2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC) | 978-1-6654-2463-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/COMPSAC51774.2021.00172

H-FFMRA: A Multi Resource Fully Fair Resources Allocation Algorithm in Heterogeneous Cloud Computing

Hamed Hamzeh[1], Sofia Meacham[1] Kashaf Khan[2] Angelos Stefanidis[3] and Keith Phalp[1]
[1]Faculty of Science and Technology, Bournemouth University, UK.
hamzehh@bournemouth.ac.uk, smeacham@bournemouth.ac.uk, kphalp@bournemouth.ac.uk
[2]British Telecom, Ipswich, UK, kashaf.khan@bt.com
[3] Xi'an Jiaotong-Liverpool University, China. Angelos.Stefanidis@xjtlu.edu.cn

***Abstract*—The allocation of multiple types of resources fairly and efficiently has become a substantial concern in state-of-the-art computing systems. Accordingly, the rapid growth of cloud computing has highlighted the importance of resource management as a complicated and NP-hard problem. Unlike traditional frameworks, in modern data centers, incoming jobs pose demand profiles, including diverse sets of resources such as CPU, memory, and bandwidth across multiple servers. Accordingly, the fair distribution of resources, respecting such heterogeneity appears to be a challenging issue. Furthermore, the efficient use of resources as well as fairness, establish trade-off that renders a higher degree of satisfaction for both users and providers. Dominant Resource Fairness (DRF) has been introduced as an initial attempt to address fair resource allocation in multi-resource cloud computing infrastructures. Dozens of approaches have been proposed to overcome existing shortcomings associated with DRF. Although all those developments have satisfied several desirable fairness features, there are still substantial gaps. Firstly, it is not clear how to measure the fair allocation of resources among users. Secondly, no particular trade-off considers non-dominant resources in allocation decisions. Thirdly, those allocations are not intuitively fair as some users are not able to maximize their allocations. In particular, the recent approaches have not considered the aggregate resource demands concerning dominant and non-dominant resources across multiple servers. These issues lead to an uneven allocation of resources over numerous servers which is an obstacle against utility maximization for some users with dominant resources. Correspondingly, in this paper, a resource allocation algorithm called H-FFMRA is proposed to distribute resources with fairness across servers and users, considering dominant and non-dominant resources. The experiments show that H-FFMRA achieves approximately %20 improvements on fairness as well as full utilization of resources compared to DRF in multi-server settings.**

***Index Terms*—Allocation, Cloud, Dominant, fairness, resource, server, utility.**

I. Introduction

Resource allocation with fairness in cloud computing has been widely considered to be the most challenging issue. Despite other distributed systems, cloud computing is particularly recognized in the heterogeneity of resources and servers [1]. In other words, a cloud data center is likely to be established by different servers, including diverse configurations in terms of resources, such as memory, processing, and disk storage [2]. Due to this heterogeneity, some jobs are CPU-intensive such as computational operations, while others are disk-intensive like database management. Accordingly, resource allocation in such specifications seems to be a significant and NP-hard problem.

The traditional data center management frameworks such as Hadoop slot scheduler [3] have ignored the resource heterogeneity as they only consider a single resource type. These approaches allocate resources in the form of partitions called slots with a constant amount of resources. Such an assumption leads to highly inefficient resource allocation [2]. To overcome this significant drawback, and as the first attempt, Dominant Resource Fairness (DRF) [4] has been proposed to deal with fair resource allocation in cloud computing. As a generalization of Max-Min fairness [5], DRF employs the progressive filling algorithm [6] to calculate shares by equalizing dominant resources. A dominant resource has been defined as the heaviest resource demand by a user. DRF has rapidly attracted much attention as it has achieved several desirable fairness features such as sharing-incentive, resource monotonicity, envy-free, Pareto-efficiency, and strategy-proof [4]. Despite these promising features, dozens of extensions have been proposed different solutions to address the potential problems associated with DRF such as works in [2] [7] [8]. The main focus of these approaches has been dealing with the trade-off between fairness and efficiency, as under the notion of DRF there is a resource wastage in some points [9].

