ACCLOUD-MAN — Heterojen Bulutlarda Güç Etkin Kaynak Ataması
ACCLOUD-MAN — Power Efficient Resource Allocation for Heterogeneous Clouds

Nazım Umut Ekici*†, Klaus Werner Schmidt†, Alper Yazar*†, Ece Güran Schmidt†

† Elektrik Elektronik Mühendisliği Bölümü, ODTÜ, Ankara, Türkiye
{nazim.ekici, schmidt, alper.yazar, eguran}@metu.edu.tr
* Savunma Sistem Teknolojileri, ASELSAN A.Ş., Ankara, Türkiye
{uekici, ayazar}@aselsan.com.tr

***Özetçe* —Bu bildiride heterojen bulut veri merkezleri için ACCLOUD-MAN isimli yenilikçi bir kaynak yönetici önerilmektedir. Heterojen bulutlarda bir kullanıcı isteği birden fazla fiziksel kaynak alternatifiyle karşılanabilir. Kaynak yönetici, isteğin hangi sunucuya atanacağının kararıyla birlikte hangi kaynak alternatifiyle karşılanacağına da karar vermelidir. ACCLOUD-MAN'ın kaynak yönetim hedefi buluttaki güç tüketimini en aza indirmektir. Yönetici bir Tamsayı Doğrusal Problem olarak modellendi ve MATLAB üzerinde simülasyon altyapısı ile birlikte gerçeklendi. Simülasyon sonucunda çözüm sürelerinin pratik uygulamalar için çok uzun olduğu görüldü. Süreleri iyileştirmek için isteklerin daha küçük gruplar halinde işlenmesi fikri öne sürüldü. Bu yöntemin başarım kaybı pahasına süreleri iyileştirdiği görüldü.**

***Anahtar Kelimeler*—Bulut bilişim, Veri merkezi, Donanım hızlandırıcı, Kaynak yönetimi, Çevreci bilişim**

***Abstract*—In this paper we propose ACCLOUD-MAN, a novel resource manager for heterogeneous cloud data centers. In heterogeneous clouds a user request can be satisfied with more than one physical resource alternative. Resource manager must decide which resource alternative will be chosen, along with the decision of the server the request will be assigned to. ACCLOUD-MAN's resource management objective is to reduce the power consumption of the cloud. Manager is modeled as an Integer Linear Problem and is implemented on MATLAB, together with a simulation platform. Simulation results show that the solution times are too long for practical applications. Dividing requests into smaller groups is suggested to improve the timings. It is seen that this method improves the timings with a penalty in decision performance.**

***Keywords*—Cloud computing, Data center, Hardware accelerator, Resource management, Green computing**

I. Giriş

Veri merkezlerinde bulunan fiziksel makine kaynakları, bulut bilişim sistemleri ile kullanıcılara sanallaştırılarak dinamik bir şekilde sunulur. Bu doğrultuda *kaynak atama yöntemleri* kullanıcıların isteklerini karşılayacak bir sanal konfigürasyonu farklı başarım hedefleri ve kısıtlarına göre belirleyerek mevcut fiziksel kaynaklarla en iyi şekilde oluşturmayı hedefler. Klasik bulut mimarilerine FPGA tabanlı donanım hızlandırıcılar (hardware accelerator-HA), [1], yanı sıra GPU (Graphics Processing Unit) ve TPU (Tensor Processing Unit) gibi yeni hesaplama kaynakları eklenerek oluşturulan heterojen bulut mimarileri hem akademi hem de uygulama alanında yeni bir konudur.

Heterojen yapıdaki bulutlarda kaynak atama yöntemlerinin CPU, hafıza, disk, bant genişliği (BW-bandwidth) gibi standart kaynaklara ek olarak yeni eklenen donanım kaynaklarını da kullanıcılara atayacak şekilde genişletilmesi gereklidir. Daha da önemlisi bulut yazılım servislerinde (Software as a Service-SaaS) kullanıcı iş isteklerine hangi fiziksel kaynakların atanacağına bulut kaynak yöneticisi karar vermektedir. Heterojen bulutlarda SaaS kullanıcı istekleri bu yeni tip hesaplama kaynakları ya da standart sunuculardaki işlemciler ile, farklı başarım ve kaynak kullanım bedeli ile karşılanabilir. Kaynak atamalarının en iyi çözüm ile ve istek geliş dinamiğine uygun hızlarda yapılması gereklidir.

