T.C.

TOKAT GAZİOSMANPAŞA ÜNİVERSİTESİ

LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS PROGRAMI

BULUT BİLİŞİM TEKNOLOJİSİ İLE GÖREV ZAMANLAMA ALGORİTMALARININ PERFORMANSLARININ BELİRLENMESİ

YÜKSEK LİSANS TEZİ

Aslan Samet BİLGİÇ

Danışman: Dr. Öğr. Üyesi Kenan ZENGİN

TOKAT- 2025

ETİK SÖZLEŞME

Tokat Gaziosmanpaşa Üniversitesi Lisansüstü Eğitim Enstitüsü tez yazım kılavuzuna göre, Dr. Öğr. Üyesi Kenan ZENGİN danışmanlığında hazırlamış olduğum “Bulut Bilişim Teknolojisi İle Görev Zamanlama Algoritmalarının Performanslarının Belirlenmesi ” adlı Yüksek Lisans tezinin bilimsel etik değerlere ve kurallara uygun, özgün bir çalışma olduğunu, aksinin tespit edilmesi halinde her türlü yasal yaptırımı kabul edeceğimi beyan ederim.

03/02/2025

Aslan Samet BİLGİÇ

İthaf

Bu yüksek lisans tezini, Babam Şehit Uzman Çavuş Ali BİLGİÇ e ithaf ediyorum.

ÖNSÖZ

Tez konusunun belirlenmesinden çalışmanın tamamlanmasına kadar yürütülen bütün süreçte bana yol gösteren ve hiçbir yardımı esirgemeyen tez danışmanım Dr. Öğr. Üyesi Kenan ZENGİN hocama;

Desteğini her zaman yanımda hissettiğim sevgili eşim ve anneme motivasyon kaynağım olan çocuklarım Ali Efe ve Yağmur'a;

Teşekkürlerimi sunarım.

ÖZET

BULUT BILİŞİM TEKNOLOJİSİ ILE GÖREV ZAMANLAMA ALGORİTMALARININ PERFORMANSLARININ BELİRLENMESİ

Bilgiç, Aslan Samet

Yüksek Lisans, Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Kenan ZENGİN

03.02.2025, V + 52 sayfa

Bulut bilişim son yıllarda güncel ve popüler bir teknoloji haline gelerek her alanda karşımıza çıkar oldu. Aslında her alanda karşımıza çıkması da bu teknolojinin neden popüler olduğunu göstermektedir. Günümüzde birçok cihazın internet bağlantısı olmasına karşın yeterli düzeyde kaynağa sahip olmamaktadır. Burada ki kaynaktan kastımız gerek işleme yeteneği gerek saklama alanı gerek ise de enerji kaynağı yeterli olmuyor. İşte burada bu problemlerin çözümüne bulut bilişim devreye giriyor. Düşük kaynağa sahip cihazlar da, büyük verilere ve ihtiyaç duyduğu yüksek karmaşıklıktaki hesaplamalara ulaşılabiliyor. Bulut hesaplamayı (Cloud Computing) adaptif hesaplama kaynaklarına, depolama alanlarına, farklı uygulamalara ve sunuculara, servis sağlayıcı ile etkileşime ihtiyaç duymadan ve minimum yönetim maliyeti ile sağlayan internet tabanlı bir hesaplama sistemi olarak tanımlayabiliriz.

Bulut Bilişim ile ilgili hizmet sunan sistem ve ticari hizmet servisleri ile OpenStack gibi açık kaynaklı sistemler araştırılmış olup araştırma çalışmaları için CloudSim kullanılmıştır. CloudSim açık kaynaklı bulut hesaplamasının altyapısı ve servislerini içeren bir simülatördür. CLOUDS Lab tarafından Java dilinde geliştirilmiştir. Java Object Oriented (nesne yönelimli) bir dil olması sebebi ile araştırmacılara bu anlamda da kullanım kolaylığı sağlamaktadır.

Bulut bilişim gibi heterojen sistemler, dağıtılmış ve ölçeklenebilir bir ortam olduğu için performanslarını değerlendirmek sorunu daha zor hale gelmektedir. Bulut Bilişiminin Performans incelemesini yaparken sistem kaynaklarının doğru kullanarak kıyaslanabilir hale getirmek ve sistemin performansı anlayabilmek için anahtar olarak kabul edilen etkili bir zamanlama (task scheduling) algoritmasına ihtiyaç duyulmaktadır. Görev Zamanlaması (task scheduling) algoritmaları istemcilerin görevlerini mevcut ve uygun sanallaştırılmış kaynaklarla eşleştirmek için kullanılan bir tekniktir. Görev zamanlama algoritması olarak; First Come First Serve (FCFS), Shortest Job First (SJF) ve Round Robin (RR) gibi geleneksel algoritmalar ve Parçacık Sürüsü Optimizasyonu (PSO) ve Gri Kurt Algoritması (GWO) gibi sezgi üstü (meta-sezgisel) algoritmalar kullanılmıştır.

Bu Algoritmaları CloudSim üzerinde 30 adet girdi ile çalıştırılarak süreç performanslarını incelenmiştir. Yapılan çalışma sonucunda sezgi üstü (meta sezgisel) algoritmaların klasik algoritmalara göre üstünlüğünü görülmüştür. Sezgi üstü algoritmaların ise gelecek çalışmalarda daha iyi performans sergileyebilmesi için hibrit algoritmalar oluşturulması ve bu yönde çalışmaların yapılarak hibritleşmiş güçlü algoritmalar üretilmesi düşünülmektedir.

Anahtar Kelimeler: Gri Kurt Optimizasyonu (GWO), Parçacık Sürü Optimizasyonu (PSO), Cloudsim, Bulut Bilişim , Görev Zamanlama Algoritmaları, Kuantum Sonrası Kriptoloji ve Kuantum Sonrası İmzalama Şemaları

ABSTRACT

DETERMINATION OF PERFORMANCE OF TASK SCHEDULING ALGORITHMS WITH CLOUD COMPUTING TECHNOLOGY

Bilgiç, Aslan Samet

Master's Thesis, Department Of Computer Engineering

Advisor: Assist. Prof. Dr. Kenan ZENGİN

03.02.2025, V + 52 page

Cloud computing has become a current and popular technology in recent years and has come across us in every field. In fact, the fact that it comes across us in every field shows why this technology is popular. Today, many devices do not have sufficient resources despite having an internet connection. What we mean by resources here is that the processing ability, storage space and energy source are not sufficient. This is where cloud computing comes into play to solve these problems. Devices with low resources can also access large data and the high complexity calculations it requires. We can define cloud computing as an internet-based computing system that provides adaptive computing resources, storage areas, different applications and servers without the need for interaction with the service provider and with minimum management cost.

Systems and commercial service services that provide services related to Cloud Computing and open source systems such as OpenStack have been researched and CloudSim has been used for research studies. CloudSim is a simulator that includes the infrastructure and services of open source cloud computing. It was developed in Java by CLOUDS Lab. Java is an object-oriented language, which provides researchers with ease of use in this sense.

Since heterogeneous systems such as cloud computing are distributed and scalable environments, the problem of evaluating their performance becomes more difficult. When

performing the performance analysis of Cloud Computing, an effective scheduling algorithm is needed, which is considered as the key to making system resources comparable by using them correctly and to understand the performance of the system. Task scheduling algorithms are a technique used to match the tasks of clients with available and suitable virtualized resources. As task scheduling algorithms; Traditional algorithms such as First Come First Serve (FCFS), Shortest Job First (SJF) and Round Robin (RR) and meta-heuristic algorithms such as Particle Swarm Optimization (PSO) and Gray Wolf Algorithm (GWO) were used.

These algorithms were run on CloudSim with 30 inputs and their process performance was examined. As a result of the study, it was seen that meta-heuristic algorithms are superior to classical algorithms. In order for meta-heuristic algorithms to exhibit better performance in future studies, it is planned to create hybrid algorithms and to produce hybridized strong algorithms by conducting studies in this direction.

Key Words: Grey Wolf Optimization (GWO), Particle Swarm Optimization (PSO), Cloudsim, Cloud Computing, Task Scheduling, Post-Quantum Cryptography and Post-Quantum Signature Scheme

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

SİMGELER VE KISALTMALAR

Kısaltmalar	Açıklama
FCFS	First Come First Serve
SJF	Shortest Job First
RR	Round Robin
PSO	Particle Swarm Optimization
GWO	Grey Wolf Optimization
CRM	Customer Relationship Management
ERP	Enterprise Resource Planning
HRMS	Human Resource Management Systems
IaaS	Infrastructure As A Service
PaaS	Platform As A Service
SaaS	Software As A Service
AWT	Average Waiting Time
UOV	Unbalanced Oil and Vinegar
LUOV	Lifted Unbalanced Oil and Vinegar
HFE	Hidden Field Equations
Gemms	Great Multivariate Short Signature
IOT	Internet of Things

1. GİRİŞ

Bulut bilişim son yıllarda güncel ve popüler bir teknoloji haline gelerek her alanda karşımıza çıkar oldu. Aslında her alanda karşımıza çıkması da bu teknolojinin neden popüler olduğunu göstermektedir. Şöyle ki, günümüzde birçok cihazın internet bağlantısı olmasına karşın yeterli düzeyde kaynağa sahip olmamaktadır. Burada ki kaynaktan kastımız gerek işleme yeteneği gerek gerek saklama alanı gerek ise de enerji kaynağı yeterli olmuyor. İşte burada bu problemlerin çözümüne bulut bilişim devreye giriyor. Düşük kaynağa sahip cihazlarda, büyük verilere ve ihtiyaç duyduğu yüksek karmaşıklıktaki hesaplamalara ulaşılabiliyor. Bulut hesaplamayı (Cloud Computing) tanımlayacak olursak, adaptif hesaplama kaynaklarına, depolama alanlarına, serverlara, farklı uygulamalara ve servislere servis sağlayıcı ile etkileşime ihtiyaç duymadan ve minimum yönetim maliyeti ile sağlayan bir internet tabanlı hesaplama sistemidir (Aiello ve ark 2009). Bu yüzden Bulut Hesaplama gelecek vaat eden bir teknolojidir. Şöyle ki teknolojinin ilerlemesi ve ağ alt yapılarının gelişmesi ile birlikte Bulut Bilişimin önünü daha da açmaktadır. Gelişen işlemci yetenekleri ve paralel programlama alanında yapılan çalışmalarda Bulut Bilişimin yeteneklerinin artmasını sağlamaktadır. Bulut Bilişim popüler bir sistem olmasına karşın şu an için hala yeni bir teknoloji olduğundan dolayı üzerinde yapılmış çok fazla çalışma olmamakla beraber yetenekleri hakkında kesin hükümler verilememektedir. Hatta gelişimini sağlayamadan ve olgunlaşamadan COVID-19 salgınından ötürü yaygın bir teknoloji haline dönüştü. Salgın ile birlikte, çalışma hayatında birçok sektörde uzaktan çalışma sistemine geçilmesi, eğitim alanında okullarda yüz yüze eğitiminin yerini uzaktan eğitimin alması, evlerde geçirilen sürenin artmasından dolayı dijital platformlarına ilginin artması ve yerelden ulusal firmalara kadar e ticaret kapsamında servis imkanlarında kapsayacak şekilde uygulamaların geçilmesi bulut bilişimin yaygın olarak kullanılmasına sebep oldu ve bu sayede aslında test süresini geçirmiş oldu. Bu süreç bu yüzden Bulut Bilişimin gelişimine ve geleceğine olumlu yönde etki etmiştir.

1.1 Araştırmanın Konusu

Bulut Teknolojisinin ilk kullanım amacı ve ilk akla gelen kısmı aslında bulutun bir depolama aracı olarak görülmesiydi. Teknolojinin gelişmesi ile birlikte bulut sadece depolama kabiliyeti olan bir sistemden artık donanımsal ihtiyaçlara tümüyle cevap veren

ve ihtiyaç duyulan veriyi hesaplayan ve işlenmiş veriyi kullanıcıya ileten bir sistem oldu. Bulut teknolojisinin kullanılabilmesi için tek başına ihtiyaç duyulan veriyi işleyip vermesi yeterli değil. İhtiyaç duyulan bu veriyi performans olarak da verimli bir şekilde sağlaması gerekiyor. Bu performansı da ölçeklendirebilmek gerekiyor. Bulut Bilişimin performansını ölçeklendirebilmek için görev zamanlama (task scheduling) algoritmalarında ki performansı incelenecektir. Araştırmanın konusu da bu doğrultuda geleneksel, sezgisel ve metasezgisel algoritmalar kullanılarak Bulut Sistemindeki performanslarının incelenmiştir ve performans tanımına güvenlik açısından da bakılarak yeni bakış açısı ve özgünlük getirilmek adına kuantum sonrası kriptolojiye değinilmiştir. Performanstan çıkan sonuçlar doğrultusunda daha verimli bir algoritma geliştirilmesine yönelik ve güvenlik anlamından çözüm önerileri sunmak olacaktır

1.2 Tezin Organizasyonu

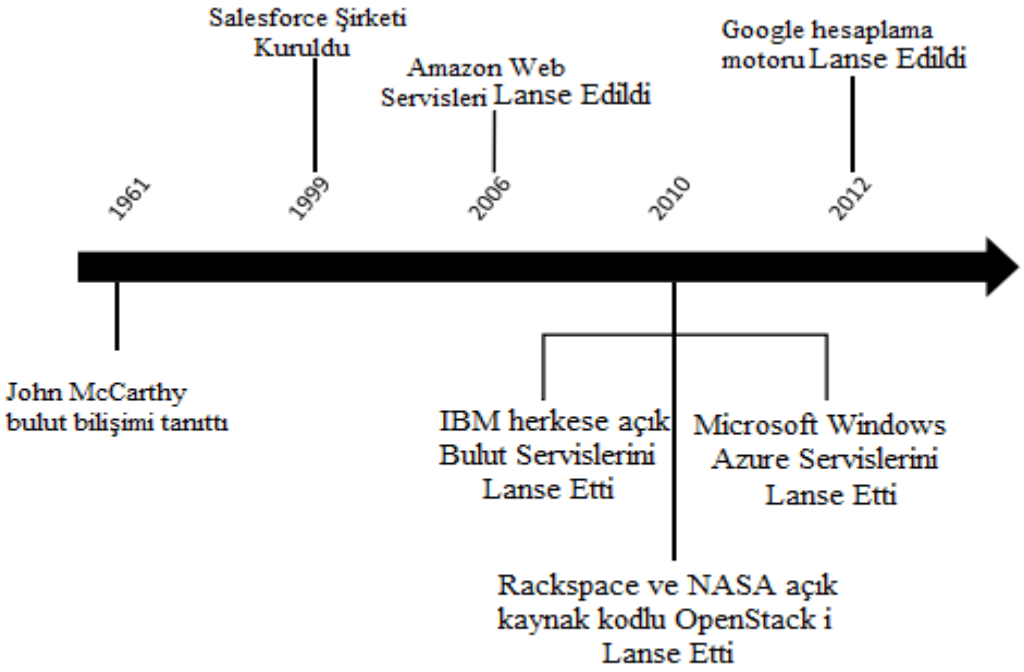
Tez çalışması 3 ana bölümden oluşmaktadır. Birinci bölümde Bulut Bilişim ile ilgili giriş düzeyinde bilgi verilmiş olup ön bilgilendirme yapılmıştır. İkinci bölümde Bulut bilişim ile ilgili literatür taraması yapılarak, çalışmada kullanılan temel terimler ve sistemler hakkında bilgi verilerek tanımlamalar yapılmıştır. Ayrıca Bulut Bilişimin güvenlik performansı farklı bir pencereden ele alınarak kuantum sonrası kriptolojiye değinilmiştir. Üçüncü bölümde ise kullanılan algoritmalar anlatılarak, algoritmaların CloudSim ile bulut simülasyon üzerinde performansları hesaplanmıştır. Hesaplanan performanslar kıyaslanmış olup çıkan sonuç neticesinde Bulut Bilişim için yeni algoritmalar geliştirilmesi anlamında önerilerde bulunmuştur.

2. KAYNAK ARAŞTIRMASI

Bulut Bilişim teknolojinin gelişmesi ile birlikte hayatımıza dahil olan yeni teknolojik gelişmelerden olup bu bölümde Bulut Bilişim sistemlerine ait literatür bilgilerine, ve incelenen konu dahilindeki temel bilgiler yer alacaktır.

2.1 Bulut Bilişim Sistemleri

Bulut Bilişim ile ilgili yapılmış olan kaynak araştırmaları incelendiğinde sadece Bulut Bilişim ile sınırlandıracak olursak 2000'li yılların başından itibaren değerlendirebilirken fikrin temeline inecek olursak, Bulut Bilişimin Başlangıcı Amerikalı bilgisayar bilimcisi John McCarthy'nin 1961 deki konuşmasına dayanır: Bu Konuşmada "Eğer savunduğum türdeki bilgisayarlar gelecekte olursa, o zaman bulut bilişim de bir gün telefon sisteminin genel bir hizmet olması gibi bilgisayar kullanımı ve hesaplaması da genel bir hizmet olacaktır. Bilgisayar hizmeti yeni ve önemli bir endüstrinin temeli olabilir." Demiştir (Garfinkel, S. 1999).Bulut bilişim ile ilgili kronolojik olarak atılmış önemli adımlar şu şekildedir.

