gao, Y., Guan, H., Qi, Z., Hou, Y., Liu, L. (2013). A multi-objective ant colony system algorithm for virtual machine placement in cloud computing, *Journal of Computer and System Sciences*,27(2), 256-264.

52. Forsman, M., Glad, A., Lundberg, L., Ilie, D. (2015). Algorithms for automated live migration of virtual machines, *Journal of System Software*, 52(2), 44-56.

53. Beloglazov, A., Abawajy, J., Buyya, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing, *Future Generic Computer Systems*, 28(5), 755-768.

54. Nasim, R., Taheri, J., Kassler, A.J. (2016). *Optimizing Virtual Machine Consolidation in Virtualized Datacenters Using Resource Sensitivity*, Cloud Com Conference, San Francisco 1-4.

55. Ramanan, P., Brown, D.J., Lee, C.C., Lee, D.T. (1989). On-line bin packing in linear time. *Journal of Algorithms*, 10(3), 305-326.

56. Luo, J.P., Li, X., Chen, M.R. (2014). Hybrid shuffled frog leaping algorithm for energy-efficient dynamic consolidation of virtual machines in cloud data centers, *Expert System Applications*, 41(13), 5804-5816.

57. Bobroff, N., Kochut, A., Beaty, K. (2007). *Dynamic placement of virtual machines for managing SLA violations*, 10th IFIP/IEEE International Symposium on Integrated Network Management, Lisbon, 347-353.

58. Kavvadia, E., Sagiadinos, S., Oikonomou, K., Tsioutsiouliklis, G., Aïssa, S. (2015). Elastic virtual machine placement in cloud computing network environments, *Computer Networks*, 93(3), 435-447.

59. Yao, L., Wu, G., Ren, J., Zhu, Y., Li, Y. (2014). Guaranteeing fault-tolerant requirement load balancing scheme based on VM migration, *Computer Systems of Journal*, 57(2), 225-232.

60. Wei, X., Hu, S., Li, H., Yang, F., Jin, Y. (2014). A Survey on Virtual Network Embedding in Cloud Computing Centers, *The Open Automation and Control Systems Journal*, 6(1), 414-425

61. Gerofi, B., Vass, Z,. Ishikawa, Y. (2011). *Utilizing memory content similarity for improving the performance of replicated virtual machines*, 2011 4th IEEE International Conference on Utility and Cloud Computing, Melbourne, 78-80.

62. Diallo, M.H., August, M., Hallman, R., Kline, M., Slayback, S.M., Graves, C. (2017). *AutoMigrate: a framework for developing intelligent, self-managing cloud services with maximum availability*, 2016 International Conference on Cloud and Autonomic Computing, Augsburg, 1-6.

63. Yang, C.T., Huang, K.L., Liu, J.C., Su, Y.W., Chu, W.C. (2013). *Implementation of a power saving method for virtual machine management in cloud*, International Conference on Cloud Computing and Big Data, Fuzhou, 1-6.

64. Al-kiswany, S., Subhraveti, D., Sarkar, P., Ripeanu, M. (2011). *VMFlock: Virtual Machine Co-Migration for the Cloud*. High Performance Distributed Computing, Edinburgh, 232-248.

65. Wood, T., Ramakrishnan, K.K., Shenoy, P. (2014). CloudNet: Dynamic pooling of cloud resources by live WAN migration of virtual machines, *IEEE/ACM Transactions on Networking*, 23(5), 1568-1583.

66. Hines, M.R., Gopalan. K. (2009). *Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning*, 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments, Binghamton, 51-60.

67. Hu, L., Zhao, J., Xu, G., Ding, Y., Chu, J. (2013). *HMDC: Live virtual machine migration based on hybrid memory copy and delta compression*, Application of Math Information Science, 7(21), 639-646.

68. Deshpande, U., Kulkarni, U., Gopalan, K. (2012). *Inter-rack live migration of multiple virtual machines*, Cloud Computing Systems Conference, Hawaii, 52-56.

69. Deshpande, U., Wang, X., Gopalan, K. (2011). *Live gang migration of virtual machines*, Cloud Computing Systems Conference, Rome, 1-6.

70. Ritea, P., Morin, C., Priol, T. (2011). Shrinker: Improving live migration of virtual clusters, *Journal of Artificial Intelligence and Bioinformatics*, 7(1), 431-442

71. Lo, S., Ammar, M., Zegura, E. (2013). *Design and Analysis of Schedules for Virtual Network Migration*, 2013 IFIP Networking Conference, Brooklyn, 1-4.

72. Lo, S., Ammar, M., Zegura, E., Fayed, M. (2014). *Virtual network migration on real infrastructure: A PlanetLab case study,* 2014 IFIP Networking Conference, Trondheim, 1–9.

73. Zhao, Y., Lo, S., Zegura, E., Ammar, M., Riga, N. (2017). *Virtual Network Migration on the GENI Wide-Area SDN-Enabled Infrastructure,* 2017 IEEE Conference on Computer Communications, Atlanta, 265-270.