Bu bildiride literatürdeki çalışmalardan farklı olarak bulut bilişim sistemlerindeki yeni hesaplama kaynaklarını kapsayan ve kullanıcı iş isteklerini karşılayacak farklı atama alternatiflerini birlikte değerlendirerek mevcut kaynaklara en iyi şekilde atayan yeni bir kaynak yöneticisi olan ACCLOUD-MAN (ACcelerated CLOUD MANagement) sunulmakta ve değerlendirilmektedir. Mevcut geliştirme safhasında ACCLOUD-MAN yeni tip hesaplama kaynağı olarak FPGA tabanlı donanım hızlandırıcıları içermekte ve çevreci bilişim kapsamında güç tüketimini en aza indirecek şekilde kaynak ataması yapmaktadır.

II. Bulut Bilişim Kaynak Ayırımı ve Önceki Çalışmalar

Bulut sistemlerinde temel olarak 3 farklı seviyede kullanıcılara hizmet verilir. Bunlar: Altyapı (Infrastructure as a Service-IaaS), Platform (Platform as a Service- PaaS) ve Yazılım (Software as a Service-SaaS) Olarak Servis'tir [2]. IaaS seviyesinde kullanıcılara istekleri doğrultusunda CPU, hafıza, disk ve ağ gibi kaynaklar ayrılmış, istenilen işletim sistemini koşturan sanal makinelar (VM) verilir. PaaS seviyesinde bulut sağlayıcısı çeşitli yazılımsal servislerin idamesini de üstlenerek bir

978-1-7281-1904-5/19/$31.00

hizmet sağlar. SaaS hizmetinde ise kullanıcılar kendi verileri ile sağlanan uygulamaları doğrudan, bir geliştirme yapmadan kullanır.

Literatürdeki kaynak yönetimi çalışmaları verilen bir isteği belli bir kaynak konfigürasyonu ile farklı olası fiziksel kaynaklara en iyi şekilde atamayı amaçlamaktadır. [3] enerjiyi ya da görev tepki zamanını minimize eden bir formülasyon sunmakta, [4] ise farklı tipte sunuculardan oluşan bir bulut yapısında adil olma kriterine göre kaynak ataması yapmaktadır. Kaynak yönetimi NP-hard bir problemdir [5], [6]. [7], enerji ve kaynak kullanımını kestirmek ve iyileştirmek için formüle ettiği çok amaçlı optimizasyon problemini Genetik Algoritma ile meta-heuristic (meta buluşsal) biçimde çözmektedir. [8] bulut veri merkezleri için ağ cihazlarının enerji kullanımını da kapsayan kestirim tabanlı buluşsal bir kaynak yönetimi önermektedir.

Bilgimize göre literatürde heterojen bulutlardaki yeni hesaplama kaynaklarını kapsayan bir çalışma olmadığı gibi SaaS için kullanıcı isteklerini saf yazılımda ya da donanım hızlandırıcılarını da içeren farklı fiziksel konfigürasyonlarda karşılayacak alternatifleri birlikte değerlendiren bir kaynak atama yöntemi bulunmamaktadır.