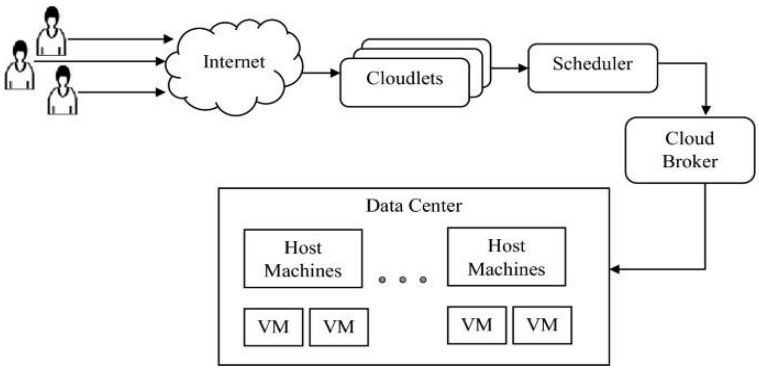


Şekil 2.1 Bulut Bilişim Kronolojik Sıralaması (Surbiryala, J. ve Rong, C. 2019).

"Bulut bilişim" kavramıyla çalışmaya başlayan ilk şirketlerden biri 1990'ların sonlarında Salesforce tarafından kuruldu (Surbiryala, J. ve Rong, C. 2019). Şirket, kullanıcılarına müşteri ilişkileri yönetimi sağlayan Yazılım Hizmeti (SaaS) sağlamaya başladı. Salesforce modeli, müşterilere Microsoft Azure ve Google'ın Uygulama Motoru gibi geliştirme platformları sağlayan "Platform Hizmeti (PaaS)"a ek olarak bulut bilişimin tipik modellerinden biridir (Surbiryala, J. ve Rong, C. 2019).

Diğer bir platform, 2006'da başlayan Amazon Elastic Compute Cloud (EC2) gibi "Altyapı Hizmeti (IaaS)" modelidir (Surbiryala, J. ve Rong, C. 2019). 2007'de ABD'deki birçok üniversite Google ve IBM ile iş birliği yapmaya başladı ve üniversitelerinde bulut bilişim programlarını teşvik etti. Bu, akademik araştırmanın maliyetini düşürmeye, kaynakları öğrenciler arasında paylaşmaya ve İnternet üzerinden erişmek için önemli bir işlem gücü veya bilgi işlem gücü oluşturmaya yardımcı oldu. Dünya genelindeki birçok üniversite daha sonraki yıllarda aynı eğilimi takip etti (Sultan, N. 2010).

Temmuz 2010'da NASA ve Rackspace, AMD, Intel ve Dell gibi çeşitli satıcılarla birlikte OpenStack adlı ortak bir proje başlattı. Daha sonra birçok başka kuruluş da projeye katıldı. OpenStack'i tanıtmak için Eylül 2012'de OpenStack Foundation adlı kar amacı gütmeyen bir kuruluş kuruldu. Şu anda 500'den fazla şirket projeyi destekliyor (Surbiryala, J. ve Rong, C. 2019). Yaklaşık 6800 şirket bulut hizmetlerini dağıtmak için OpenStack kullanıyor (Surbiryala, J. ve Rong, C. 2019).



Şekil 2.2 Bulut Bilişim Çalışma Mimarisi (Khan, M. S. A. ve Santhosh, R. (2022))

Bulut Teknolojisinin ilk kullanım amacı ve ilk akla gelen kısmı aslında bulutun bir depolama aracı olarak görülmesiydi. Teknolojinin gelişmesi ile birlikte bulut sadece depolama kabiliyeti olan bir sistemden artık donanımsal ihtiyaçlara tümüyle cevap veren ve ihtiyaç duyulan veriyi hesaplayan ve işlenmiş veriyi kullanıcıya ileten bir sistem oldu.

Bulut terimi bilgisayarla ilgili kaynakların havuzuna karşılık gelir (Khan ve ark. 2022). Havuz terimini örnekleyecek olursak, havuzu tarım arazilerini sulamak için kullanılan sulama havuzu olarak düşünülürse, bu havuzun tüm arazileri sulaması gerekir. Bu sulama sürecini yönetilmesi için bir çizelgenin, bir sıranın, bir kuralın ve yöntemin olması gerekir ki sulama sırasında haksızlıklar oluşmasın, herkes faydalanabilsin, herkes sırasına uysun ve bundan doğacak sorunlar olmasın. Çünkü her arazi aynı zamanda, aynı miktarda, aynı sefer su istemez. Kusursuz bir sıra ve çizelgeme yapmak gerekir ki suyu verimli kullanıp, verimliliği arazilere de yansıtıp doğru yöntemle maksimum ürün ve verimlilik alınsın. Aksi takdirde hiç sırası gelmemiş arazilerde susuzluktan dolayı verimsizlik hatta hiç ürün alamama, gerekenden çok daha fazla su verilmesi sonucu yine doğru kalitede ürün alınmaz. İşte bu yüzden bu Bulut hesaplamada bilgisayar kaynaklarının doğru bir şekilde planlanması ve tahsisinin yapılması gerekir. Bilgisayarların kaynak kullanımında üç etkene dikkat edilmelidir. İşlemci yoğunluklu, Depolama yoğunluklu ve Ağ trafiği yoğunluklu kullanımını iyi bir şekilde organize edilmelidir. Şöyle ki İşlemci yoğunluklu bir kullanım gerçekleştiğinde işlemci yuvalarında rekabet olacak ve diğer kaynaklar (bellek, disk vb.) kullanılmadan boşa harcanacak. Sonuç olarak hizmet kalitesi yüksek olmayacak ve darboğazlar oluşacak. Bu nedenle, diğer kaynakların kullanımı az olacağından bazı kaynaklar boşa harcanması nedeniyle enerji kaybı yaşanacak ve sistem verimsiz hale gelecektir. Sistemin verimliliği için temel kriter performans ve güvenlikten zafiyet vermeden maliyetleri düşürmek sektör için önem arz etmektedir.

Bulut Bilişimin tanımlandıktan sonra, bulut bilişimin avantajlarını ve dezavantajları şu şekildedir.

2.1.1 Bulut Bilişimin Avantajları

Bulut bilişimin kullanıcılara sağlamış olduğu avantajlar şu şekildedir.

2.1.1.1 Azaltılmış Maliyetler

Bulut sistemleri kullanıldığı zaman, sistemin yönetim ve bakım maliyetlerini azaltırken, pahalı sunuculara, yönlendiricilere, kablolamaya ve bu sisteme optimal çalışma ortamını sağlayacak bir alan ve ortama ihtiyaç kalamamaktadır.

Enerji maliyetlerinin ölçeklenmiş hali ile incelenecek olursa; yaygın bir 300W sunucu yılda yaklaşık 2628 KWH enerjiye tüketimi gerçekleştirirken ayrıca soğutma için fazladan 748 KWH elektriğe ihtiyaç duyuluyor (Wang X ve Chen M. 2008). Bulut

bilişim, ekonomik kılan faktörlerden bir tanesi enerji tüketimini azaltarak enerji maliyetinden tasarruf sağlamasıdır.

Ayrıca kullanılan cihazlar için sistem yükseltmelerine, güncel donanımsal geliştirmelere ve ek bireysel lisansların maliyetlerini azalmasına katkıda sağlar. Sadece cihazlar olarak kısıtlamamak gerekir. Sunucu ve sistemlerde Oluşabilecek sorunlarda müdahale edebilmesi, gerekli bakım ve güncelleştirmelerin gerçekleştirebilmesi için yetkin bilişim personelleri bulundurmak gerekmektedir. Bu durumda ek maliyet olarak hesaplandığında oluşan bu maliyetlerden minimum etkilenmemek bulut bilişim için çok önemli bir avantajdır.

2.1.1.2 Depolama Avantajı

Gelişen teknoloji ile birlikte artık herhangi basit bir verinin, belgenin boyutu nerdeyse bir dönemin işletim sisteminin veya sisteminin toplam depolama kapasitesine sahip olması her geçen gün tüm teknolojik ürünlerde depolama ihtiyacı artarak ve devam etmektedir. Depolamanın yanı sıra kullanıcılar tarafından depolanan verilere hızlı bir şekilde erişim talebi de eklenince, yine teknolojik gelişmelerinde etkisi mekanik depolama yöntemlerinin bırakılarak performans odaklı olarak elektronik depolama yöntemleri kullanana diskleri kullanılması da istenilen depolama alanın maliyetini yükseltmekte olup bazı durumlarda bu talebin karşılanmasını mümkün kılmamaktadır. Bulut Sistemlerinin sağlamış olduğu bu depolama avantajını kullanırken ihtiyaçları doğru belirler ve doğru araçlar ile kullanılırsa maliyet olarak avantaj sağlanabilir. Ayrıca depolanmış bu verilerin yedekleme ve geri yükleme süreçleri de yerel bir sisteme göre çok daha kolay ve performanslıdır.

2.1.1.3 Performans Avantajı

Her cihaz yeterli donanımsal özelliklere veya daha kapsamlı özelliklere sahip olmuyor bu gerek maliyet gerekse de cihazın kullanım alanı, enerji veya boyutsal olarak mümkün kılmıyor. Bulut sistemleri ile cihaz arasındaki bağlantı hızı optimum düzeyde sağlandığı takdirde yapılacak hesaplama veya işlemleri istenilen performansta gerçekleştirilebilir. Sistemdeki tüm cihazlar ihtiyaç duydukları kaynaklara erişmiş oldukları bulut sistemi ile ulaşmış oluyorlar. Buda performans anlamında ciddi bir avantaj sağlıyor.

2.1.1.4 Erişilebilirlik

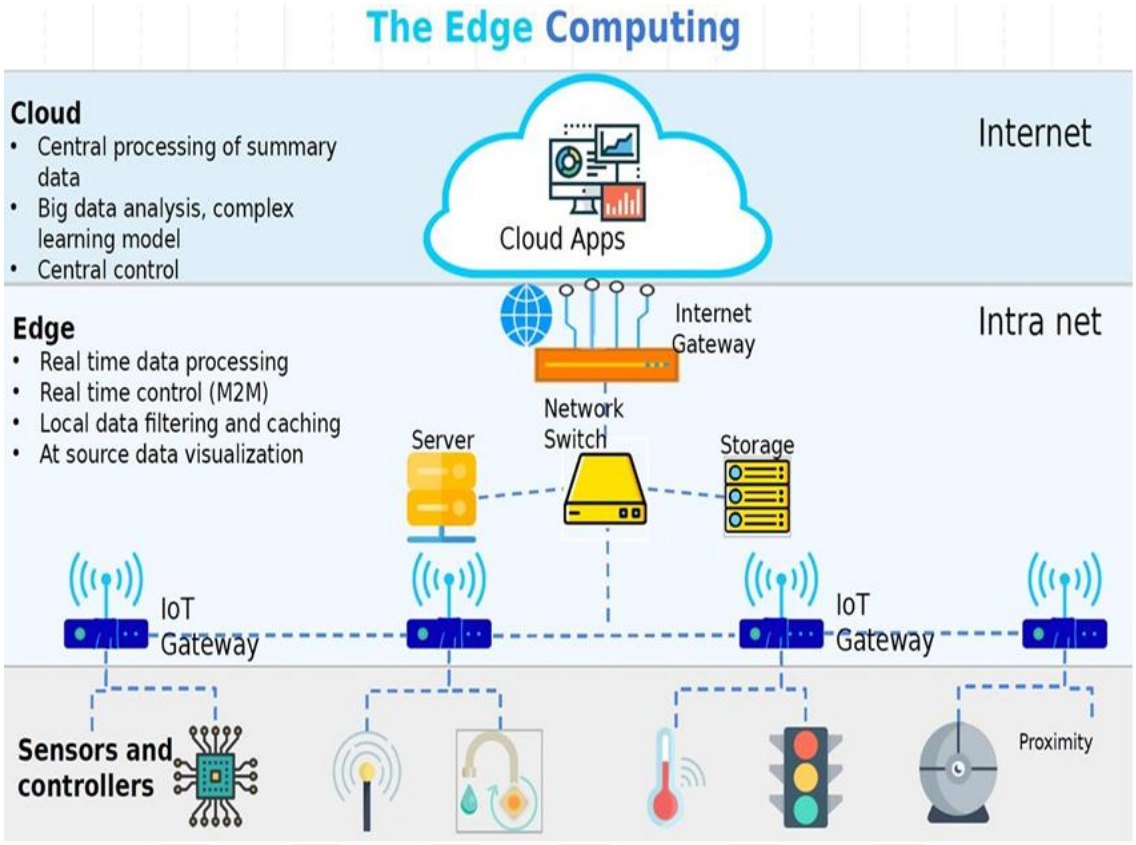
Bulut bilişimi sistemlerinin kurulum ve kullanıma alma süreci bir sistem odası hazırlama ve kullanıma alma süreçlerinde çok daha kolay ve kısa olması iş hayatında da erişilebilirliği yeni çalışma düzenlerinde tercih sebebi haline getirdi. Özellikle COVID-19 salgınından sonra eğitimden iş hayatına, alışverişten eğlenceye kadar birçok alanda yeni alışkanlıklar ve sistemlerin hayatımıza girmesi ile birlikte esneklik ve erişilebilirlik bir avantajdır. Bulut bilişim sistemlerinin mobil aygıtlar için de daha verimli ve etkili imkanlar sağlaması eğlence ve kişisel kullanımların dışında iş hayatında erişilebilirliğin mobiliteyi dönüşmesi de önemli bir avantajdır.

2.1.1.5 Verilerde Standartlaşma ve Sürdürülebilirlik

Seçilmiş olan Bulut sistemindeki veri işleme yöntemleri, yaklaşım modelleri ve depolama yöntemi ne ise kullanıcılarında bu düzeni benimsemek zorunda kalması sebebi ile tüm kullanıcıların bu doğrultuda işlem yaptığından standart haline belge ve veriler oluşmaktadır. Oluşabilecek kullanıcı kaynaklı eksiklikleri engellerken rutin güncellemelerde veya versiyon geçişlerinde standartlara göre testlerin gerçekleştirilmesi sebebi ile satabil çalışan sistemler sağlar. Sistemlerin bu yöntem ve modelleri çoğunlukla genel geçerdir. Bu yüzden standartlaşma sistem değişikliklerinde sürdürülebilirliği de sağlamaktadır. Ayrıca standartlaşmış ve hedeflenen verilere ulaşıldığı zaman raporlama ve analizlerin daha doğru çıkması sonucu yapılmış olan iş ve işlemlerin verimliliği de irdelenebilmektedir. Sistemlerin kesintisiz olarak devam ettirilebilmesi kullanıcılar için önemli bir avantajdır.

2.1.1.6 Nesnelerin İnternetine ve Akıllı Cihazlara İmkan Sağlaması

Literatüre bu alanda dahil olan gelişmelerden bir tanesi de Nesnelerin İnterneti. Nesnelerin İnterneti-IOT kullanılan araç gerecin her birinin bir bilişim cihazı olarak çalışarak veri alışverişi yapıp gerek veri toplama gerek kontrolünü sağlayan sistemlerdir. Bu tanımlamanın güncel terimlerinde akıllı ev teknolojileri, akıllı cihazlarda vardır. Bu tanımların bulut bilişim teknolojisi ise bu bağlantıya imkan sağlamasıdır. Nesnelerin internetinde bağlantı türleri farklılık göstermektedir fakat Nesnelerin İnterneti de ayrı bir tez konusu olacak bir konu olduğundan dolayı detaylarına girilememiştir.



Şekil 2.3 Nesnelerin İnterneti için Edge Computing Mimarisi (Hakan Uzuner 2020)

2.1.2 Bulut Bilişimin Dezavantajları

Bulut bilişimin kullanıcılara sağlamış olduğu avantajları yanında ortaya çıkan dezavantajları şu şekildedir.

2.1.2.1 Ağa Bağımlılık ve Bağlantı Problemleri

Bulut Bilişimin en büyük dezavantajı bağlantı ile ilgili, çünkü çevrimiçi çalışabilen bir sistem olduğundan, sistemin en belirgin ihtiyacı bağlantıdır. Bağlantı hızının yeterli olmaması, altyapının gerekli bant genişliğini sağlayamaması bu alanda oluşabilecek teknik problemler en büyük dezavantajdır. Bağlantı ile ilgili problemler sonucu sistemin avantajlarını kullanıcıya yansıtamadığı durumlar oluşmaktadır. Şöyle ki performans odaklı kurgulanmış olan verimli bir sisteme bağlantı hızı yeterli olmadığı veya bağlantı problemleri yaşandığı zaman performans alamayacağı için sistem verimsiz gözükebilir. Bu yüzden Bulut Bilişim sistemi kurgulanırken bağlantı durumu ve imkanları düşünülerek Sistem kullanılmalıdır.

2.1.2.2 Veri Güvenliği

Veri güvenliği konusu bulut sistemleri için gri bir konudur. Şöyle ki güvenlik konusu avantajdır da söylenebilirken dezavantajıdır da diyebiliriz. Ticari olarak veya bireysel olarak ciddi bir kuruluşa ait bulut bilişim ürünü kullanılırken kuruluş tarafından çok daha etkin ve profesyonelce güvenlik tedbirleri, sistemleri kullanılabilir. Veri Güvenliğinin yönetimi için alanında uzman yetkin personeller ile bu süreç avantaja dönüşürken kullanıcı hatası kaynaklı tüm sisteme sızma sonucu veri güvenliği ihlaline sebep olup dezavantaja dönüşebilmektedir.

2.1.3 Bulut Bilişim Hizmet Modelleri

Bulut Bilişim Hizmet Modellerinden yaygın olan üç hizmet modeli şu şekildedir.