74. Eramo, V., Miucci, E., Ammar, M., Lavacca, F.G. (2017). An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures, *IEEE/ACM Transport Networks*, 25(4), 2008-2025.

75. Wang, H., Li, Y., Zhang, Y., Jin, D. (2015). *Virtual machine migration planning in software-defined networks*, IEEE Conference on Computer Communications Hong Kong, 487–495.

76. Kreutz, D., Ramos, F.M. V., Esteves, P., Esteve, C. (2015). Software-Defined Networking: A Comprehensive Survey. *IEEE Journal of Computer Communications*, 103(1), 14–76.

77. Nasim, R., Kassler, A.J., (2015). *Network-centric Performance Improvement for Live VM Migration*, 2015 IEEE 8th International Conference Cloud Computing, New York City, 106–113.

78. Voorsluys, W., Broberg, J., Venugopal, S., Buyya, R. (2009). *Cost of Virtual Machine Live Migration in Clouds: A Performance Evaluation*, IEEE International Conference on Cloud Computing, Bangalore, 254–265.

79. Xiaoyong, Y., Ying, L., Tong, J., Tiancheng, L., Zhonghai, W. (2015). *An Analysis on Availability Commitment and Penalty in Cloud SLA*, IEEE 39th Annual Computer Software and Applications Conference, Taiwan, 914–919.

80. Goudarzi, H., Ghasemazar, M., Pedram, M. (2012). *SLA-based Optimization of Power and Migration Cost in Cloud Computing*, 12th IEEE/ACM International Symposium on Cluster, Ottawa, 172–179.

81. Borhani, A.H., Hung, T., Lee, B., Qin, Z., Bagheri, Z. (2017) *Network-Aware VM Migration Heuristics for Improving the SLA Violation of Multi-Tier Web Applications in the Cloud*, 2017 25th Euromicro International Conference on Parallel, St. Petersburg, 454–462.

82. Wang, Y., Keller, E., Biskeborn, B. (2008). *Virtual Routers on the Move: Live Router Migration as a Network-Management Primitive*, ACM SIG on Data Communications Conference, Seattle, 288-292.

83. İnternet: Design and Evaluation of Virtual Network Migration Mechanisms on Shared Subsratate. URL: https://smartech.gatech.edu/bitstream/handle/1853/54379/LO-DISSERTATION-2015.pdf. Son Erişim Tarihi: 18.12.2018

84. M., Riga, N. (2018). *Virtual Network Migration,* 2018 IEEE Conference on Computer Communications, Honolulu, 255-260.

85. İnternet: Beloglazov, Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing, URL: http://beloglazov.info/thesis.pdf., Son Erişim Tarihi: 03.07.2017

86. Demirci, M. (2015). *A Survey of Machine Learning Applications for Energy-Efficient Resource Management in Cloud Computing Environments*, 2015 IEEE 14th International Conference Mach Learn Application, Miami, 1185–1190.

87. Ferreto, T.C., Netto, S., Calheiros, R.N. (2011). Server consolidation with migration control for virtualized data centers, *Future Generation Computer Systems*, 27(3), 1027-1034.

88. Fan, X., Weber, D., Barroso, L.A. (2007). *Power provisioning for a warehouse-sized computer,* 34th Annual International Symposium on Computer Architecture, San Diego, 13-23.

89. Rybina, K., Dargie, W., Strunk, A. (2013). *Investigation into the energy cost of live migration of virtual machines*, 2013 Sustainable Internet and ICT for Sustainability, Palermo 1–8.

90. Mazumdar, S., Pranzo, M. (2017). Power efficient server consolidation for Cloud data center, *Future Generation Computer Systems*, 70(1), 4-16.

91. Sunil, K., Santhi, P. (2015). Heuristics based server consolidation with residual resource defragmentation in cloud data centers, *Future Generation Computer Systems,* 50(1), 87-98.

92. Wang, Y., Wang, X. (2014). Performance-controlled server consolidation for virtualized data centers with multi-tier applications, *Sustain Computer Informatics Systems.* 4(1), 52-65.

93. Sudevalayam, S., Kulkarni, P. (2011). *Affinity-aware modeling of CPU usage for provisioning virtualized applications,* 2011 IEEE 4th International Conference on Cloud Computing, Washington, 1-4.

94. Hosseini, M., Rahmani A.M., Sahafi, A. (2018). A survey study on virtual machine migration and server consolidation techniques in DVFS-enabled cloud datacenter: Taxonomy and challenges, *Journal of King Saud University - Computer and Information Sciences,*32(1), 20-32.

95. Al-Azzoni, I. (2015). Server consolidation for heterogeneous computer clusters using Colored Petri Nets and CPN Tools. *Journal of King Saud University - Computer and Information Sciences,* 21(1), 46-62.

96. Wang, F., Ling, R., Zhu, J., Li, D. (2015). *Bandwidth guaranteed virtual network function placement and scaling in datacenter networks,* 34th International Performance Computing and Communications Conference, Nanjing, 1-6.