III. ACCLOUD-MAN Formülasyonu

A. Bulut Bilişim Kaynak Yönetimi

Bu çalışmada İşlemci (cpu), FPGA ($fpga$), Hafıza, Disk ve Ağ genişliği olarak 5 fiziksel kaynak tipi ele alınmakla birlikte yaklaşımımız herhangi bir sayıda kaynak çeşidi için kullanılabilir. FPGA kaynağının önceki çalışmamızda, [1], sunulduğu şekilde sanallaştırıldığı ve modül adı verilen bağımsız parçalar halinde donanım hızlandırıcı olarak işlemciyle birlikte veya tek başına kullanıcıya sunulduğu varsayılmıştır. ACCLOUD-MAN kaynak yöneticisinin iki temel hedefi vardır. Birincisi güç tüketimini en aza indirmek, ikincisi ise atama kararını bulutun çalışmasını aksatmayacak biçimde gerçek zamana yakın bir hızda yapmaktır. Güç tüketimi Bölüm III-B'de modellendiği gibi mevcut durumda çalışan düğümlere yeni işler atandığında artmaktadır. Mevcut çalışan sunucularda yeterli kaynak bulunmadığı durumda yeni sunuculara güç verilmesi gerekmektedir. Bu güç koşan işten bağımsız sabit bir değerdir. İş atandığında yeni açılan sunucunun güç kullanımı ayrıca artacaktır.

Kullanıcılar buluttan iki şekilde kaynak isteyebilirler. Bölüm II'deki servis modellerine uygun olarak IaaS/PaaS (IPaaS) istekler için kullanıcı buluttan istediği fiziksel kaynak miktarını açık olarak belirtir. SaaS istekler için ise kullanıcı kullanmak istediği bulut uygulamasını iletir.

SaaS istekler için gereken fiziksel kaynak miktarı ACCLOUD-MAN tarafından belirlenir. Bir SaaS isteği karşılayacak birden fazla fiziksel kaynak alternatifi olabilir. Örneğin 1 GB boyutundaki bir dosyayı sıkıştırmakta kullanılacak uygulamanın koşması için bir (veya daha fazla) işlemci çekirdeği veya bir (veya daha fazla) FPGA modülü atanabilir. Çevreci bilişim kapsamında güç tüketiminin düşük olması hedefi göz önüne alındığında, güç ihtiyacı en az olan alternatifin seçilmesi bu hedefi sağlıyor gibi görünse de alternatifin gerektirdiği kaynaklar açık olan bir sunucuda bulunamıyor olabilir. Böyle bir durumda güç ihtiyacı daha fazla olan alternatifin halihazırda açık olan bir sunucuya atanması daha az toplam güç tüketimi ile sonuçlanabilir. Bu sebeple kaynak yöneticisi yalnızca isteklerin hangi sunucuya atanacağına değil, hangi alternatifinin kullanılacağına da karar vermelidir. Bu çalışmada kullanıcının bulutta ne kadar süre kalacağının belirli olmadığı varsayılmaktadır.

ACCLOUD-MAN, SaaS istekleri için önceden bilinen kaynak alternatifleri arasından güç kullanımını en az artıran atamayı seçer. Alternatifler arasında kullanıcı tarafından gözlenebilen fark işin tamamlanma süresidir. SaaS kapsamında bulutta koşan yazılımların olabilecek farklı girdi tipleri için bir ön çalışma ile profillerinin çıkarıldığı ve bu profillere göre alternatiflerin servis sağlayıcı tarafından bir veritabanında tutulduğu varsayılmaktadır.

B. Kaynak Atama Modeli

ACCLOUD-MAN yöntemi için yukarıda tanımladığımız problem, Tamsayı Doğrusal Problem (Integer Linear Problem-ILP) olarak formüle edilmiştir. Buluttaki i numaralı sunucuda kullanılabilecek işlemci, FPGA, hafıza, disk ve ağ bant genişliği (bandwidth) kaynakları sırasıyla c_i, f_i, m_i, d_i, b_i olarak tanımlanmıştır.

$\widehat{REQ_j} = \{req_{j,1}, req_{j,2}, \ldots, req_{j,n}\}$ gelen j numaralı istek içinde fiziksel kaynak alternatiflerini barındıran bir kümedir. Her alternatif fiziksel kaynaklardan oluşur: $req_{j,n} = \{\hat{c}_{j,n}, \hat{f}_{j,n}, \hat{m}_{j,n}, \hat{d}_{j,n}, \hat{b}_{j,n}\}$. Bütün isteklerin kümesi REQ, bütün sunucuların kümesi PM olarak isimlendirilmiştir.