2.1.3.1 Altyapı Hizmeti (IaaS)

Kullanıcılara sanal olarak ihtiyacı olan depolama, sunucu ve ağ ekipmanları gibi bilgi işlem kaynaklarına erişim imkanı sağlayan bulut bilişimin temelini oluşturan hizmet modelidir. Kullanıcılara altyapı gereksinimlerini karşılayacak şekilde kaynak tercihlerini yapabilme esnekliği sunmaktadır. Sunulan bu altyapının yönetimi kullanıcıya ait olup buda kullanıcılara esneklik sağlamaktadır. Altyapı hizmetlerine örnek olarak Windows Live Services, Amazon Elastic Compute Cloud (EC2), Carbonite, Ibackup, Joyent, Rackspace Hosting, GoGrid, OpenStack, OpenNebula ve Eucalyptus verilebilir.

2.1.3.2 Platform Hizmeti (PaaS):

Kullanıcılara internet üzerinden kendi uygulamalarını geliştirebileceği bir tür bulut hizmeti modelidir. Kullanıcılara ihtiyaçları doğrultusunda altyapıya ait sunucuları, depolama cihazlarını, ağ alt yapısını ve işletim sistemini kullanabilmesi ve çalıştırabilmesi için gerekli olan platformu sağlar. Bu sayede kullanıcılar bu platformda donanımsal ve altyapı süreçlerini uygulama geliştirebilir, test edebilir ve çalıştırabilir. Platform bulut hizmet modeli kullanıcılara otomatik ölçeklendirme ve yük dengeleme sunarak kesintisiz bir şekilde artan trafiği yönetebilme imkanı sağlar. Platform hizmetlerine örnek olarak Windows Azure, Google App Engine, Force.com ve CloudBees verilebilir.

2.1.3.3 Yazılım Hizmeti (SaaS):

Kullanıcılara internet üzerinden web sayfası veya ayrı ile arayüz ile hedefteki program veya sisteme erişim imkanı sağlayan son kullanıcılara yönelik bulut hizmet modelidir. Bu sayede kullanıcılar CRM, ERP ve HRMS gibi yazılım uygulamalarına erişim sağlayabilmekte olup sektörel olarak avantaj sağlamaktadır. Son kullanıcılara yönelik oldğundan Bulut Bilişim denildiği zaman ilk akla gelen gerek bireysel gerek kurumsal gereksinimlere olanak sağlayan hizmet modelidir. Yazılım hizmetlerine örnek olarak Google Docs, Office 365 , Salesforce CRM ve SAP ERP verilebilir.

Bulut Bilişimin hizmet modellerinin kullanımına örnekler şu şekildedir.

Şekil 2.4 Bulut Bilişim Hizmet Modelleri Kullanım Örnekleri (Bala ve ark (2021))

Eğer bulut bilişim kullanılmayıp yerel olarak işletme içerisinde konumlandırılmış bir sistem kullanılıyor ise tabloda belirtilmiş olan tüm süreçleri kullanıcı kendisi yürütülmek zorunda kalırken, Bulut Bilişimin hizmet modellerinin kullanıldığında ise alınan hizmete göre turuncu renkteki süreçleri kullanıcının kontrolünde iken yeşil renkli süreçler bulut sağlayıcı tarafından gerçekleştirilecektir.

Tablo 2.1 Bulut Bilişim Hizmet Modelleri Kullanımının Sağlamış Olduğu Avantajlar

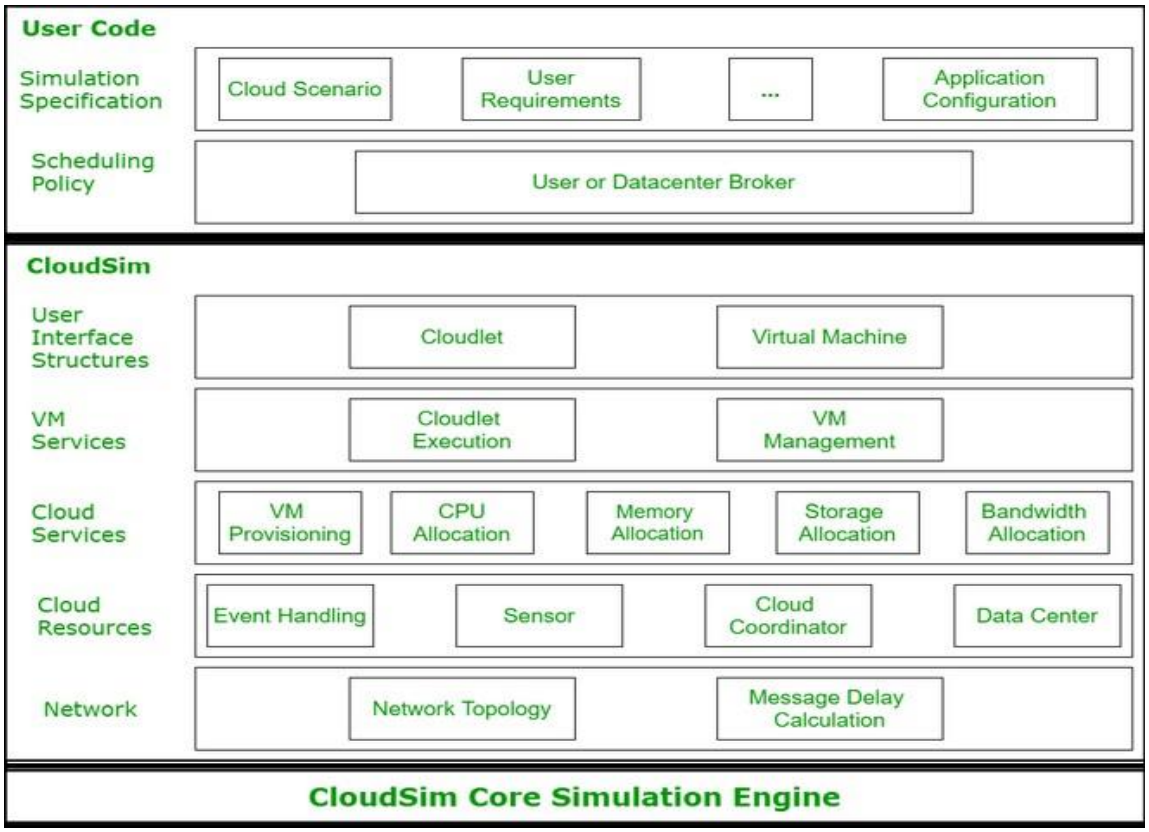
İşletme İçerisinde	Alt Yapı Hizmeti	Platform Hizmeti	Yazılım Hizmeti
Uygulamalar	Veri	Veri	Veri
Veri	Uygulamalar	Uygulamalar	Uygulamalar
Çalışma Zamanı	Çalışma Zamanı	Çalışma Zamanı	Çalışma Zamanı
Orta Katman	Orta Katman	Orta Katman	Orta Katman
İşletim Sistemi	İşletim Sistemi	İşletim Sistemi	İşletim Sistemi
Sanallaştırma	Sanallaştırma	Sanallaştırma	Sanallaştırma
Sunucular	Sunucular	Sunucular	Sunucular
Depolama	Depolama	Depolama	Depolama
Ağ İletişimi	Ağ İletişimi	Ağ İletişimi	Ağ İletişimi
Kullanıcı tarafından kontrol edilecek sistemler		**Bulut Sağlayıcısı tarafından kontrol edilecek sistemler**	

2.1.4 Cloudsim

Araştırmamızda Bulut Bilişim ile ilgili hizmet sunan sistem ve ticari hizmet servisleri ve OpenStack gibi açık kaynaklı sistemler araştırılmış ve ücretli sistemler olması sebebi ile ücretsiz olan OpenStack gibi açık kaynak kodlu (open source) sistemlerin ise başlangıç için karmaşık ve zor yapıda olmasına sebebi ile CloudSim kullanılmıştır. CloudSim *açık kaynaklı bulut hesaplamasının altyapısı ve servislerini içeren bir simülatördür.* CLOUDS Lab tarafından Java dilinde geliştirilmiştir. Bulut sistemleri üzerinde araştırma ve akademik çalışma gerçekleştirmek için iyi bir alternatiftir. Cloudsim'in Java ile geliştirilmiş olması ve Javanın nesne yönelimli ve yaygın bir dil olması sebebi ile

araştırmacılara bu anlamda kullanım kolaylığı sağlamaktadır. CloudSim i iki katmanlı bir yapıya sahiptir.

1. CloudSim katmanı; Sanal Makineler (VirtulMachine), Cloudlet'ler, Ana Bilgisayarlar gibi temel varlıkların yani kaynakların oluşturulmasını ve yönetimi ile birlikte ağla ilgili yürütülmenin de sağlandığı katmandır
2. Kullanıcı Kodu katmanı; Kullanıcı tarafından kontrol edilen senaryoya göre donanımsal özelliklerin ve gereksinimlerini oluşturulduğu katmandır.



Şekil 2.5 Cloudsim Mimarisi (Anonim, 2021)

Simülasyon sırasında kullanılan en yaygın sınıflardan bazıları şunlardır:

- Datacenter: Herhangi bir bulut ortamının temel donanım ekipmanını, yani Veri Merkezini modellemek için kullanılır. Bu sınıf, Veri Merkezi'nin işlevsel gereksinimlerini belirlemeye yönelik yöntemlerin yanı sıra Sanal Makinelerin tahsis edilmesini belirlemeye yönelik yöntemler sağlar.

- Host: Bu sınıf, sanal makinelerin yönetimi ile ilgili eylemleri yürütür. Ayrıca, sanal makinelere bellek ve bant genişliği sağlamanın yanı sıra sanal makinelere CPU çekirdekleri tahsis etmeye yönelik tanımlamaları gerçekleştirir.
- VM: Bu sınıf, bir Sanal makinenin bant genişliğini, RAM' ini ve MIPS boyutunu tanımlayan verileri sağlarken aynı zamanda bu parametreler için ayarlayıcı ve alıcı yöntemleri sağlayarak bir sanal makineyi temsil eder.
- Cloudlet: Cloudlet sınıfı, işleme görevini, bellek erişim görevini veya dosya güncelleme görevi gibi bir sanal makine üzerinde çalıştırılan herhangi bir görevi temsil eder. Sanal Makineler sınıfına benzer yöntemler sunarken aynı zamanda bir görevin yürütme süresini, durumunu, maliyetini ve geçmişini tanımlayan yöntemleri sağlar. Testler de bu bölümün verileri kullanılır
- DatacenterBroker: Sanal Makine oluşturma, yönetme, yok etme ve cloudlet'ların sanal makineye sunulması dahil olmak üzere Sanal Makinelerin işleyişinden sorumludur.
- CloudSim: Gerekli tüm bulut varlıkları tanımlandıktan sonra simülasyon ortamının başlatılmasından ve daha sonra tüm varlıkların yok edildikten sonra durdurulmasından sorumlu sınıftır.

2.2 Görev Zamanlama:

Gerek güvenlik gerekte performans zafiyeti oluşturmadan enerji maliyetlerini de azaltarak kaynaklarımızı da verimli kullanmak için uygun bir algoritmaya ihtiyaç duyulmaktadır. İşte bu süreçleri çizelgelemek, planlamak ve organize etmek için ihtiyaç duyulan doğru ifade görev zamanlama algoritmalarıdır.

Görev zamanlaması (task scheduling), verimli bir algoritma kullanarak istemcilerin görevlerini mevcut ve uygun sanallaştırılmış kaynaklarla eşleştirmek için kullanılan bir tekniktir (Kanani, B. ve Maniyar, B. 2015).

Zamanlama (Scheduling), süreçlerin mevcut Merkezi İşlem Birimi'nde (Central Processing Unit) (CPU) çalışacak şekilde atanma biçimini ifade eder (Mora ve ark. 2017). Belirli CPU planlama teknikleri veya algoritmaları geliştirilmiştir. Bu algoritmaların genel amacı, CPU yu en üst düzeyde verimli kullanmak, ortalama bekleme süresini (average waiting time) (AWT) ve ortalama geri dönüş süresini (average turnaround time)

(ATAT) en aza indirerek, verimi artırmayı amaçlamaktadır. Bulut bilişim gibi heterojen ve dağıtılmış bir ortam olduğu için görev zamanlama sorunu daha zor hale gelir. Bu nedenle, sistemin performansı için anahtar olarak kabul edilen etkili bir görev zamanlama algoritmasına ihtiyaç vardır (Kanani, B. ve Maniyar, B. 2015). Bulut bilişim ile ilgili halen en çok tartışılan ve aşılamayan sorunların başında herkes tarafından kabul edilmiş, tam manasıyla sunabileceği ve vaat ettiği performansı bulut sistemlerinde kanıtlanmış ve yansımış bir görev zamanlama algoritması modeli henüz yoktur. Bulut sistemlerinde, görev planlama algoritmalarının üç yaygın kategorisi vardır (Soltani ve ark. 2017).

2.2.1 Geleneksel Algoritmalar

Geleneksel algoritmaların tasarımında detaylar yerine daha sade genel geçer kuralların ve daha basit matematiksel işlemlerin algoritmaya dönüşmüş halleridir. Gündelik hayatta dahi elinizdeki işleri sıraya koymak için ilk akla gelen, kullanılan ve işe yarayan bu yöntemler bilgisayar bilimleri için de görev zamanlama algoritmaları olarak tasarlanmış olup etkin şekilde de kullanılmaktadır. Bazı geleneksel algoritmalar şu şekildedir.

- First Come First Serve (FCFS),
- Shortest Job First (SJF),
- Longest Job First (LJF)
- Round Robin (RR)

2.2.2 Sezgisel Algoritmalar

Sezgisel kelimesi eski Yunancadan "heuriskein" kelimesinin karşılığı olarak, problemleri çözmek için yeni yöntemler geliştirme anlamının karşılığıdır (Ç. Aladağ, 2009). Optimizasyon problemlerinde sezgisel, optimale yakın sonuçlar bulan, stratejilere dayalı, hızlı ve pratik bir çözüm olarak kabul edilmektedir. Problem optimum çözümü bulunamayacak kadar karmaşıksa, sezgisel yöntemler sezgiye veya bazı deneysel kayıtlara dayanan karar kuralları ile belirli sayıda adımdan sonra en iyi olmasa da tatminkar bir sonuç verirler (Kaya ve Fığlalı, 2016). Problemi çözmek için aslında bir yaklaşım gerçekleştirilmektedir. Bu yaklaşımlarımı optimizasyon temellerini kullanarak veya probleme örnek olacak durumlar üzerinden yaklaşımlar gerçekleştirerek çözüme yönelinmektedir. Bazı sezigisel algoritmalar şu şekildedir.

- A Star Arama Algoritması

- Demet Araması (Beam Search)
- Tepe Tırmanma Algoritması (Hill Climbing)
- En İyi Öncelikli Arama Algoritması (Best First Search)
- Açgözlü Yaklaşımı Algoritması (Greedy Best First Search)
- Benzetimli Tavlama Algoritması (Simulated Annealing)
- Geri İzleme Algoritması (Backtracking)

2.2.3 Metasezgisel Yöntemler

Meta kelime manası olarak "daha ileri" veya "üst seviye" anlamına gelmektedir. Sezgi üstü algoritmalar sezgisel algoritmalara göre sınırları daha geniş kullanım alanı bulunmaktadır. Literatürde bazı problemlerde sezgisel performans olarak iki yönlü olarak hem daha iyi hem daha kötü performans sergilediği çalışmalar bulunmaktadır. Sezgisel algoritmalara göre en büyük artısı uyarlanabilirlik olduğu düşünülmektedir. Daha fazla probleme çözüm olarak hükmedebilmektedir.

Hem sezgisel, hem metasezgisel algoritmaları birer yaklaşım algoritmalarıdır. Yaklaşımları genel olarak başka problemlere benzeterek, bu problemlerin çözüm yöntemleri düşünülerek ve benzetilerek aramaktadır. Burada benzetilen örnek alınan kimi zaman insan, kimi doğada yer alan bazen hayanı, bazen bir sürü bazen evren örnek alınmıştır. Bu durum tanımlanacak olursa günümüzde kullanılan birçok araç, düşünce ve teori gerek tabiatı gerek hayvanları gerek ise insanları gözlemleyerek oluşturulmuş veya taklit edilmiştir. Buna bilimsel olarak biyomimikri denir. Biyolojik sistemlerden esinlenerek bilişim problemlerin çözümünde kullanılan birçok yöntem ortaya konulmuştur. Örneğin, yapay sinir ağları insan beyninin basitleştirilmiş bir modelidir. Bir çok algoritma bu yöntem dahilinde geliştirilmiştir. Bazı metasezgisel algoritmalar şu şekildedir.