97. Sedaghat, M., Elmroth, E. (2016). Decentralized cloud datacenter reconsolidation through emergent and topology-aware behavior, *Future Generation Computer Systems,* 56(1), 51-63.

98. İnternet: Virt-LM, *URL:* https://research.spec.org/icpe_proceedings/2011/p307.pdf Son Erişim Tarihi: 18.09.2018.

99. Rajagopalan, S. (1999). Adoption timing of new equipment with another innovation anticipated, *IEEE Transport Engineering Management*, 46(1), 14-25.

100. İnternet: Cloud Computing Services URL: https://cloud.google.com/. Son Erişim Tarihi: 16.03 2019

101. İnternet: Bulut Bilişim ve Büyük Veri. URL: https://www.b3lab.org/. Son Erişim Tarihi: 16.04. 2019.

102. İnternet: NetworkX documentation. URL: https://networkx.github.io/documentation/latest/_modules/networkx/algorithms/shortest_paths/astar.html. Son Erişim Tarihi: 29.04.2019

103. İnternet: Bağımsız düğümler (Anti Clique, Independent Set) – Bilgisayar Kavramları. URL: http://bilgisayarkavramlari.sadievrenseker.com/2008/08/02/bagimsiz-dugumler-anticlique-independent-set/. Son Erişim Tarihi: 30.06.2017.

104. İnternet: Maximum Independent Set, URL: http://www.gise.cse.iitb.ac.in/MAXIMUM_INDEPENDENT_SET.pdf. Son Erişim Tarihi: 30.07.2017.

105. İnternet: min spanning tree URL: http://bilgisayarkavramlari.sadievrenseker.com/tag/minimum-spanning-tree/. Son Erişim Tarihi: 30.06.2017.

106. Graham, R.L., Hell, P. (1985). On the History of the Minimum Spanning Tree Problem. *IEEE Annals of the History of Computing*, 7(1), 43–57.

107. Turk, S., Radeke, R., Lehnert, R. (2010). *Network migration using ant colony optimization*, 9th Conference of Telecommunication, Ghent, 1–6.

108. Gower, J.C., Ross, G.J.S. (1969). Minimum Spanning Trees and Single Linkage Cluster Analysis, *Journal of the Royal Statistical Society*, 18(1), 54–64.

109. İnternet: Kruskal Asgari Tarama Ağacı Algoritması. URL: http://bilgisayarkavramlarisadievrenseker.com/2007/12/24/kruskal-asgari-tarama-agaci-algoritmasi/. Son Erişim Tarihi: 01.07.2017.

110. İnternet: CPU nedir? URL: http://www.bilgisayarsistemleri.net/cpu/cpu-nedir-cpu-ne-ise-yarar-t65.html. Son Erişim Tarihi: 05.04.2019.

111. İnternet: RAM Nedir? URL: http://bilgisayarhocam.blogspot.com/2012/03/ram-nedir.html Son Erişim Tarihi: 05.04.2019.

112. İnternet: Internet Delay Space Study at Rice University. URL: http://www.cs.rice.edu/~eugeneng/papers/IMC06.pdf. Son Erişim Tarihi: 05.04.2019.

113. İnternet: IPERF: A Framework for Automatic Construction of Performance Prediction Models. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.40.2777.pdf. Son Erişim Tarihi: 02.04.2018.

114. İnternet: NetworkX. URL: https://networkx.github.io/. Son Erişim Tarihi: 25.03.2019.

115. Afaq, M., Rehman, S.U., Song, W.C. (2015). A Framework for Classification and Visualization of Elephant Flows in SDN-Based Networks, *Procedia Computer Science*, 65(1), 672–681.

116. Ahuja, R.K., Mehlhorn, K., Orlin, J., Tarjan, R.E. (1990). Faster algorithms for the shortest path problem. *Journal of Association for Computing Machinery*, 37(2), 213–223

ÖZGEÇMİŞ

Kişisel Bilgiler



Soyadı, adı	: DEMİRAL, Yasemin
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 04.11.1991, Afyonkarahisar
Medeni hali	: Bekar
Telefon	: 0 (533) 660 04 29
e-mail	: yasemin.demiral@gazi.edu.tr

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Yüksek lisans	Gazi Üniversitesi / Bilgisayar Mühendisliği	Devam Ediyor
Lisans	Anadolu Üniversitesi/Bilgisayar Mühendisliği	2014
Lise	Afyon Anadolu Öğretmen Lisesi	2009

İş Deneyimi

Yıl	Yer	Görev
2016-Halen	TÜBİTAK BİLGEM	Araştırmacı

Yabancı Dil

İngilizce

Yayınlar

1. Demiral, Y., Demirci, M. (2017). *An experimental study on efficient virtual network migration,* 2017 International Conference on Computer Science and Engineering, *Antalya* 522–527.

2. Demiral, Y., Demirci, M. (2018). *An investigation of hypervisor effect on virtual networks performance,* 2018, 26th Signal Processing and Communications Applications Conference, *İzmir,* 1-4.

Hobiler

Fotoğraf çekmek.



GAZİ GELECEKTİR...