Karar değişkenleri aşağıdaki şekilde tanımlanmıştır:

- $s_{i,j,n}$ değişkeni j numaralı isteğin n numaralı alternatifinin i numaralı sunucuya atandığını gösterir. $\forall i \in PM$, $\forall j \in REQ$ ve $\forall n, 1 \leq n \leq |\widehat{REQ_j}|$, $s_{i,j,n}$ ikili bir değişkendir.
- pm_j değişkeni j numaralı isteğin atandığı sunucuyu gösterir. $0 \leq pm_j \leq |PM|$ ve $\forall j \in REQ$, $pm_j \in \mathbb{Z}$.
- alt_j değişkeni j numaralı isteğin atandığı alternatifi gösterir. $1 \leq alt_j \leq |\widehat{REQ_j}|$ ve $\forall j \in REQ$, $alt_j \in \mathbb{Z}$.
- q_i atamalar sırasında i numaralı sunucunun kullanım durumunu gösterir. $\forall i \in PM$, q_i ikili bir değişkendir.

IPaaS istekler için fiziksel kaynak gereksinimi bellidir ve tek bir alternatif vardır. SaaS istekler için yukarıda bahsedilen alternatif veritabanına uygun olarak alternatifler belirlenir.

Karar değişkenleri arasındaki ilişkileri gösteren aşağıdaki denklemler (1)-(5) sunucuların kendilerine yapılan atamaları karşılayacak kaynakları olmasını sağlar.

$$\sum_{j} \sum_{n=1}^{|\widehat{REQ_j}|} \hat{c}_{j,n} s_{i,j,n} \leq c_i \quad \forall i \in PM \tag{1}$$

$$\sum_{j} \sum_{n=1}^{|\widehat{REQ_j}|} \hat{f}_{j,n} s_{i,j,n} \leq f_i \quad \forall i \in PM \tag{2}$$

$$\sum_{j} \sum_{n=1}^{|\widehat{REQ_j}|} \hat{m}_{j,n} s_{i,j,n} \leq m_i \quad \forall i \in PM \tag{3}$$

$$\sum_{j} \sum_{n=1}^{|\widehat{REQ}_j|} \hat{d}_{j,n} s_{i,j,n} \leq d_i \quad \forall i \in PM \tag{4}$$

$$\sum_{j} \sum_{n=1}^{|\widehat{REQ}_j|} \hat{b}_{j,n} s_{i,j,n} \leq b_i \quad \forall i \in PM \tag{5}$$

Eğer i numaralı sunucu atama için kullanıldıysa $q_i = 1$ olmalıdır. M_1 yeterince büyük bir sabittir.

$$\sum_{j} \sum_{n=1}^{|\widehat{REQ}_j|} s_{i,j,n} \leq q_i \cdot M_1 \quad \forall i \in PM \tag{6}$$

Her istek yalnızca bir alternatif kullanılarak yalnız bir sunucuya atanır.

$$\sum_{i} \sum_{n=1}^{|\widehat{REQ}_j|} s_{i,j,n} = 1 \quad \forall j \in REQ \tag{7}$$

Denklemler (8) ve (9) sırasıyla pm_j ve alt_j değişkenlerini j numaralı isteğin atandığı sunucunun numarasına ve seçilen alternatifin numarasına eşitler.

$$pm_j = \sum_{i} \sum_{n=1}^{|\widehat{REQ}_j|} (s_{i,j,n} \cdot i) \quad \forall j \in REQ \tag{8}$$

$$alt_j = \sum_{i} \sum_{n=1}^{|\widehat{REQ}_j|} (s_{i,j,n} \cdot n) \quad \forall j \in REQ \tag{9}$$

Bulutun anlık güç tüketim miktarı açılan her yeni sunucuyla ve iş atanan her CPU çekirdeği ve FPGA modülü ile artar. Kullandığımız güç tüketim artışı modelinde; ON_i parametresi i numaralı sunucunun atama öncesindeki açık olma durumunu göstermektedir, sunucu açıksa 1'e kapalıysa 0'a eşittir. $(1 - ON_i)q_i$ ifadesi eğer sunucu kapalıysa ve bu atama sırasında iş atandıysa 1 aksi taktirde 0 olmaktadır. P_{ONi} i numaralı sunucunun açık olma güç tüketimini, P_{CPUi} ve P_{FPGAi} sunucunun iş atanan CPU çekirdeği ve FPGA modülü başına güç tüketimini gösterir. Buna göre P_{NewPM} terimi, yeni atamalarla açılan sunucuların açık olma güç tüketimi toplamlarına eşittir. $P_{compute}$ terimi, yeni iş atanan CPU çekirdekleri ve FPGA modüllerinin tüketecekleri güç miktarına eşittir. Bu terimlerle F'in minimize edilmesi, yapılacak atamalar sonrasında buluttaki güç tüketimi artışının en az olması anlamına gelir.