- Genetik Algoritma (GA)
- Karınca Kolonisi Optimizasyonu (Ant Colony Opt.) (ACO)
- Parçacık Sürü Optimizasyonu (Particle Swarm Opt.) (PSO)
- Yapay Arı Kolonisi Optimizasyonu (Artificial Bee Colony Opt.) (ABC)
- Diferansiyel Gelişim Algoritması (Differential Evolution Algorithm) (DEA)

- Benzetim Tavlama Algoritması (Simulated Annealing) (SA)
- Yerçekimi Arama Algoritması (Gravity Search Algorithm) (GSA)
- Gaz Brownian Hareketi Optimizasyonu (Gases Brownian Motion Opt.) (GBMO)
- Isı Transferi Arama (Heat Transfer Search) (HTS)
- Elektromanyetik Alan Optimizasyonu (Electromagnatic Field Opt.)(EFO)
- Optikten Esinlenen Optimizasyon (Optic Inspired Optimization)(OIO)
- Ağırlıklı Süperpozisyon Çekimi (Weighted Superposition Attraction (WSA)
- Orman Optimizasyonu Algoritması (Forest Optimization Algorithm)(FOA)
- Kasırga Temelli Optimizasyon Algoritması (Hurricane Based Optimization Algorithm)
- Kara Delik Optimizasyon Algoritması
- Su Döngüsü Optimizasyon Algoritması
- Meyve Sineği Optimizasyon Algoritması
- Krill Sürü Optimizasyon Algoritması
- Bakteri Yiyecek Arama Davranışı
- Yarasa Algoritması
- Ateş Böceği Algoritması
- Aslan Algoritması
- Gri Kurt Algoritması
- Yunus Balığı Algoritması
- Çalı Kolonisi Algoritması
- Yapay Alg Algoritması
- Virüs Koloni Arama Algoritması
- Köpekbalığı Koku Alma Optimizasyon Algoritması
- Sosyal Örümcek Algoritması

- Ağaç-Tohum Algoritması (Tree-Seed Algorithm) (TSA)

Görüldüğü üzere meta sezgisel algoritmaların geliştirilmekte ve sayısı artmaktadır.

2.2.4 Hibrit Yöntemler

Aynı problem için tasarlanmış iki veya daha fazla algoritmayı sentezleyerek probleme daha iyi sonuçlar getirmeyi hedefleyen algoritmalardır. İki algoritmanın birleşmesi yerine sentezlenmesi ifadesi kullanılması sebebi tamamen birleşmek yerine algoritmaların ilgili probleme karşı etkili olan fonksiyon ve özelliklerinin gelen veriler doğrultusunda kullanılarak algoritmaların bireysel olarak sağladığı performanstan daha iyi sonuç almak hedeflenir. Hibrit tanımı da tam olarak iyi özelliklerin seçilerek ortaya daha verimli yeni türlerin çıkarılmasıdır. Bazı hibrit algoritma örnekleri şöyledir.

- ACO + PSO
- GA + BF
- ABC + PSO
- SA + PSO

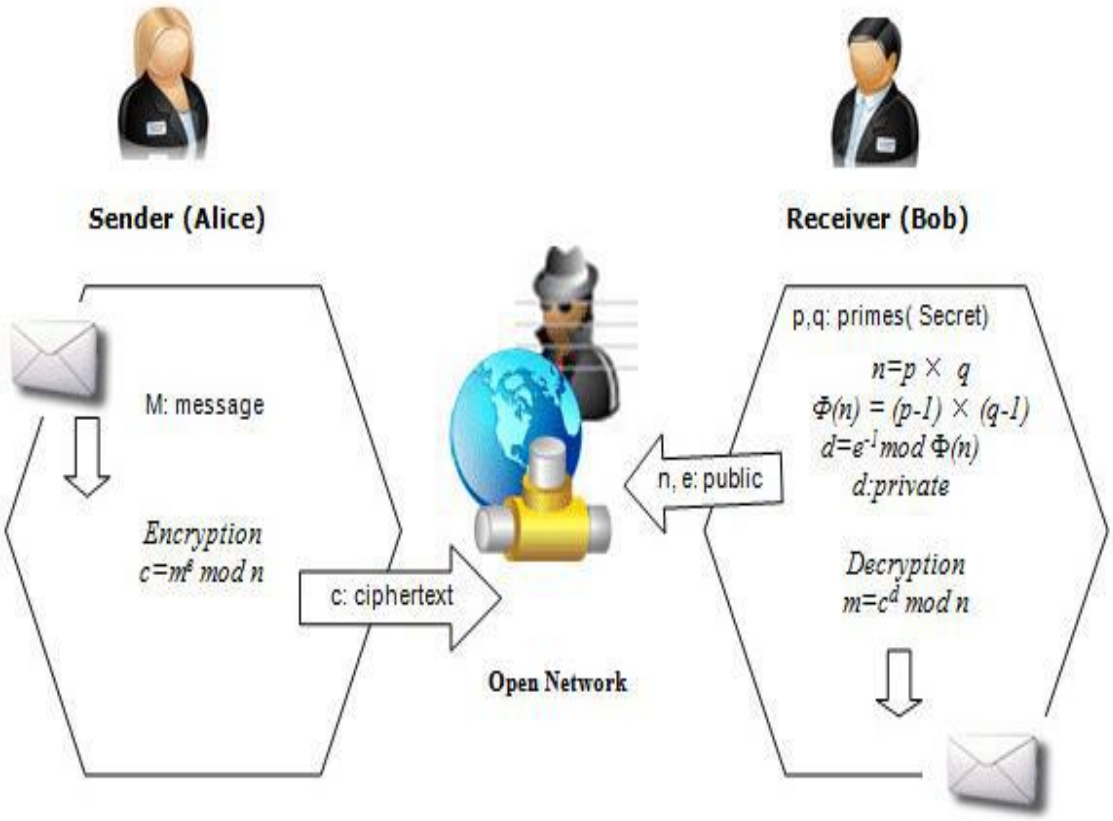
2.3 Kuantum Sonrası Bulut Bilişimde Güvenlik

Tez konusu olarak bulut bilişimin görev zamanlama algoritmalarında ki performansı incelenirken performans tanımını daha genel olarak düşünülmelidir. Performans değerlendirmesine güvenlik performansı da dahil olmalıdır. Çünkü istenilen süreç olarak performansı sağlarken güvelik olarak zafiyet vermesi performans olarak yeterli ve verimli anlamına gelmiyor. Bu yüzden performans incelemesi yapılırken güvenlik alanı da ele alınmıştır.

Bulut bilişim ile ilgili güvenlikten bahsederken gelişmekte olan, ilgi gösterilen ve her geçen yıl daha çok kullanılan ve bu yüzden daha uzun yıllarda kullanılacak bir teknoloji olduğu için, güncel güvenlik teknolojileri yerine kuantum sonrası için bulut bilişim sistemlerinin güvenlik durumu düşünülerek yeni bir bakış açısı getirilmeye çalışılmıştır. Bu bağlamda öncelikle güncel güvenlik tanımlamaları yapılarak kuantum sonrası kriptoloji geçiş yapılmıştır ve açık anahtarlı sistemler ve imzalama şemaları ele alınmıştır.

Açık Anahtarlı Sistemler, internet ve elektronik imzalama işlemlerinde önemli bir yer teşkil etmektedir. Açık anahtarlı sistemlerin çalışma mantığı şu şekildedir.

Açık anahtarlı (Asimetrik) sistemlerde şu an en çok kullanılan sistem RSA dır ve RSA şu şekilde çalışmaktadır.

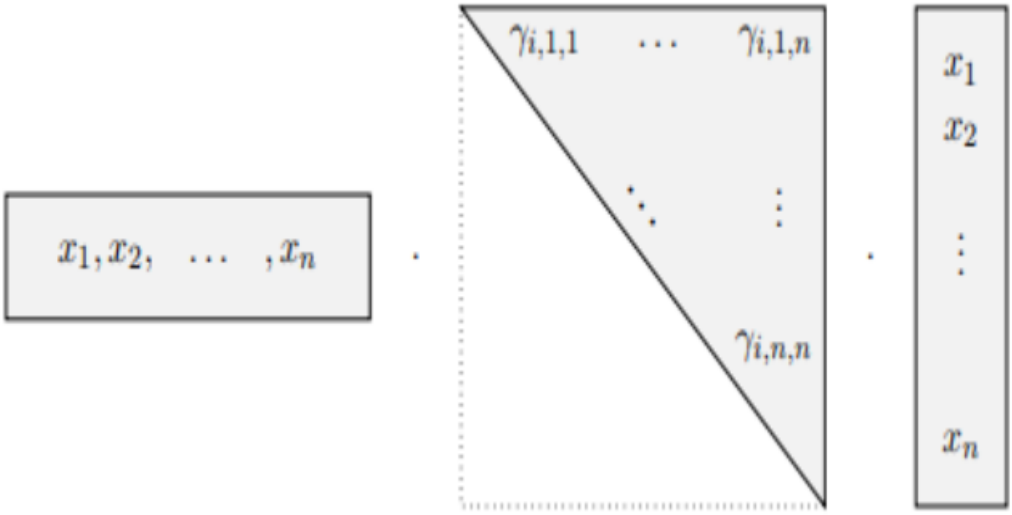


Şekil 2.6 RSA İmzalama Şeması (Al-Anie ve ark (2011))

RSA şifrelemesinin güvenliği de çarpanlara ayırmanın zorluğuna dayanmaktadır. Şu an mevcut RSA sistemleri 1024 bitlik gizli anahtar (p,q,d) ve 2048 bitlik açık anahtar (n) kullanmaktadır. Şu an mevcut cihazlar ile ve hatta süper bilgisayarlar ile bile 2048 bitlik sayının iki asal çarpanı olan p q anahtarlarını bulmak çok uzun süreler almaktadır. Bu yüzden de gizli anahtarlara ulaşmak bir hayli zordur ve geçerlilik süresi içerisinde çözülemediği için güvenli kabul edilmektedir. Fakat kuantum bilgisayarlar ile bu zorluk da ortadan kalkıyor. Geçerlilik süresi içerisinde çözüm sağlayacak performans sağlamaktadır. Kuantum Sonrası İçin güvenlikte en güçlü aday olarak Çok Değişkenli Kuadratik Kriptoloji (Multivariate Quadratic Cryptography) gösterilmektedir (Duong, D. ve ark 2017).

2.3.1 Multivariate Quadratic Polynomials

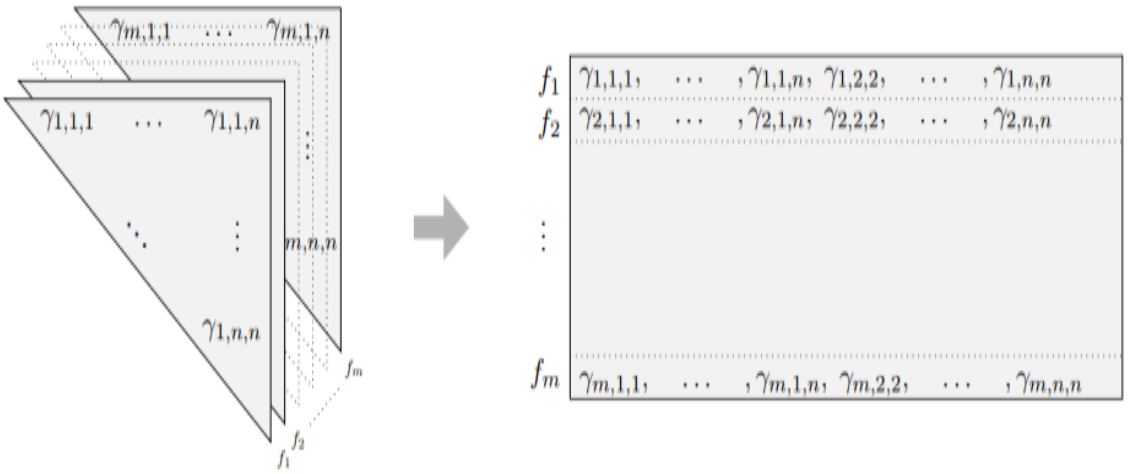
Güvenliği artırmak için Çok Değişkenli Kuadratik Polinomlar kullanılarak kuantum bilgisayar ataklarına karşı direnç sağlanması hedeflenmektedir. Çok Değişkenli Kuadratik Polinomlar da polinomlar birden çok değişkene bağlı ve polinomlar çoğunlukla ikinci dereceden kullanılarak güvenlik seviyesi arttırılmaktadır. Polinomların anlaşılabilmesi için geometrik şekiller kullanılarak katsayılar matrisi Şekil 2.7 de olduğu gibi gösterilmektedir (Czypek, P. 2012).



Şekil 2.7 Çok Değişkenli Kuadratik Polinomlar Katsayılar Matrisi Gösterimi (Czypek, P. (2012))

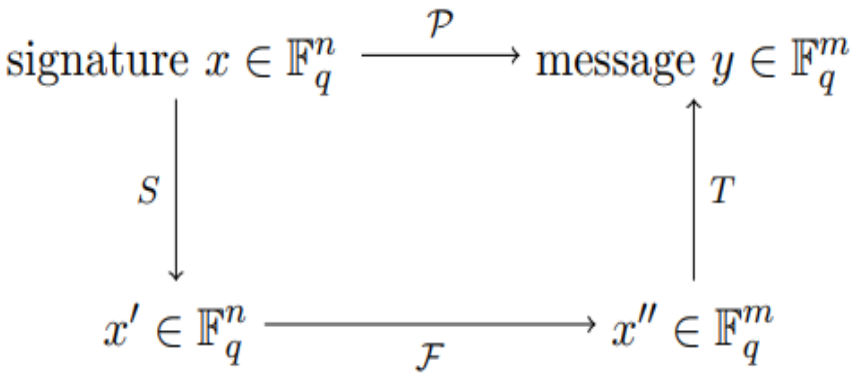
Çok Değişkenli Kuadratik Polinomlar Katsayılar Matrisi Denklemi şu şekildedir.

$$\triangleq f_i(x_1, \ldots, x_n) = \sum_{1 \le j \le k \le n} \gamma_{i,j,k} x_j x_k$$



Şekil 2.8 Çok Değişkenli Kuadratik Polinomlar Katsayılar Matrisi Gösterimi (Czypek, P. (2012))

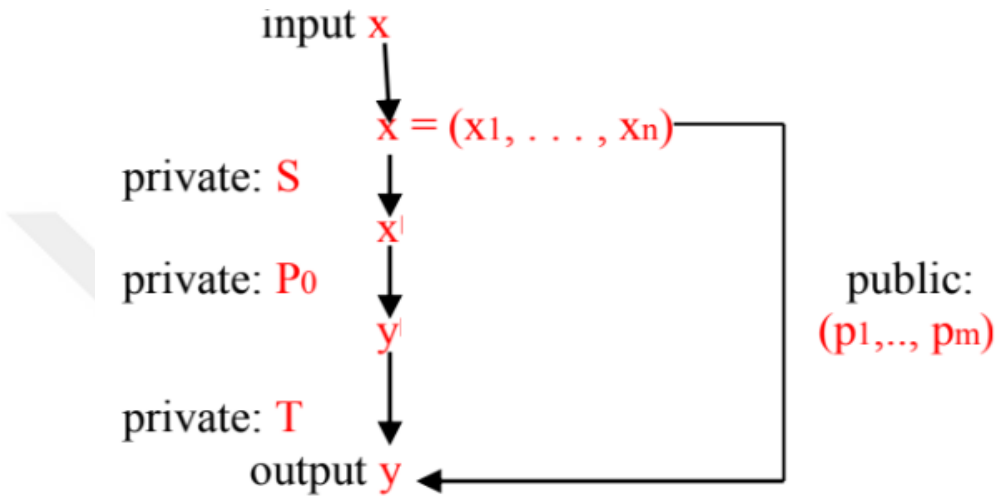
Multivariate Quadratic sistemlerde küçük haritalarda dönüştürmek kolay fakat büyük haritaları dönüştürmek zor olduğu için birbirleri ile ilişkili 2 tanede doğrusal harita, 1 tane de merkezi harita oluşturulmaktadır. T haritası mesajı kabul etmekte S haritası da son mesaj değişkenini sağlamaktadır (Czypek, P. 2012). Burada harita diye belirtilen yapı katsayılar (değişkenler) matrisidir. Bu matrisler de macaulay matrisleridir.



Şekil 2.9 Çok Değişkenli Kuadratik Polinomlar İmzalama Süreci (Czypek, P. (2012))

Bu şekilde lineer dönüşümler yapılmaktadır.

$$\mathcal{P} = T \circ \mathcal{F} \circ S$$



Şekil 2.10 Çok Değişkenli Kuadratik Polinomlar Lineer Dönüşümü (Czypek, P. (2012))

Dönüştürürken de;

$$T^{-1}(y)=x'' \longrightarrow \mathcal{F}^{-1}(x'')=x' \longrightarrow S^{-1}(x') = x$$

şeklinde olmaktadır.

2.3.2 Unbalance Oil and Vinegar Scheme (UOV)

Unbalance Oil and Vinegar Scheme (UOV) 1999 da Kipnis, Patarin, Goubin tarafından tanıtılmıştır. Unbalance Oil and Vinegar Scheme (UOV) 'ın ismi değişkenlerin iki grup halinde vinegar ve oil olarak gruplandırılmasından gelmektedir. Bu isimlerin özelliklerini taşımaktadır. Gerçek hayatta da yağ ve sirkeyi aynı kaba konulduğu düşünülürse sirke altta kalır yağ da üstünde olur eğer karıştırılmaz ise birbirleri ile ayrışık bir şekilde dururlar. UOV şemasında da ilk başta bu şekilde ayrı duruyor katsayılar ve çok değişkenli

kuadratik polinomlar ve bu katsayılar kullanılıyor. Yine gerçek yağ ve sirke karışımı düşünülürse, bu sefer karışım karıştırılırsa ve yağ ve sirke birbirine karışır, artık bu karışımı ayrıştırmak için özel yöntemler gerekir ve bu süreç de zordur. Çok değişkenli kuadratik polinomlarda da bu oil ve vinegar değişkenlerini merkez polinomda karıştırıldıktan sonra aynı yağ ve sirke karışımı gibi bu değişkenleri bulabilmek yani bu imzalama ve şifreleme işlemini kırabilmek çok ciddi bir şekilde zorlaşıyor.

Bu şemaların iki türü bulunmaktadır. Dengeli ve Dengesiz olarak ayrılmaktadır.

Dengeli Şemalarda (Balance Oil and Vinegar Scheme) oil ve vinegar değişkenleri miktarı eşit sayıda yer almaktadır. Değişkenlerin eşit miktarda yer alması güvenlik açığı yaratmaktadır. Bu katsayıların eşit miktarda bulunması tahmin edilebilmeyi ve hesaplanabilmeyi kolaylaştırıldığından daha kolay kırılmasına sebep olmaktadır. Bu yüzden de bu yöntem 1998 yılında Kipnis Shamir tarafından kırıldı.

Dengesiz Şemalarda (Unbalance Oil and Vinegar scheme (UOV)) oil ve vinegar değişkenlerinin sayısı eşit miktarda yer almamaktadır. Vinegar değişkeni sayısı daha fazla olmaktadır fakat bu miktar çok da fazla olmayarak değişkenler arasında ki fark yakın tutulmaktadır (How HT ve ark 2006). Aynı gerçek yağ ve sirke karışımının da eğer sirke miktarı çok fazla miktarda yer alırsa, örnek olarak sirke yağın 10 katı o zaman bu karışım artık nerdeyse tamamen sirkeden oluşmakta olup artık bu karışım olmaktan çıkmaktadır. Dengesiz şemalar da durum ideal miktarları ile şu şekildedir ;

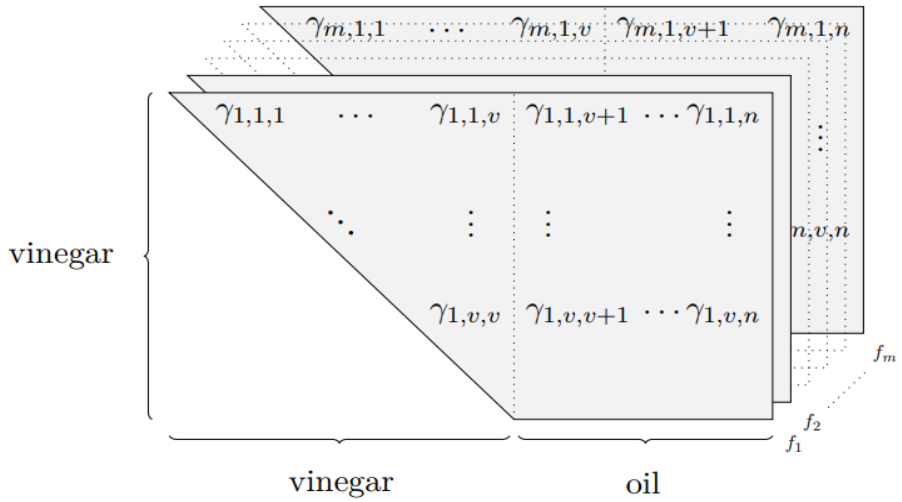
• Genel yapı N=V+O şeklinde olan sistemlerde

• V=2o veya V=3o miktarında seçilmektedir (How HT ve ark 2006).

Unbalance Oil and Vinegar Scheme denklemleri şu şekildedir.

$$\int_k (x_1, \dots, x_n) := \sum_{i \in V, j \in V} \gamma_{k,i,j} x_i x_j + \sum_{i \in V, j \in O} \gamma_{k,i,j} x_i x_j$$

$$p(x_1, \dots, x_n) = \underbrace{\sum_{i=1}^{v} \sum_{j=i}^{v} \alpha_{ij} \cdot x_i \cdot x_j}_{v \times v \text{ terms}} + \underbrace{\sum_{i=1}^{v} \sum_{j=v+1}^{n} \beta_{ij} \cdot x_i \cdot x_j}_{v \times o \text{ terms}} + \underbrace{\sum_{i=1}^{n} \gamma_i \cdot x_i}_{\text{linear terms}} + \delta$$



Şekil 2.12 UOV Katsayılar Matrisi Gösterimi (Czypek, P. (2012))

Dengesiz Şemalar (Unbalance Oil and Vinegar scheme (UOV)) 1999 yılından beri kırılamıyor (Yang, B. Y.).

Avantajı

- Oldukça güvenli olması

Tablo 2.2 de gösterilen güvenlik parametrelerinden de anlaşılacağı üzere verimli değil, verimlilik için sadece güvenli olması yetmiyor ayrıca performans açısından da iyi olması gerekmektedir (Yang, B. Y.).

- Şemaların hızlı olmaması
- Anahtarların büyük olması
- İmzaların büyük olması

Bu sebeplerden ötürü de düşük kaynaklı cihazlarda kullanmak mümkün olmamaktadır Onun İçin UOV şemasının üzerinde iyileştirmeler yapılarak çeşitli varyasyonlar geliştirilmiştir.

Tablo 2.2 UOV Güvenlik Parametreleri Tablosu (Yang, B. Y.)

UOV için Güvenlik Parametreleri (F,o,v)					
Güvenlik Seviyesi(bit)	Şema	Açık Anahtar Boyutu(kB)	Gizli Anahtar Boyutu (kB)	Hash Boyutu (bit)	İmza Boyutu (bit)
80	UOV($F_{16,}$ 40,80)	144,2	135,2	160	480
	UOV($F_{256,}$ 27,54)	89,8	86,2	216	648
100	UOV($F_{16,}$ 50,100)	280,2	260,1	200	600
	UOV($F_{256,}$ 34,68)	177,8	168,3	272	816
128	UOV($F_{16,}$ 64,128)	585,1	538,1	256	768
	UOV($F_{256,}$ 45,90)	409,4	381,8	360	1080
192	UOV($F_{16,}$ 96,192)	1964,3	1786,7	384	1152
	UOV($F_{256,}$ 69,138)	1464,6	1344	552	1656
256	UOV($F_{16,}$ 128,256)	4644,1	4200,3	512	1536
	UOV($F_{256,}$ 93,186)	3572,9	3252,2	744	2232

2.3.3 Rainbow

Rainbow Unbalance Oil and Vinegar scheme (UOV) ın çok katmanlı bir varyasyonudur. UOV farklı olarak bir liner dönüşüm daha eklenerek anahtarı küçültmek hedeflenmektedir. İsminden de anlaşılacağı üzere gökkuşağı gibi farklı katmanları var fakat bunlar birbirine karışmayacak şekilde tasarlanmıştır. Çok katmanlı bir yapıya sahip ve bu katmanlar birbirine bağlı ve hiyerarşik olarak devam etmektedir. Bir katmanda yapılan sonuç diğer tarafta da kullanıldığı için çözüm hazır bulunuyor bu da performans sağlıyor bunun yanı sıra en önemli nokta birbirine bağlı olduğu için daha küçük anahtarlar ile güvenliği fazlasıyla sağlamaktadır. Rainbow denklemleri şu şekildedir.

$$f_k(u_1, \dots, u_n) := \sum_{i \in V_1, j \in V_1} \gamma_{k,i,j} u_i u_j + \sum_{i \in V_1, j \in O_1} \gamma_{k,i,j} u_i u_j$$

$$\text{for } k = 1, \dots, o_1$$

$$f_k(u_1, \dots, u_n) := \sum_{i \in V_1 \cup O_1, j \in V_1 \cup O_1} \gamma_{k,i,j} u_i u_j + \sum_{i \in V_1 \cup O_1, j \in O_2} \gamma_{k,i,j} u_i u_j$$

$$\text{for } k = o_1 + 1, \dots, o_1 + o_2$$

2007 yılında beri herhangi bir zayıflığı bulunamamıştır (Yang, B. Y.) Avantajları

- Oldukça güvenli olması
- UOV ye göre daha küçük parametreler kullanıyor olması
- Daha küçük imza oluşuyor olması
- Daha küçük açık anahtar ve daha küçük gizli anahtar oluşuyor olması
- Verimli olması
- Düşük özellikli cihazlar içinde kullanılabilir olması

Tablo 2.3 UOV Güvenlik Parametreleri Tablosu (Yang, B. Y.)

Rainbow için Güvenlik Parametreleri (F,o,v)					
Güvenlik Seviyesi(bit)	Şema	Açık Anahtar Boyutu(kB)	Gizli Anahtar Boyutu (kB)	Hash Boyutu (bit)	İmza Boyutu (bit)
80	UOV($F_{16,}$ 20,20,20)	33,4	22,3	160	228
	UOV($F_{256,}$ 19,12,13)	25,3	19,3	200	352
100	UOV($F_{16,}$ 25,25,25)	65,9	43,2	200	288
	UOV($F_{256,}$ 27,16,16)	57,2	44,3	256	472
128	UOV($F_{16,}$ 32,32,32)	136,6	87,6	256	368
	UOV($F_{31,}$ 28,28,28)	123,2	74,5	280	420
	UOV($F_{256,}$ 36,281,22)	136	102,5	344	632
192	UOV($F_{16,}$ 48,48,48)	475,9	301,8	384	564
	UOV($F_{31,}$ 44,40,40)	360,1	245,2	420	630
256	UOV($F_{16,}$ 64,64,64)	1194,4	763,9	512	776

Tablo 2.3 de görüldüğü üzere UOV e göre fazlasıyla verimli bir sistem.

2.3.4 Lifted Unbalanced Oil and Vinegar signature scheme (LUOV)

UOV 1997 yılında J. Patarin tarafından önerildi ve 20 yıldır kripto analizlere karşı başarılı sonuçlar verdi. Hızlı ve küçük imza üretmesi avantajları fakat en büyük dezavantajı açık anahtarı çok büyük olması ve buna çözüm olarak da açık anahtarı çok iyi bir şekilde azaltan Lifted Unbalanced Oil and Vinegar signature scheme (LUOV) iyileştirilmesi geliştirildi. LUOV şeması UOV imzalama şemasından farkı ise bir takım farklı yolları kullanıyor olmasıdır.

İlk değişiklik anahtar üretiminde açık anahtarın daha büyük bir kısmının seçimin mümkün kılmak için anahtar üretim algoritması değiştiriliyor. Anahtar üretim algoritması ise şu şekilde çalışmaktadır (Smith-Tone, D.).

Anahtar üretimi içimi Gizli anahtar olarak gizli bir tohum kullanılıyor. LUOV imzalama şemalarında kriptolojik olarak güvenli rastgele veri akışını sağlamak için genişletilebilir SHAKE fonksiyonları kullanılıyor (Smith-Tone, D.). İlk başta bir tohum anahtar Keccak1600 ile özetleme (Hash) işlemi gerçekleştiriliyor ve ardından bu özet halinde bulunan anahtar matematiksel işlemlerden geçirilerek bytelar halinde çıktı verilerek anahtar üretim algoritmasında kullanılacak veriler elde ediliyor. Bu yaklaşım açık tohumun (public_seed), T matrisinin , mesajın özetlenmiş parçası olan h nin , v vinegar değişkenin ve P açık haritasının kısmıları C,L,Q_1 parçalarını üretmek için kullanılmaktadır.

Keccaksponge nesnesini ve keccak işlemini açıklayacak olursak, burada bir özetleme (hashing) işlemi yapılıyor. Keccak aynı SHA3 gibi çok bilinen bir özetleme algoritmasıdır. SHA2 ile kıyaslandığı zaman daha hızlı ve daha güvenli SHA3 ile de hız konusunda kıyaslamalar yapılmaktadır. Sponge fonksiyonu ise girdiyi, girilen miktar büyüklüğünde ve ekleme (padding) yapılmış bloklar oluşturup ardından XOR işlemleri yapılıyor. Çalışması da şu şekilde Sponge başlangıç durumunda olmalı tüm durumlarda çalışabilmesi için. Sponge isteğe bağlı şekilde uzun diziler halinde rastgele veriler (byte) sağlayabiliyor. Squaze operasyonu ile birlikte bir tane sponge nesnesi ve bir b sayısı alıyor girdi gibi ve çıktı olarak da sıralı bir b bytelık dizi çıkartılıyor ve spongenin durumu güncelleniyor.

Aslında tüm bunları özetleyecek ve örneklendirecek olursak squzeing işlemi sıkma sıkıştırma anlamına geliyor sponge ise sünger anlamında çevrilebilmektedir. Gizli tohum

dediğimiz ise gizli anahtar. Gizli tohumu sünger (sponge) yapısına gömülüyor yani, sünger tohumu emiyor ve ardından çeşitli matematiksel yorumlamalar ve işlemler ile anahtar üretimi için kullanılan değişkenlerin üretimi sağlanıyor ve anahtar üretimi gerçekleşiyor. Ama Bu şekilde ifade edildiği zaman çok bir şey ifade etmiyor ve akıllarda çok bir şeyler canlanmıyor. Örneklendirilecek olursa buradaki sponge diye ifade edilen yapıyı bir bulaşık süngeri olarak matematiksel işlemleri ise su, gizli tohum olarak ifade edilen yapıyı ise deterjan olarak kabul edilirse süngerde su ile deterjan birleştiği zaman çok fazla köpük oluşturur ve sponge diye ifade edilen sıkma (özetleme) işlemi ile de süngeri sıkıldığı zaman köpük oluşur ve kullanılır. Buradaki köpüğü su ve deterjan olarak ayırabilmek neredeyse imkansıza yakın ya da çok zor bu yüzden deterjana ulaşamıyor ve ondan üretilen köpük kullanılabiliyor. Burada biz gizli anahtar güvene alınarak ulaşılamıyor ve bu gizli anahtardan açık tohum public_seed, T matrisinin , mesajın özetlenmiş parçası olan h nin , v vinegar değişkenin ve P açık haritasının kısmıları C,L,Q_1 parçaları üretiliyor ve bu parçalar ile de anahtar üretimi sağlanmaktadır.

- İkinci değişiklik açık anahtarda

UOV şeması F_2 üzerinde çalışıyorken geniş bir alan üzerinde çalışıyor şöyle ki F_2 de katsayılarım ve işlem sonuçlarım 1 ve 0 dan ibaret olduğu için çok fazla sayıda 1 ve 0 lardan oluşan boyut olarak çok büyük açık anahtar elde edilecek bunun yerine LUOV 2 lik taban yerine F_2r üzerinde çalışıyor bu sayede 2^r lik tabanda daha fazla sayı ile çalışabildiği için açık anahtar kalıntıları 1 ve 0 yani 2'lik tabana göre daha az oluyor. Bu sistemin avantajı da bu oluyor ve açık anahtar hareketli bir cisim genişleme alanı kullanıyor LUOV un ismi de buradan geliyor Lifted UOV. Çözüm yapılırken P(X) =y için aynı şekilde y nin $F^{M}{}_{2}{}^{r}$de $F^{M}{}_{2}$ göre kıyaslaması daha zor bir hale geliyor ve bu güvenlik seviyesini arttırmaktadır.

- Üçüncü değişiklik ise T haritası matris şeklinde gösteriliyor. V x M lik kullanılıyor

bu hızlı ve imzalama anahtar üretimini güvenlik olarak etkilemiyor, çünkü açık anahtar rastgele olarak T haritası ile birlikte gizli anahtar eşitliğinden oluşuyor. Bu değişikliğe yapılan ataklar orijinal UOV şemasını da etkileyecektir yani bu değişiklik UOV a göre herhangi bir zafiyet oluşturmamıştır.

Ataklara karşı normal UOV a göre çok daha dayanıklı ve %20 ile %50 arasında daha kısa sürede karşı koyabilmektedir (Smith-Tone, D.).

Avantajları (Smith-Tone, D.).

- Küçük imzaya sahip olması
- Sade bir aritmetiği olması
- Mesaj geri gönderimine sahip olması
- Belirli bir sınırı yok sadece belirli bir aralık için değil sınırsız mesajı imzalayabiliyor, bu yüzden daha kolay bir şekilde uygulanabiliyor olması
- Esnek bir yapıya sahip olması
- Çeşitliliğe sahip yani farklı farklı zor durumlarda da çalışıyor olması
- Herhangi bir harici kaynağa ihtiyaç duymuyor ve bu yüzden düşük kaynaklı cihazlarda da kullanılabiliyor olmasıdır.

2.3.5 DualModeMS

DualModeMS çok değişkenli imzalama şemalarında diğerlerine göre ayrıcalıklı öneme sahip olduğu düşünülmektedir (Perret, L. 2017). Geleneksel çok değişkenli imzalama şemaları ile keskin ayrımları vardır. Küçük açık anahtarlar ile geniş imzalar üretiliyor. Matsumoto ve Imai yapıları olarak da isimlendiriliyor. NIST tarafından önerilen diğer çok değişkenli imzalama şemalarına göre kullanışlı bir aday olacağı düşünülmektedir (Perret, L. 2017). İnnerlayer diye isimlendirilen iç katman ve Outer layer diye isimlendirilen dış katman olmak üzere iki katmanlı bir yapıdan oluşuyor. DualModeMS nin iki katmanlı bir yapıya sahip olduğu bu katmanların iç ve dış katmanlar olduğu ve iç katmanın temelinde HFEv vardır (Perret, L. 2017).

HFE Hidden Field Equations Gizli Cisim Eşitliği olarak tercüme edebiliriz. “+”, “-“, “f”, “v” olmak üzere Farklı varyasyonları mevcuttur. V harfi, sirke (vinegar) değişkenini ifade etmektedir. HFEv kuantum sonrası için en çok umut vaat eden imzalama şeması olarak belirtilmektedir (Petzoldt, A. 2017). En dikkat çeken yanı ise kısa imzalama şemalarının olmasıdır. Çok değişkenli polinonlar sonlu cisim üzerinde çalışmaktadır. F_q çoğunlukla q=2 olarak ve F_2 de çalışmaktadır.