Bu güç modeli ile, amaç fonksiyonu (objective function) F aşağıdaki şekilde tanımlanmıştır.

$$P_{NewPM} = \sum_{i} P_{ONi}(1 - ON_i)q_i \tag{10}$$

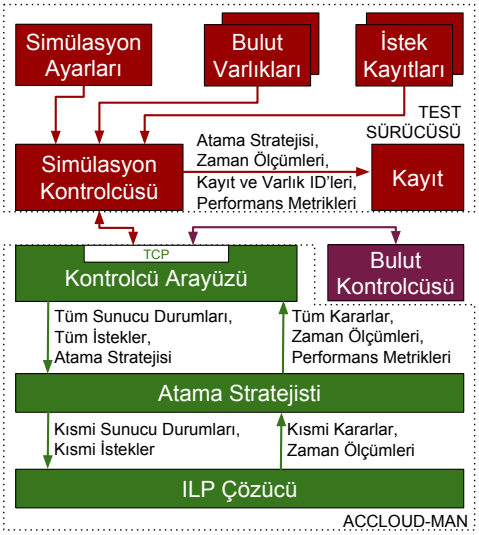
$$P_{compute} = \sum_{ijn} \left(P_{CPUi} \hat{c}_{j,n} + P_{FPGAi} \hat{f}_{j,n} \right) s_{i,j,n} \tag{11}$$

$$\min \quad F = P_{NewPM} + P_{compute} \tag{12}$$

Yukarıdaki formulasyonda PM ve REQ kümelerinin büyüklüğü sırası ile N ve M ile gösterilirse; her istek için tek alternatif olduğu durumda kaynak atama problemi $\mathcal{O}(N^M)$ hesaplama karmaşıklığına sahiptir. İstekler için ortalama L alternatif olduğu durumda bu karmaşıklık $\mathcal{O}((NL)^M)$'dir. Bu hesaplama karmaşıklığını düşürmek için, istekler daha küçük gruplara bölünebilir ve bu şekilde işlenebilirler. İsteklerin k'lı gruplara bölündüğü varsayılırsa işlenmesi gereken $M' = \lceil M/k \rceil$ grup mevcuttur. Her grup için $(NL)^k$ kombinasyon, toplamda ise $M'(NL)^k$ kombinasyon mevcuttur. Bu şekilde yeni hesaplama karmaşıklığı $\mathcal{O}(M(NL)^k)$ olacaktır. Polinom bir karmaşıklık olduğu için daha iyi bir ölçeklenme beklenir. Ancak isteklerin bölünmesiyle alınan kararların en iyi çözümden sapması da beklenmelidir.

IV. Simülasyon Gerçekleştirimi

Yukarıdaki formulasyon, Şekil 1'de görüldüğü gibi MATLAB üzerinde bir simülasyon ortamı ile birlikte gerçeklenmiştir. Simülasyon Kontrolcüsü Bulut Varlıkları veritabanından aldığı fiziksel ve yazılımsal kaynak bilgileri ile bir bulut veri merkezini simüle eder. İstek Kayıtları dosyasından okuduğu istek olaylarını Kontrolcü Arayüzü üzerinden ACCLOUD-MAN'a iletir ve gelen kararlara göre kaynak atamalarını canlandırır. Simülasyon Ayarları bloğu, hangi istek bölme algoritmalarının hangi istek ve varlık dosyaları ile test edileceğini Simülasyon Kontrolcüsü'ne belirtir. Kayıt bloğu yapılan tüm simülasyonların verilerini toplar ve kaynak atama stratejilerinin etkinliklerini kıyaslayıp raporlar. Kontrolcü Arayüzü, Simülasyon Kontrolcüsü ile bir TCP Soketi üzerinden haberleşir ve Atama Stratejistine arayüz sağlar. Atama Stratejisti bloğu gelen istekleri uygun şekilde gruplar ve ILP Çözücü bloğuna gönderir. ILP Çözücü bloğu istek grubunu sunuculara yukarıdaki formülasyona göre yerleştirir ve yerleştirme kararlarını Atama Stratejisti'ne iletir. Atama Stratejisti tüm istek grupları için verilen kararları topladıktan sonra dış dünyaya çıkarılmak üzere Kontrolcü Arayüzü'ne iletir.