İç katmanın güvenlik incelemesinde Great Multivariate Short Signature ile benzerlik gösterdiği görülmektedir (Perret, L. 2017).

Great Multivariate Short Signature kalıp olarak şemaları HFE sistemidir. (Casanova, A ve ark. 2017) İmza doğrulama süreci çok hızlı açık anahtarı da fazla büyük değil (Casanova, A ve ark. 2017). Bu yüzden artıları olan bir sistem. GeMSS den

farklılıkları mevcut çünkü imza doğrulama aşamalarında Merkle Tree fonksiyonu kullanılıyor. Bu yüzden de farkı yinelemeli olması. Açık ve gizli anahtar üretiminde ise iç katmanın güvenlik seviyesi girdi olarak kullanılıyor ve imza üretmek için rootları hesaplayabilmek içinde berlekamp algoritması kullanılıyor. Berlekamp Algoritması verilen çıktı için en kısa Linear Feedback Shift Register (LFSR) dizisini bulmaya yaramaktadır.

Bilinen ataklara karşı ise; Direkt ataklara karşı etkili kapsamlı aramalarda ve kuantum algoritmaları, kuantum eşitlilikleri kullanılarak daha iyi sonuçlar elde edilebiliyor. Direkt ataklara göre yaklaşım algoritmaları kullanılarak daha iyi sonuçlar elde edilebiliyor. Gröbner temellerine (Üreteç ve tanımlı bağıntılar tarafından sunulmuş herhangi bir Liener cebirin liner tabanı nasıl bulunur sorusu ile ortaya çıkmış bir teoridir.) Bu zamana kadar ki doğrusal olmayan ataklarda en iyi yöntemler gröbner temellerinden faydalanmaktadır. Gröbner temelleri Literatürde çok önemli bir algoritma olmasına rağmen sonlu cisim aritmetiği polinom çözümlemelerinde referans yazılım olarak MAGMA ve FGb devam ediyor (Perret, L. 2017). Kipnis-Sahmir atakları (kuantum sonrası en önemli ataklardan bir tanesi) ve diferansiyel ataklar karşı da sonuçlar yer alıyor.

3. MATERYAL VE METOTLAR

3.1 First Come First Serve (FCFS)

First Come First Serve algoritması, çalışma mantığı açısından kuyruk veri yapısındaki First In First Out yapısına benzemektedir. En Bilgisayar mimarisinde işletim sistemleri başta olmak üzere birçok alanda kullanılan bir algoritmadır. Uygulaması ve anlaşılması basit olmasına karşın Kesintisiz ve ilkel bir algoritma olması ciddi bir dezavantaj oluşturmaktadır. Kesintisiz bir algoritma olmasından dolayı bir işlem bitmeden diğeri başlayamayacağından çok kısa bir iş dahi olsa kaynaklara erişemediğinden CPU açlığı oluşmasına sebep olur. İşlemlerin ortalama bekleme süreleri uzundur. Kaynaklar etkin bir şekilde kullanılamadığı için performans olarak verimsizdir.

Tablo 3.1 First Come First Serve Algoritmasının Sözde Kodu

Adım 1. İşlemler çalışma zamanları birlikte girilir. (bt=burst time)

Adım 2. Tüm işlemlerin bekleme süreleri bulunur. (wt=wait time)

Adım 3. İlk işlem için beklemeye gerek olmadığından wt=0 olur

Adım 4. Tüm işlemlerin bekleme süreleri bulunur

wt[i]=bt[i-1]+wt[i-1]

Adım 5. Tüm işlemler için dönüş süresi bulunur

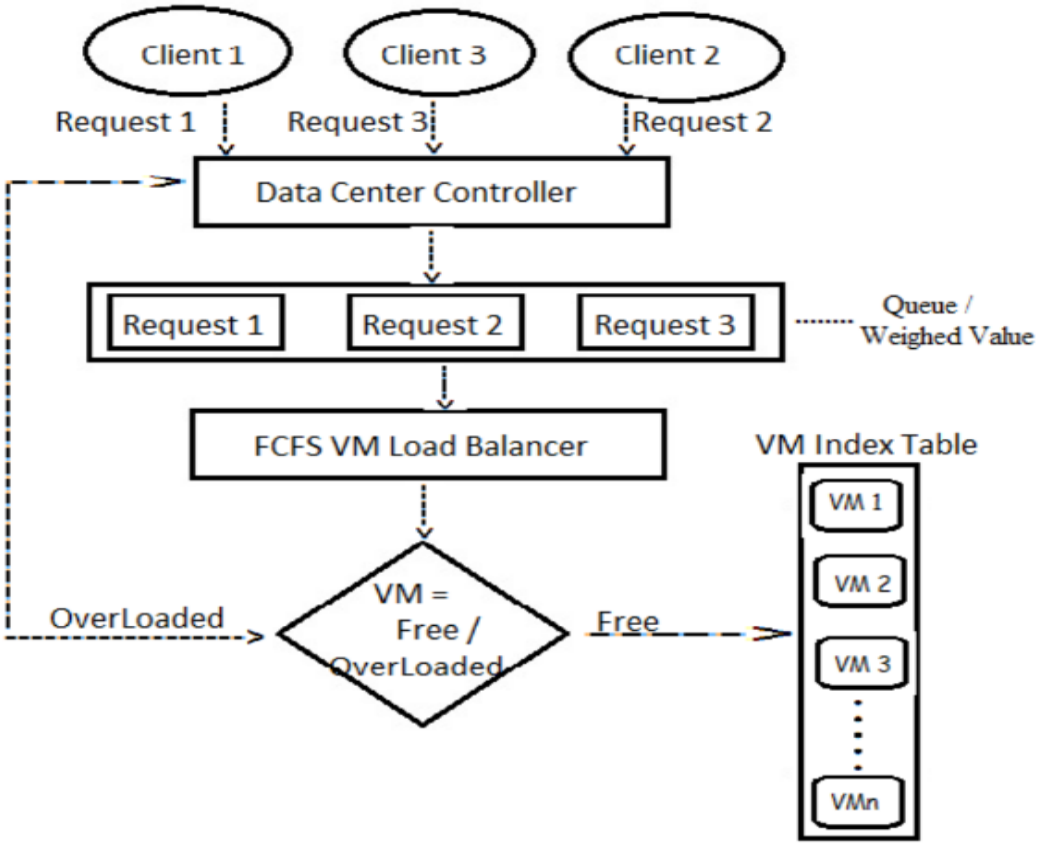
Dönüş süresi =bekleme süresi(wt) + çalışma süresi(bt)

Adım 6. Ortalama bekleme süresi hesaplanır.

Toplam bekleme süresi / başlamamış işlemler

Adım 7. Ortalama geri dönüş süresi hesaplanır.

Toplam geri dönüş süresi / başlamamış işlemler



Şekil 3.1 First Come First Serve Algoritması Diyagramı (Pattanaik, P. A.,ve ark (2015, February))

3.2 Shortest Job First (SJF)

En kısa işin ilk tamamlandığı bir çizelgeleme algoritmasıdır. Hâlihazırda mevcut olan işler, çalışma sürelerine göre sıralanır. Bu sıralama için özellikli bir sıralama algoritması yerine genel de brute force olarak ilkel olarak sıralaması yapılarak en kısa süren iş öncelikli olarak sıralanarak küçükten büyüğe doğru işler sırayla alınır. Çalışma mantığı olarak kesintisiz bir algoritma olan bu yaklaşımda bir iş başladıktan sonra başka bir işin araya giremez. Eldeki en kısa işi yaparak ortalama bekleme süresini azaltarak performansı artırmaya çalışılır. Tüm işlerin çalışma sürelerinin birbirlerine yakın süreler olduğu ve bilindiği durumlarda verimli olur. Algoritmanın verimli çalışabilmesi tüm işlerin çalışma zamanları tam olarak bilinmesi gerekir fakat her durumda çalışma sürelerini bilmek mümkün olmaz. Teoride ilkel ve kesintisiz algoritmalar arasında optimal performans sergileyeceği gözükse de uygulamada çalışma sürelerinin her zaman tam olarak bilinememesinden dolayı performans beklentisini karşılayamadığı durumlar vardır. Kritik olmayan işler için uygulaması kolay olduğundan tercih edilebilir. Çalışma süresi uzun

olan işlerin geride kalacağı için kesintisiz algoritmaların ortak sorunu olan CPU açlığı yaşanmasında sebep olur.

Tablo 3.2 Shortest Job First Algoritmasının Sözde Kodu

Adım 1. İşlemler çalışma zamanlarına göre sıralanır.

Adım 2. En kısa çalışma süresine sahip olan iş başlatılır.

Adım 3. Tamamlanma süresi hesaplanır

Tamamlanma süresi=İşlemin Yürütülmesinin sonuna ulaştığı zamandır.

Adım 4. Geri dönüş süresi hesaplanır

Geri Dönüş Süresi = Herhangi bir Sürecin Tamamlanma Süresi- Herhangi bir Sürecin Varış Süresi.

Adım 5. Bekleme süresi bulunur

Bekleme Süresi = İşlemin Geri Dönüş Süresi - İşlemin Çalışma Süresi

Adım 6. Ortalama bekleme süresi hesaplanır.

Toplam bekleme süresi / başlamamış işlemler

Adım 7. Ortalama geri dönüş süresi hesaplanır.

Toplam geri dönüş süresi / başlamamış işlemler

3.3 Round Robin Algoritması

Round Robin, zaman paylaşımlı sistemlerde kullanılmak üzere tasarlanmış bir algoritmadır. Algoritmaya göre bir işlem belirlenen süre içerisinde bitmese dahi beklemeye alınarak yeni işleme geçilir. Böylece uzun bir işlem diğer işlemlerin yapılmasını engellemez. CPU açlığı diye nitelendirilen, işlemin ihtiyaç duyduğu kaynağa erişememe problemi ortadan kalkar.

Veri aktarımı dairesel bir sırada yapılır. Örneğin 1. işlemin tamamlanma süresi 8 ms, 2. işlemin tamamlanma süresi 14 ms ve 3. işlemin tamamlanma süresi 5 ms olarak varsayılırsa, kuantum süresi olarak da 3 ms olarak belirlemiş olursa, ilk süreç 1 başlatılır. 3 milisaniyelik süre bittiğinde 1. işlem geçici olarak durdurulur. Daha sonra 2. işleme

geçilir ve tekrar 3 ms işlenir. 3 saniyelik zaman dilimi bittiğinde 2. işlem de geçici olarak durdurulur ve 3. işlem başlatılır.3. İşlem 3 ms boyunca işlenir. 3 milisaniyelik süre bittiğinde 3.işlem de geçici olarak durdurulur ve başa dönülerek 1.işlem başlatılır. 1. süreç kaldığı yerden devam eder. Süreç bu şekilde devam eder. Böylece 1. 2. ve 3. işlemler tamamlanmış olur ve işlemler birbirinin bitmesini beklemez. Kesintili bir algoritmadır. Buradaki önemli nokta ise kuantum sayısını iyi belirlenmiş olması bu sayede sistemin daha verimli olması sağlanabilir.

Tablo 3.3 Round Robin Algoritmasının Sözde Kodu

Adım 1. Kuantum süresi belirlenir.

Adım 2. İşlemler geldiği sırasıyla kuantum süresi kadar çalıştırılır

Adım 3. Kuantum süresi bitiminde işlem tamamlanmamış olsa da bir sonraki işleme geçilir. Tüm işlemler için bu süreç tekrarlanır.

Adım 4. Tamamlanmamış işlemler kuantum süresi kadar çalıştırılır ve bu tüm işlemler bitene kadar devam ettirilir.

Adım 5. Ortalama bekleme süresi hesaplanır.

Toplam bekleme süresi / başlamamış işlemler

Adım 6. Ortalama geri dönüş süresi hesaplanır.

Toplam geri dönüş süresi / başlamamış işlemler

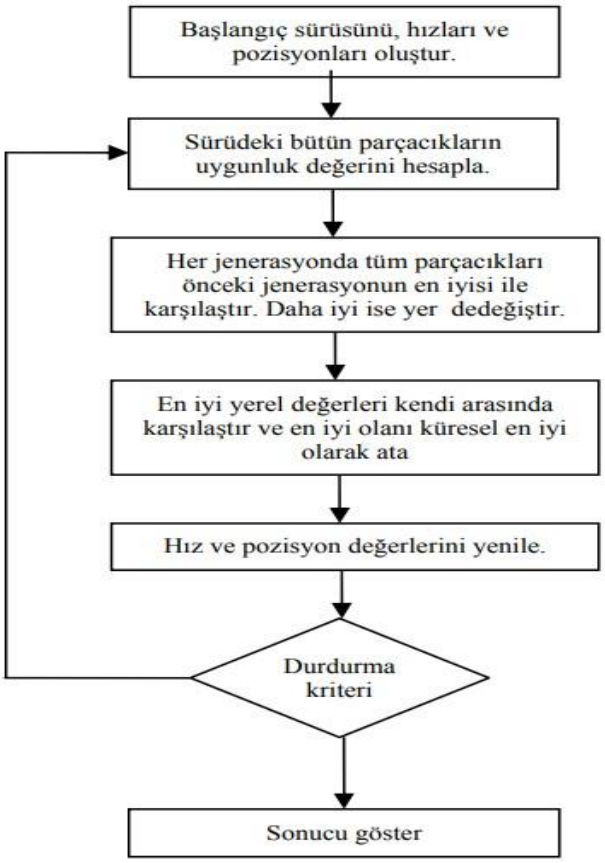
3.4 Particle Swarm Optimization (PSO)-Parçacık Sürü Optimizasyonu

Günümüzde kullanılan birçok araç, düşünce ve teori gerek tabiatı gerek hayvanları gerek ise insanları gözlemleyerek oluşturulmuş veya taklit edilmiştir. Buna bilimsel olarak biyomimikri denir. Biyolojik sistemlerden esinlenerek bilişim problemlerin çözümünde kullanılan birçok yöntem ortaya konulmuştur. Yapay sinir ağları bu yöntemlerdendir. İnsan beyninin basitleştirilmiş bir modeli düşünülerek tasarlanmıştır. Yine Genetik algoritma da bu yöntemlerdendir. İnsan evriminden türetilmiştir. Biyolojik sistemlerin bir başka türü olan sosyal sistemler vardır. Sosyal sistemler bireyin çevresiyle ve diğer bireylerle olan etkileşimlerini kapsamlı olarak davranışlarını inceler. Bu davranışların terim olarak sürü zihniyeti denilir.

PSO kavramı, özünde sosyal yaşamın sadeleştirilmiş bir simülasyonudur. Daha sonra bu simülasyon bir optimizasyon yöntemine dönüştürülerek kullanılmaya başlanmıştır. Parçacık Sürü Optimizasyonu (PSO); 1995'te J. Kennedy ve R.C.Eberhart tarafından, kuş sürülerinin davranışları incelenerek popülasyon tabanlı bir stokastik optimizasyon oluşturulmuştur (Aiello ve ark. 2009) . Lineer olmayan problemleri çözümü için tasarlanmıştır. Çok değişkenli optimizasyon problemlerinin çözümü için kullanılır. PSO, genetik algoritmalar gibi evrimsel hesaplama teknikleriyle birçok benzerlik gösterir. Sistem rastgele çözümler içeren bir popülasyonla başlar ve nesilleri güncelleyerek en uygun çözümü arar. PSO'da parçacık olarak adlandırılan olası çözümler, o anda optimum parçacığı takip eder ve problem uzayında dolaşır. PSO Algoritmasında ayarlanması gereken parametre sayısı az olduğundan, uygulanması oldukça basittir. PSO; optimizasyon fonksiyonlarında, bulanık sistem kontrollerinde ve yapay sinir ağları eğitimi gibi birçok alanda başarıyla uygulanabilmektedir (Zhou ve ark. 2006).

Kuşlar yiyecek ararken, yiyeceğe en yakın kuşu takip ederler. Parçacık olarak adlandırılan her tekil çözüm, arama uzayında bir kuş gibidir. Parçacık hareket ettiğinde koordinatlarını bir fonksiyona göndererek parçacığın uygunluk değerinin ölçülmesini sağlar. Bir parçacık koordinatlarını, hızını, şimdiye kadar elde ettiği en iyi uygunluk değerini ve bu değeri aldığı koordinatları hatırlamalıdır. Çözüm uzayındaki her bir boyuttaki hızının ve yönünün her seferinde nasıl değişeceği, komşularının en iyi koordinatları ile kendi kişisel en iyi koordinatlarının bir kombinasyonu olarak belirlenir.

Algoritma temel olarak aşağıdaki basamaklardan oluşur;



Şekil 3.2 First Come First Serve Algoritması Diyagramı (Koç, T. ve ark (2021))

Tablo 3.4 PSO Algoritmasının Sözde Kodu

Adım 1. Rasgele olarak başlangıç sürüsü (başlangıç konumları ve hızları ile) oluşturulur.

Adım 2. Sürüde yer alan parçacıkların uygunluk değerleri bulunur

Adım 3. Her bir parçacık için yerel en iyi (pbest) bulunur.

Adım 4. Yerel en iyiler içerisinden global en iyi (gbest) seçilir.

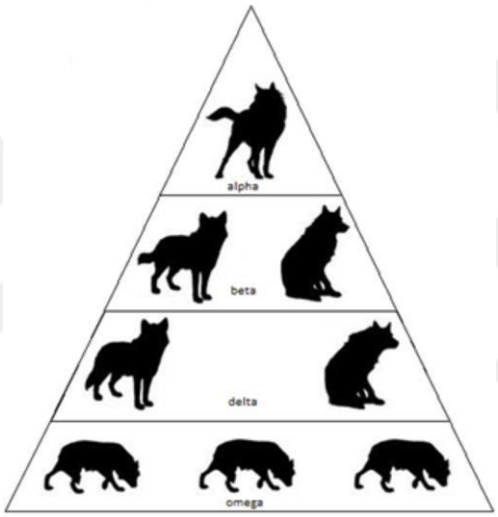
Adım 5. Konum ve hızlar aşağıdaki gibi yenilenir

Vid=W* Vid + c1*rand1 *(Pid - Xid)+c2*rand2 * (Pid –Xid)

Xid = Xid + Vid

3.5 Grey Wolf Optimization (PSO)-Gri Kurt Optimizasyonu

Gri Kurt Optimizasyon (GWO) algoritması, kurtların davranışları incelenerek avlanma ve liderlik özellikleri baz alınarak 2014 yılında Mirjalili tarafından geliştirilmiş olan popülasyon tabanlı sezgi (metasezgisel) üstü bir optimizasyon algoritmasıdır. Bir sürü algoritmasıdır. Sürüde kendi aralarında yeteneklerine ve özelliklerine göre görev dağılımı yapılmaktadır. Görev dağılımında kurtların, bazıları arama kurtları bazıları avlarının konumu bildirerek bir haberci gibi diğer kurtlara bildirir. Diğer kurtlar ava yaklaşarak avı kuşatırlar. Kurtların sürü içerisinde dört gurup halinde hiyerarşik bir yapıya sahiptirler. (bkz. Şekil 3.3)



Şekil 3.3 Gri Kurtların Hiyerarşisi (İnaç, T ve ark (2022))

Alfa Kurtlar(α) : Sürünün Lideri olan kurtlar alfa kurdu olarak adlandırılırlar. Alfa kurt grubun yönetimi için en yetenekli ve grup tarafından kabul edilen en iyi kurttur ve grubun karar vericisi konumundadır. Grup için yaşamsal döngülerindeki önemli olan olaylar olan avlanma, uyuma yeri, uyanma zamanı gibi konularda karar vermekle sorumludur. Alfa sürüyü yöneten kurttur.

Beta Kurtlar(β) : Hiyerarşik sırada kurt grubunun içindeki ikinci kurt beta kurttur. Beta kurt alfa kurdun yardımcısı durumundadır. Bu yüzden grubun kararlarında alfa kurtlara yardımcı olarak karar süreçlerinin içerisinde yer almaktadır.

Delta Kurtlar(δ) : Delta kurtlar hiyerarşik olarak alfa ve beta kurtların altında yer aldığında hiyerarşide üçüncü kurttur. Delta Kurtlar avcı ve koruyucu kategorisinde yer

almaktadır. Bölge sınırlarını korurlar oluşacak bir tehlikede sürüyü uyararak sorumluluklarındandır.

Omega Kurtlar(ω): Hiyerarşinin en altında yer alan gri kurtlar omega kurtlardır. Omega kurtları diğer kurtların himayesinde yer almaktadırlar.

Algoritma olarak ifade ederken;

Problemin en iyi çözümüne sırasıyla alfa (α), beta (β) ve delta (δ) olarak adlandırılır. Geriye kalan tüm çözümler omega (ω) olarak sınıflandırılır.

Avlanma sırasında avlanmayı etkili kılan ve Algoritmaya dönüşmesine sebep olan adımlar şu şekildedir.

3.5.1 Avı Kuşatma Adımı:

Algoritmanın çalışma sırasında kurtlar konumlarını α, β yada δ etrafında yenilerler. Gri kurtlar algoritmasında avı kuşatmak için denklem 3.1 ve 3.2 kullanılmaktadır.

$$D = |C \cdot XP\ (t) - X(t)| \quad (3.1)$$

$$X(t + 1) = X_P\ (t) - A \cdot \mathrm{D} \quad (3.2)$$

Denklemlerde yer alan $X_{(t)}$ kurdun konumu, t döngü sayısını X_P av pozisyonunu karşılık gelmektedir. A ve C avın konum vektörünü, X de bir gri kurdun konumuna karşılık gelmektedir. A ve C değerlerini denklem 3.3 ve 3.4'e göre hesaplanır.

$$A=|2.a.r1-a| \quad (3.3)$$

$$C=|2.a.r2\ | \quad (3.4)$$

a'nın bileşenlerini iterasyonlar esnasında doğrusal olarak 0 ile 2 arasında yer almaktadır. [0,1] ve bu rastgele vektörler kurdun arama uzayında herhangi bir noktaya ulaşmasını sağlar. Böylece gri kurt, 3.5 ve 3.6 denklemlerine göre herhangi bir rastgele konumda avın etrafındaki uzayda konumunu ayarlayabilir. Benzer şekilde 2 ve 3 boyutlu uzay n boyutlu bir arama uzayına genişletilebilir ve gri kurdun şimdiye kadar bulunan en iyi çözüm etrafında hareket etmesine izin verir.

$$D=|C.Xp(t)-X(t)| \quad (3.5)$$

$$X(t+1)=|Xp(t)-A.D| \quad (3.6)$$

3.5.2 Avlanma:

Alfa, beta ve delta türünde ki gri kurtlar avın mevcut konumu hakkında beklenenden daha fazla bilgiye sahiptir. Bundan dolayı varılan en iyi ilk üç çözüm kaydedilir ve diğer kurtların konumlarına göre güncellenir ve bu işlemlerde aşağıda yer alan denklemler kullanılır.

$$D\alpha = |C1.X\alpha - X| \quad (3.7)$$

$$D\beta = |C2.X\beta - X| \quad (3.8)$$

$$D\delta = |C3.X\delta - X| \quad (3.9)$$

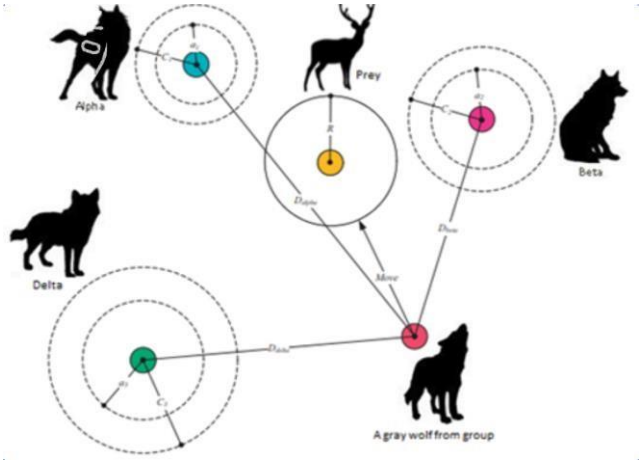
$$X1 = |X\alpha - A1.D\alpha| \quad (3.10)$$

$$X2 = |X\beta - A2.D\beta| \quad (3.11)$$

$$X3 = |X\delta - A3.D\delta| \quad (3.12)$$

$$X(t+1) = ((X1+X2+X3))/3 \quad (3.13)$$

Denklemlerde yer alan Dα, Dβ, Dδ av ve kurtlar (alfa, beta, delta) arasındaki mesafeleri; Xα, Xβ, Xδ alfa, beta ve delta kurtlarının konumlarını; X gri kurdun i. yinelemesindeki konumunu; C1, C2, C3, A1, A2, A3 ise alfa, beta ve delta kurtları için katsayı vektörlerini; X1, X2, X3 alfa, beta ve delta kurtları için deneme vektörlerine temsil etmektedir. Şekil 2.4'te gri kurt grubunun avlanma mekanizması gösterilmektedir. Her yinelemede, en iyi üç kurdun konumları güncellenmektedir. X(t+1) ise avın yeni konumuna karşılık gelmektedir.



Şekil 3.4 Avlanma Esnasında Gri Kurtların Avlanma Mekanizması (İnaç, T ve ark (2022))

3.5.3 Ava Saldırma:

Alfa, beta ve delta kurtlar arasındaki mesafe yaklaşık olarak geçerli çözümle hesaplanır. Mesafeler hesaplandıktan sonra, geçerli çözümün güncel konumu hesaplanır. Bu aşamada, a değeri azaltılarak işleme devam edilir. Bu yüzden A'nın aralığı da değiştirilerek azaltılır. A [-1,1] aralığında rastgele değerler aldığında, aramaya aracılık eden kurdun bir sonraki konumu, geçerli konum ile avın konumu arasında rastgele bir yerde olacaktır.

Tablo 3.5 Gri Kurt Algoritmasının Sözde Kodu

Adım 1. Gri kurtların pozisyonlarını başlatılır

Adım 2. Gri kurtların maliyet değerleri hesaplanır

Adım 3. En iyi gri kurt alfa kurt olarak belirlenir- Xα

İkinci en iyi gri kurt beta kurt olarak belirlenir- Xβ

Üçüncü en iyi gri kurt delta kurt olarak belirlenir-Xδ

Adım 4. İterasyon Sayısı kadar

Her bir kurt için kurtların konumlarını denklem 3.13 e göre güncellenir.

Adım 5. İterasyon Sayısı kadar Denklem 3.3 ve 3.4 deki değişkenler güncellenir

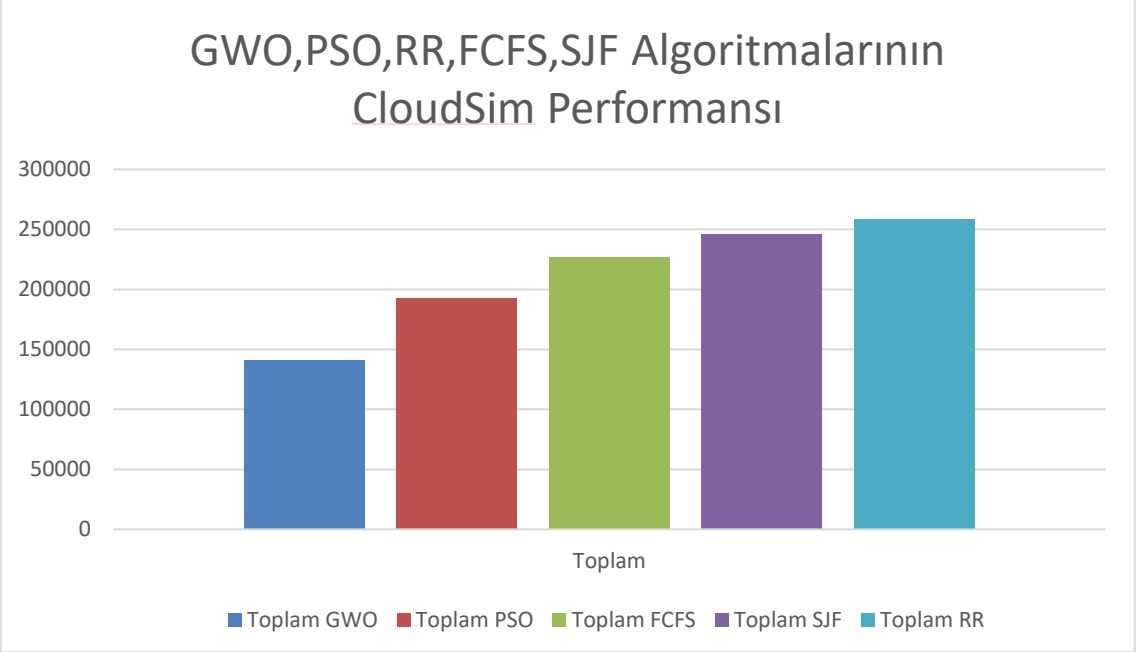
Adım 6. İterasyon Sayısı kadar Her bir kurdun konumu hesaplanır

Adım 7. Kurtlar için avı pozisyonu olan Xα , Xβ, Xδ parametleri hesaplanır

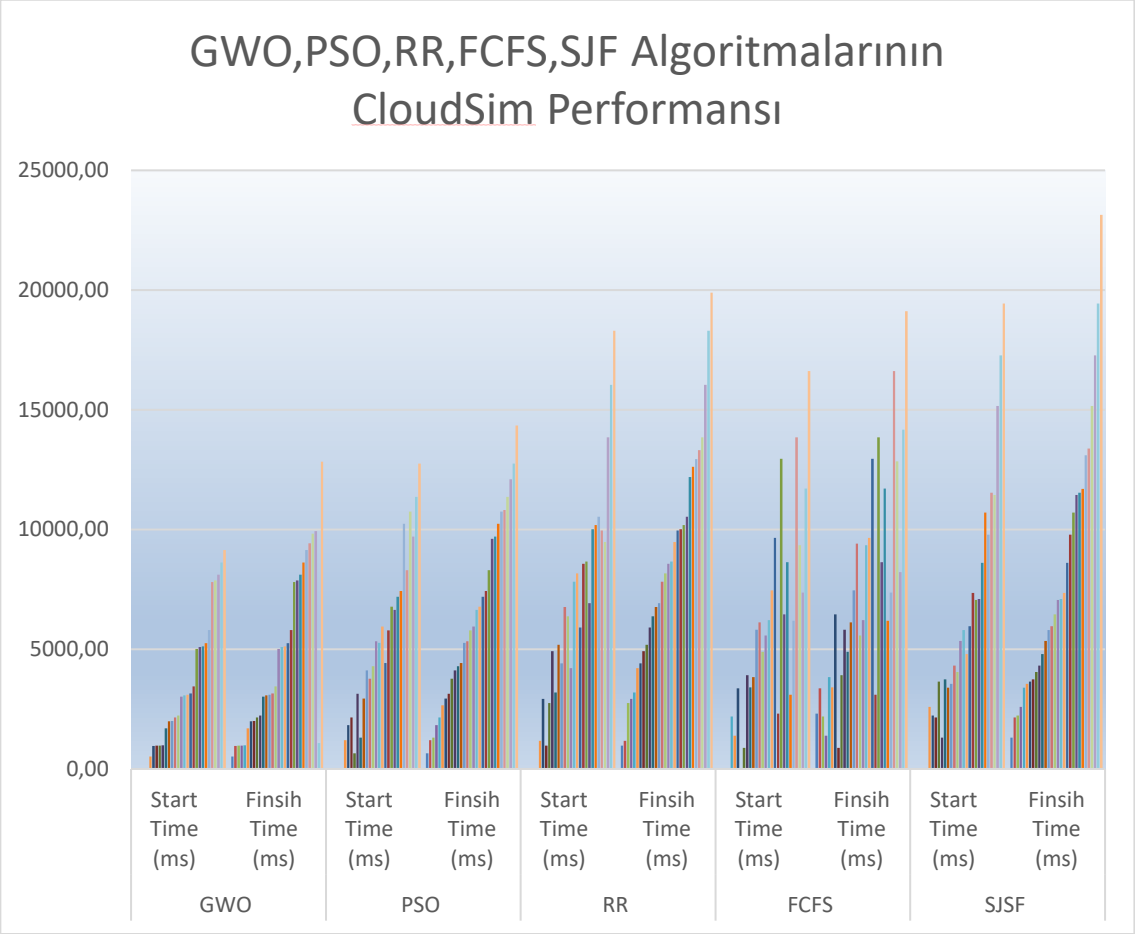
Adım 8. Alfa Kurt (α) döndürülür

4. Bulgular

Aşağıdaki grafikde (Şekil 3.7) görüldüğü üzere beş algoritmanın CloudSim üzerindeki 30 adet girdi ile süreç performanslarını görüyoruz. Kategorize edildiği zaman Particle Swarm Optimization (PSO)-Parçacık Sürü Optimizasyonu ve Grey Wolf Optimization (GWO)- Gri Kurt Optimizasyonu sezgi üstü (meta-heuristic) algoritmalardan olup en iyi performansları sağlamıştır. Grey Wolf Optimization (GWO)- Gri Kurt Optimizasyonu daha güncel bir algoritma olması sebebi ile PSO algoritmasına göre daha iyi bir performans sergilemiş olup üstünlük göstermiştir. Round Robin, First Come First Serve, and Shortest Job First Algoritmaları ise geleneksel algoritmalardan olup kendi aralarında ise First Come First Serve, and Shortest Job First algoritması kesintisiz çalışmakta olup, Round Robin algoritması kesintili olarak çalışmaktadır ve en kötü performansa Round Robin algoritması ulaşmıştır. Şekil 3.9'da tabloada Algoritmaların zaman verileri Şekil 3.8'de ki grafikte ise CloudSim üzerinde ki 30 adet girdi ile süreç performansını ortalama süresi gösterilmektedir.



Şekil 4.1 Tüm Algoritmaların Süreç Performans Ortalaması.