Şekil 1: Bulut kaynak yöneticisinin yazılımsal mimarisi

Simulasyon gerçekleştirimin mevcut durumu TCP soket üzerinden simülasyon kontrolcüsünün yanı sıra gerçek bulut kaynak atamalarında kullanılan OpenStack yazılımı ile haberleşebilecek şekilde tasarlanmıştır [9].

V. ACCLOUD-MAN Değerlendirmeleri

Bölüm IV'da sunulan simülasyon ortamında ILP Çözücü Bloğu MATLAB'ın ILP çözücüsü ve MATLAB üzerinde ça-

TABLO I: TOMLAB CPLEX çözücü ile çözülen ILP'nin hesaplama süreleri. Sonuçlar saniye cinsindendir. Rastgele üretilen 100 istek seti ile oluşturulan problemler için ortalama sonuçlardır.

	Sunucu Sayısı							
İstek Sayısı	**10**	**20**	**50**	**100**	**150**	**200**	**300**	**400**
2	0.01	0.01	0.02	0.03	0.03	0.04	0.06	0.07
5	0.02	0.03	0.05	0.09	0.13	0.30	0.45	0.65
7	0.03	0.05	0.12	0.42	0.71	1.00	1.88	2.95
10	0.16	0.60	1.33	4.01	6.70	10.93	28.79	46.81

lışan TOMLAB yazılımına ait çözücüler ile gerçeklenmiştir [10]. Denenen 5 ILP çözücü içerisinden en hızlı çözüme ulaşan TOMLAB-CPLEX çözücüsü olmuştur. Bir sonraki en hızlı sonucu MATLAB ILP çözücüsü vermiştir, ancak TOMLAB-CPLEX ile aralarında ortalama 30 kat hız farkı mevcuttur. Bu nedenle mevcut gerçekleştirimde TOMLAB-CPLEX çözücüsü kullanılmaktadır.

Aktif bir buluta benzer bir davranış göstermesi için sunucular rastgele kaynak miktarları ile bir kısmı açık olacak şekilde ilklenmiştir. İstekler, rastgele miktarlarda gereksinimlere sahip ortalama 2 kaynak alternatifi ile üretilmiştir. İstenen kaynak miktarları 0.8 işlemci çekirdeği ve 0.4 arası FPGA modülü arasında düzgün dağılmıştır. İstek geliş zamanları arasında ve başlatılan sanal makinelerin bulutta kullanım süreleri düzgün bir dağılım ile rastgele belirlenmiştir.

Farklı sunucu sayıları ve aynı anda değerlendirilen istek sayılarına göre çözüm elde etme süreleri Tablo I'de gösterilmektedir. Tabloda gösterilenden daha büyük sunucu ve istek sayıları için çözüm üretme süreleri, makul süreleri aşmıştır. Örneğin, 50 sunucuya sahip bir buluta 20 isteğin yerleştirilmesi 70 saniye; 25 isteğin yerleştirilmesi ise 1990 saniye (33 dakika) sürmüştür.

Tablo I'de sunulan sonuçlara göre, Bölüm III-B'de bahsedilen istek gruplama yaklaşımı ile yapılan testlerde 200 sunucu için gelen 10 istek 5'erli 2 grup halinde atandığında çözüm 10.93 s yerine toplam 0.60 sn'de elde edilmektedir. Aynı atamalar 400 sunucu için aynı gruplamayla 46.81 sn yerine 1.30 sn'de tamamlanmaktadır. Sunucu sayısı ikiye katlandığında tüm isteklerin birlikte değerlendirildiği durumda işlem süresinin 4.28 kat, isteklerin 2'ye bölündüğü durumda ise 2.17 kat arttığı gözlenmiştir.