Şekil 4.2 Tüm Algoritmaların Başlangıç ve Bitişlerine ait Süreç Performansı

Tablo 4.1 Tüm Algoritmaların Başlangıç ve Bitişlerine ait Süreç Bilgileri

GWO		**PSO**		**RR**		**FCFS**		**SJSF**	
Start Time (ms)	Finish Time (ms)	Start Time (ms)	Finish Time (ms)	Start Time (ms)	Finish Time (ms)	Start Time (ms)	Finish Time (ms)	Start Time (ms)	Finish Time (ms)
0,20	525,25	0,10	654,07	0,10	973,17	0,20	2315,67	0,10	1307,99
0,20	962,95	0,10	1207,16	0,10	1168,10	0,20	3363,63	0,10	2150,41
0,20	972,29	0,10	1307,99	0,10	2748,74	0,20	2191,80	0,10	2233,50
0,20	973,64	0,10	1827,72	0,10	2929,42	0,20	1393,29	0,10	2589,59
0,20	985,59	0,10	2147,86	0,10	3199,00	2191,80	3841,00	0,10	3390,68
525,25	1700,86	1207,16	2662,50	1168,10	4213,29	1393,29	3402,86	2589,59	3554,99
962,95	1987,84	1827,72	2945,98	2929,42	4410,27	3363,63	6458,91	2233,50	3652,40
972,29	1999,32	2147,86	3138,78	973,17	4920,08	0,20	881,20	2150,41	3735,12
973,64	2145,60	654,07	3773,34	2748,74	5181,34	881,20	3913,45	3652,40	4053,84
985,59	2224,70	3138,78	4121,64	4920,08	5902,94	3913,45	5813,84	1307,99	4313,54

1700,86	3015,92	1307,99	4287,26	3199,00	6377,72	3402,86	4887,72	3735,12	4792,70
1987,84	3073,89	2945,98	4426,83	5181,34	6766,06	3841,00	6124,32	3390,68	5345,68
1999,32	3098,71	4121,64	5264,48	4410,27	6917,63	5813,84	7459,27	3554,99	5798,03
2145,60	3152,07	3773,34	5337,59	6766,06	7823,63	6124,32	9411,24	4313,54	5965,19
2224,70	3452,89	4287,26	5781,94	6377,72	8164,19	4887,72	5566,78	4053,84	6452,89
3015,92	5012,11	5337,59	5942,62	4213,29	8562,43	5566,78	6214,35	5345,68	7063,38
3073,89	5085,55	5264,48	6642,25	7823,63	8660,08	6214,35	9345,36	5798,03	7093,40
3098,71	5120,86	5942,62	6779,08	8164,19	9472,08	7459,27	9652,03	4792,70	7351,06
3152,07	5250,55	4426,83	7194,19	5902,94	9952,74	9652,03	12953,46	5965,19	8607,92
3452,89	5800,11	5781,94	7433,60	8562,43	10017,78	2315,67	3100,32	7351,06	9780,38
5012,11	7810,80	6779,08	8307,33	8660,08	10188,33	12953,46	13848,63	7063,38	10713,90
5085,55	7872,67	6642,25	9611,82	6917,63	10537,28	6458,91	8629,40	7093,40	11442,54
5120,86	8117,49	7194,19	9701,55	10017,78	12187,18	8629,40	11704,63	8607,92	11541,56

5250,55	8617,87	7433,60	10237,21	10188,33	12617,65	3100,32	6190,26	10713,90	11696,76
5800,11	9142,98	10237,21	10743,08	10537,28	12936,33	6190,26	7367,75	9780,38	13102,76
7810,80	9422,22	8307,33	10809,36	9952,74	13319,60	13848,63	16622,12	11541,56	13383,04
7872,67	9823,79	10743,08	11363,18	9472,08	13852,28	9345,36	12849,95	11442,54	15164,54
8117,49	9925,43	9701,55	12093,48	13852,28	16037,96	7367,75	8221,94	15164,54	17273,76
8617,87	1078,18	11363,18	12751,08	16037,96	18308,06	11704,63	14169,00	17273,76	19443,16
9142,98	12828,43	12751,08	14339,05	18308,06	19896,03	16622,12	19122,36	19443,16	23144,87

5. Sonuç ve Gelecek Çalışmalar

Sonuç ve gelecek çalışmaları ele alınan konuları ayrıştırarak değerlendirmek daha anlamlı olacağı düşünüldüğünden görev zamanlama algoritmalar üzerinden, kuantum sonrası güvenlik önlemleri ve kuantum sonrası performans olarak değerlendirilmiştir. Bu çalışmada bulut bilişim ve kuantum sonrası imzalama şemaları ile ilgili temel çalışmalar ve araştırılmalar yapılarak gelecek çalışmalar için kuantum sonrası bulut bilişimin kapıları aralanmıştır.

5.1 Görev Zamanlama Algoritmaları ile İlgili Sonuçlar ve Gelecek Çalışmalar

Sonuç olarak sezgi üstü algoritmaların üstünlüğü açık şekilde görülmektedir. Gelecek çalışmalar olarak hibrit sistemler üzerinde çalışılabilir. Particle Swarm Optimization (PSO)-Parçacık Sürü Optimizasyonu ve Grey Wolf Optimization (GWO)- Gri Kurt Optimizasyonu algoritmalarından iyi bir performans gözlemlendiği için daha da geliştirmek adına aynı yönelimlere sahip sezgi üstü algoritmalar ile sentezlenerek hibrit algoritmalar geliştirilerek daha iyi sonuçlar elde edilmesi sağlanabilir.

5.2 Kuantum Sonrası İmzalama Aşamaları İlgili Sonuçlar ve Gelecek Çalışmalar

Sonuç olarak amacımız kuantum sonrası için güvenli sistemler oluşturmak. Fakat sadece güvenli olması yeterli olmuyor artık güvenlik düşük ölçekli cihazlardan, yüksek kapasiteli cihazlara kadar ciddi bir önem arz etmekte ve bu yüzden verimlilikte güvenli olması kadar önemli bir kıstas haline geliyor. Bu yüzden gelecek çalışmalar açısından sistemleri ele alırken dezavantajlarını azaltacak verimliliği artıracak iyileştirmeler düşünülmelidir.

Anlatılan sistemlerden UOV un dezavantajları olan anahtar boyutunun büyüklüğünden kaynaklanan yavaşlık probleminin çözümü için Rainbow sisteminde liner dönüşüm sayısı artırılarak anahtar boyutu küçültmüş daha da iyileştirmek için yine liner dönüşüm sayısı artırılabilir. Yine Rainbow un hızlı olmasının sebeplerinden olan katmanlar arası bağımlılığı, sistemi hızlandırmak adına daha da artırılabilir.

LUOV için açık anahtarı diğer çok değişkenli imzalama sistemlerine göre oldukça küçük boyutta fakat dezavantaj oluşturabilecek kısım açık anahtarın kalan kısmı birçok kuantum sonrası imzalama şemalarının açık anahtarından daha büyük, daha iyi

bir açık anahtar sistemi ile değiştirilme yaparak ya da optimize edilmiş r genişletme miktarı kullanılarak iyileştirme yapılabilir.

DualModeMS için ise avantaj ve dezavantajlarına bakarsak bu yapı çok az yer kaplıyor bu bir avantaj, güvenlik konusunda da oldukça iyi fakat gizli anahtar boyutu fazla büyük ve bu da imza doğrulama aşamasının çok fazla zaman almasına sebep oluyor. Dezavantajlarını azaltmak için programlama da "λ" olarak gösterilen iç katmanın güvenlik seviyesini düşürülerek güvenlikten kısmi bir feragat edilerek boyut da azaltma yapılabilir, yine imza üretimi için kullanılan S ve T matrislerinde küçültme ve iyileştirme yapılabilir. Aslında imza boyutunun büyüklüğünden en çok imza doğrulama aşamasının uzun sürmesinden dolayı şikayetçiyiz, bu aşamayı iyileştirmek için de imza doğrulama aşamasında yinelemeli yapılar olduğu için program paralel şekilde programlanabilir.

5.3 Kuantum Sonrası Bulut Bilişim Performansı ile İlgili Sonuçlar ve Gelecek Çalışmalar

Bu çalışmada kuantum sonrasını güvenlik önlemleri için ele alınmıştır. Gelecek çalışmalarda kuntum cihazların gerek maliyetinden gerekse de yapısı gereği muhafazası zor bir teknoloji olması sebebi ile bu sistemlerden kurulu bir altyapı üzerine bulut sistemleri ile son kullanıcılara bu performansı iletilebilmesini sağlamak için fikirler varken bunun yanında bu sistemi daha etkin kullanılabilmesi için bu alanda görev zamanlama algoritmaları geliştirilerek bulut bilişime yeni kapıların açılması sağlanabilir.

6. Kaynaklar

Aiello, M., Catarci, T. & Mecella, M. (2009). Smart Homes Infrastructures and Interactions (SHII 2009). In 2009 18TH IEEE INTERNATIONAL WORKSHOP ON ENABLING TECHNOLOGIES: INFRASTRUCTURES FOR COLLABORATIVE ENTERPRISES (pp. 234-235). (IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises Proceedings). IEEE (The Institute of Electrical and Electronics Engineers).

Al-Anie, H. K., Alia, M. A. & Hnaif, A. A. (2011). E-voting protocol based on public-key cryptography. *International Journal of Network Security & Its Applications*, *3*(4), 87-98.

Alghamdi, M. I. (2022). Optimization of load balancing and task scheduling in cloud computing environments using artificial neural networks-based binary particle swarm optimization (BPSO). *Sustainability*, *14*(19), 11982.

Alhaidari, F. & Balharith, T. Z. (2021). Enhanced round-robin algorithm in the cloud computing environment for optimal task scheduling. *Computers*, *10*(5), 63.

Alhaidari, F., Balharith, T. & Eyman, A. Y. (2019, April). Comparative analysis for task scheduling algorithms on cloud computing. In 2019 International Conference on Computer and Information Sciences (ICCIS) (pp. 1-6). IEEE.

Anonim Multivariate Quadratic Public-Key Cryptography Part 1: Basics Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A. ve Zaharia, M. (2010). A view of cloud computing. *Communications of the ACM*, *53*(4), 50-58.

Anonim, 2021 https://www.geeksforgeeks.org/what-is-cloudsim/

Bala, R., Gill, B., Smith, D., Wright, D. & Ji, K. (2021). Magic quadrant for cloud infrastructure and platform services. *Gartner (July 27, 2021)*.

Bocaniala, C. D. & da Costa, J. S. (2006). Application of a novel fuzzy classifier to fault detection and isolation of the DAMADICS benchmark problem. *Control engineering practice*, *14*(6), 653-669.

Casanova, A., Faugere, J. C., Macario-Rat, G., Patarin, J., Perret, L. & Ryckeghem, J. (2017). *GeMSS: a great multivariate short signature* (Doctoral dissertation, UPMC-Paris 6 Sorbonne Universités; INRIA Paris Research Centre, MAMBA Team, F-75012, Paris, France; LIP6-Laboratoire d'Informatique de Paris 6).

Czypek, P. (2012). Implementing multivariate quadratic public key signature schemes on embedded devices. *Diss. Ph. D. thesis, Diploma Thesis, Chair for Embedded Security, RUB*.

Ç. Aladağ, (2009). Yapay Sinir Ağlarının Mimari Seçimi İçin Tabu Algoritması, Doktora Tezi. Ankara: Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü.

Duong, D. H., Petzoldt, A., Wang, Y. & Takagi, T. (2017). Revisiting the cubic UOV signature scheme. In *Information Security and Cryptology–ICISC 2016: 19th International Conference, Seoul, South Korea, November 30–December 2, 2016, Revised Selected Papers 19* (pp. 223-238). Springer International Publishing.

Gaidhane, P. J. & Nigam, M. J. (2018). A hybrid grey wolf optimizer and artificial bee colony algorithm for enhancing the performance of complex systems. *Journal of computational science, 27*, 284-302.

Garfinkel, S. (1999). *Architects of the information society: 35 years of the Laboratory for Computer Science at MIT*. MIT press.

Ghoshal, S. P. (2004). Optimizations of PID gains by particle swarm optimizations in fuzzy based automatic generation control. *Electric power systems research, 72*(3), 203-212.

Giannoutakis, K. M., Filelis-Papadopoulos, C. K., Gravvanis, G. A. & Tzovaras, D. (2021). Evaluation of self-organizing and self-managing heterogeneous high performance computing clouds through discrete-time simulation. *Concurrency and Computation: Practice and Experience, 33*(17), e6326.

Goyal, T., Singh, A. & Agrawal, A. (2012). Cloudsim: simulator for cloud computing infrastructure and modeling. *Procedia Engineering*, *38*, 3566-3572.

Guerra, F. A. & Coelho, L. D. S. (2008). Multi-step ahead nonlinear identification of Lorenz's chaotic system using radial basis neural network with learning by clustering and particle swarm optimization. *Chaos, Solitons ve Fractals*, *35*(5), 967-979.

Hakan Uzuner 2020, FOG, EDGE ve Mist Computing Nedir? https://www.hakanuzuner.com/fog-edge-ve-mist-computing-nedir/

Hou, Y., Gao, H., Wang, Z. & Du, C. (2022). Improved grey wolf optimization algorithm and application. *Sensors*, *22*(10), 3810.

How, H. T., Liew, T. H., Kuan, E. L., Yang, L. L. & Hanzo, L. (2006). A redundant residue number system coded burst-by-burst adaptive joint-detection based CDMA speech transceiver. *IEEE transactions on vehicular technology*, *55*(1), 387-396. https://2017.pqcrypto.org/school/slides/1-Basics.pdf

Huang, X., Li, C., Chen, H. & An, D. (2020). Task scheduling in cloud computing using particle swarm optimization with time varying inertia weight strategies. *Cluster Computing*, *23*(2), 1137-1147.

Huang, Y. J., Liu, F. H. & Yang, B. Y. (2012). Public-key cryptography from new multivariate quadratic assumptions. In *Public Key Cryptography–PKC 2012: 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012. Proceedings 15* (pp. 190-205). Springer Berlin Heidelberg.

İnaç, T., Dokur, E., & Yüzgeç, U. (2022). A multi-strategy random weighted gray wolf optimizer-based multi-layer perceptron model for short-term wind speed forecasting. *Neural Computing and Applications*, *34*(17), 14627-14657.

Kanani, B. & Maniyar, B. (2015). Review on max-min task scheduling algorithm for

cloud computing. *Journal of emerging technologies and innovative research*, *2*(3), 781-784.

Khan, M. S. A. & Santhosh, R. (2022). Task scheduling in cloud computing using hybrid optimization algorithm. *Soft computing*, *26*(23), 13069-13079.

Khan, T., Tian, W., Zhou, G., Ilager, S., Gong, M. & Buyya, R. (2022). Machine learning (ML)-centric resource management in cloud computing: A review and future directions. *Journal of Network and Computer Applications*, *204*, 103405.

Koç, T., Eke, İ., & Tezcan, S. S. (2021). Parçacık Sürü Optimizasyonu ve Genetik Algoritma Kullanılarak Birleşik Isı ve Güç Ekonomik Dağıtımı Probleminin Çözümü. *International Journal of Engineering Research and Development*, *13*(3), 230-241.

Li, C., Qouneh, A. & Li, T. (2011, June). Characterizing and analyzing renewable energy driven data centers. In *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems* (pp. 131-132).

Li, X., Zheng, M. C., Ren, X., Liu, X., Zhang, P. & Lou, C. (2014). An improved task scheduling algorithm based on potential games in cloud computing. In *Pervasive Computing and the Networked World: Joint International Conference, ICPCA/SWS 2013, Vina del Mar, Chile, December 5-7, 2013. Revised Selected Papers* (pp. 346-355). Springer International Publishing.

Masdari, M., Salehi, F., Jalali, M. & Bidaki, M. (2017). A survey of PSO-based scheduling algorithms in cloud computing. *Journal of Network and Systems Management*, *25*(1), 122-158.

Mora, H., Abdullahi, S. E. & Junaidu, S. B. (2017, November). Modified median round robin algorithm (MMRRA). In *2017 13th international conference on electronics, computer and computation (ICECCO)* (pp. 1-7). IEEE.

Özsağlam, M. Y. & Çunkaş, M. (2008). Optimizasyon problemlerinin çözümü için parçaçık sürü optimizasyonu algoritması. *Politeknik Dergisi*, *11*(4), 299-305.

Perret, L. DualModeMS: A Dual Mode for Multivariate-based Signature 20170918 draft.

Petzoldt, A. (2017). On the complexity of the hybrid approach on HFEv. *Cryptology ePrint Archive*.

Pattanaik, P. A., Roy, S., & Pattnaik, P. K. (2015, February). Performance study of some dynamic load balancing algorithms in cloud computing environment. In *2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN)* (pp. 619-624). IEEE.

Serkan, K. A. Y. A. & FIĞLALI, N. (2016). Çok amaçlı optimizasyon problemlerinde Pareto optimal kullanımı. *Sosyal Bilimler Araştırma Dergisi*, *5*(2), 9-18.

Smith-Tone, D. NIST Post-Quantum Standardization Project Round 1 Multivariate Digital Signature Candidates Survey.

Soltani, N., Soleimani, B. & Barekatain, B. (2017). Heuristic algorithms for task scheduling in cloud computing: a survey. *International Journal of Computer Network and Information Security*, *11*(8), 16.

Sultan, N. (2010). Cloud computing for education: A new dawn?. *International Journal of Information Management*, *30*(2), 109-116.

Surbiryala, J. & Rong, C. (2019, August). Cloud computing: History and overview. In *2019 IEEE Cloud Summit* (pp. 1-7). IEEE.

Tawfeek, M. A., El-Sisi, A., Keshk, A. E. & Torkey, F. A. (2013, November). Cloud task

scheduling based on ant colony optimization. In *2013 8th international conference on computer engineering and systems (ICCES)* (pp. 64-69). IEEE.

Wang, X., & Chen, M. (2008, February). Cluster-level feedback power control for performance optimization. In *2008 IEEE 14th International Symposium on High Performance Computer Architecture* (pp. 101-110). IEEE.

Yang, B. Y. Multivariate Quadratic Public-Key Cryptography Part 3: Small Field Schemes.

Zhou, J., Duan, Z., Li, Y., Deng, J. & Yu, D. (2006). PSO-based neural network optimization and its utilization in a boring machine. *Journal of Materials Processing Technology*, *178*(1-3), 19-23.