VI. Sonuç ve Gelecek Çalışmalar

Bu çalışmada, SaaS iş isteklerinin birden fazla fiziksel kaynak alternatifi ile karşılanabildiği bir bulut bilişim kaynak atama modeli ve hem SaaS ile birlikte IaaS/PaaS isteklerin sunuculara minimum güç tüketimi ile atayan ACCLOUD-MAN kaynak yöneticisi önerilmiştir. Kaynak atama problemi bir ILP problemi olarak formüle edilmiş ve TOMLAB/MATLAB çözücüleri ile çözülmüştür. İsteklerin bölünerek işlenmesinin çözüm zamanını çok önemli ölçüde azalttığı gözlenmiştir.

Bir sonraki adımda istek bölme yaklaşımlarının analitik yöntemlerle ve simülasyon aracılığıyla incelenmesi ile en iyi çözümden en az sapacak şekilde sistematik oluşturulması ele alınacaktır. Aynı kullanıcıdan gelen isteklerin aynı sunucuya atanarak bantgenişliği kullanımının iyileştirilmesi, istek geliş zamanlarının ve bitiş zamanlarının kestirilmesi de ACCLOUD-MAN'a eklenecek özellikler arasındadır. ACCLOUD-MAN önce simülasyon ortamında gerçekçi istek iş yükü izleri ile arkasından gerçek bir bulut üzerinde gerçekleştirilerek değerlendirilecektir.

TEŞEKKÜR

Bu çalışma, 117E667-117E668 nolu proje kapsamında TÜBİTAK tarafından desteklenmektedir. Yazarlar desteklerinden dolayı TÜBİTAK'a ve ASELSAN A.Ş.'ye teşekkür eder.

Kaynaklar

[1] A. Yazar, A. Erol, and E. G. Schmidt, "Accloud (accelerated cloud): A novel fpga-accelerated cloud archictecture," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*. IEEE, 2018, pp. 1–4.

[2] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in *2009 Fifth International Joint Conference on INC, IMS and IDC*, Aug 2009, pp. 44–51.

[3] J. Cao, K. Li, and I. Stojmenovic, "Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers," *IEEE Transactions on Computers*, vol. 63, no. 1, pp. 45–58, Jan 2014.

[4] W. Wang, B. Liang, and B. Li, "Multi-resource fair allocation in heterogeneous cloud computing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 10, pp. 2822–2835, Oct 2015.

[5] A. Yousafzai, A. Gani, R. M. Noor, M. Sookhak, H. Talebian, M. Shiraz, and M. K. Khan, "Cloud resource allocation schemes: review, taxonomy, and opportunities," *Knowledge and Information Systems*, vol. 50, no. 2, pp. 347–381, Feb 2017.

[6] A. M. Caulfield, E. S. Chung, A. Putnam, H. Angepat, J. Fowers, M. Haselman, S. Heil, M. Humphrey, P. Kaur, J. Y. Kim, D. Lo, T. Massengill, K. Ovtcharov, M. Papamichael, L. Woods, S. Lanka, D. Chiou, and D. Burger, "A cloud-scale acceleration architecture," in *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2016.

[7] F. Tseng, X. Wang, L. Chou, H. Chao, and V. C. M. Leung, "Dynamic resource prediction and allocation for cloud data center using the multiobjective genetic algorithm," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1688–1699, June 2018.

[8] M. Tarahomi and M. Izadi, "A prediction-based and power-aware virtual machine allocation algorithm in three-tier cloud data centers," *International Journal of Communication Systems*, vol. 32, no. 3, p. e3870, 2019, e3870 IJCS-18-0389.R1. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/dac.3870

[9] "Open Source Software For Creating Private and Public Clouds," https://www.openstack.org/, accessed: 2019-02-02.

[10] "TOMLAB Optimization," https://tomopt.com/tomlab/, accessed: 2019-02-02.