In addition to the resource heterogeneity, the diversity of servers is obvious in modern data centers [10]. DRF assumes the whole system as a supercomputer where resources are shared in a single resource pool. In such a case, applying DRF separately in all servers causes inefficient allocation as some users may have dominant resources across different servers [2].

Unfortunately, there are substantial gaps associated with recent developments. Firstly, while Jain's fairness index is being widely used in single resource environments, there is a lack of measures to evaluate the fairness in cloud computing systems. Secondly, the fairness-efficiency trade-off in recent approaches has not taken into account non-dominant resources

978-1-6654-2463-9/21/$31.00 ©2021 IEEE
DOI 10.1109/COMPSAC51774.2021.00172

in allocation decisions. Moreover, under these mechanisms, some users' allocation regarding dominant resources is not maximized at all. This problem is noticeably phenomenal in server diversification as some users could be adversely affected concerning utility maximization. Consequently, recent approaches have satisfied Pareto-efficiency from a general perspective, nonetheless, to the best of our knowledge it is not satisfied individually for a certain number of users.

Taking into account all the above-mentioned issues, in this paper, we present H-FFMRA as an extension of our previous work [11] in multiple servers. H-FFMRA captures multi-resource allocation, considering dominant and non-dominant resources for each server. It inherits all principal allocation policies in FFMRA that are applicable in multi-server settings. The experiments, conducted in the Cloudsim and driven by randomly generated workloads confirm that not only FFMRA achieves full resource utilization, but also it obtains fully fair resource allocation.

This paper is organized as follows. Section II, gives an overview of related works in multi-resource allocation. Section IV, outlines H-FFMRA. The V section illustrates the fairness measures associated with H-FFMRA. Section VI analysis the performance of H-FFRA allocation. Finally, section VII, concludes and discusses future work.

II. Related works

In [2] the generalization of DRF has been proposed, aiming to address the efficiency problem associated with DRF in heterogeneous servers. Despite [12], DRFH has performed *global dominant shares* to achieve Max-Min fairness. DRFH satisfies some fairness features, nonetheless, it is unable to guarantee the *sharing -incentive* in certain points. Furthermore, DRFH fails to meet *bottleneck resource fairness* as it maximizes the minimum global dominant shares. In U-DRF [13], authors have investigated how DRF is achievable when users have multiple intricate jobs. Accordingly, tasks are dependent on each other with placement constraints. Similar to other approaches, UDRF has satisfied some desirable fairness properties. Based on the experiments, UDRF has presented better job completion time compared to DRF, as well as reducing resource wastage. Authors in [8] have proposed a new allocation model called PS-DSF as an extension of DRF that has taken into account the weight of allocated resources to every individual with respect to each server where the placement constraint is considered as well. PS-DSF has tried to assign the most efficient server to each user, leading to achieving higher resource utilization than other allocation mechanisms. Unfortunately, this mechanism does not satisfy the Pareto-efficiency feature.

III. Motivation

In FFMRA, the main focus was on a single server, whereas modern data centers are composed of multiple servers with distinct configurations. Applying both DRF, and FFMRA in a naive extension form, and separately across all servers violates Pareto-efficiency corresponding to DRF and intuitive fairness in FFMRA. Although, performing FFMRA individually in each server meets efficiency, users may receive an equal allocation based on their demand profiles. To have a better comprehension of the problem, we refer to the example in Fig. 1.

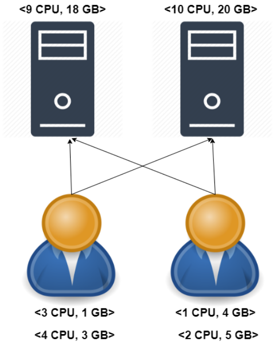


Fig. 1: An example of a system with two servers and two users each submits two tasks across both servers.

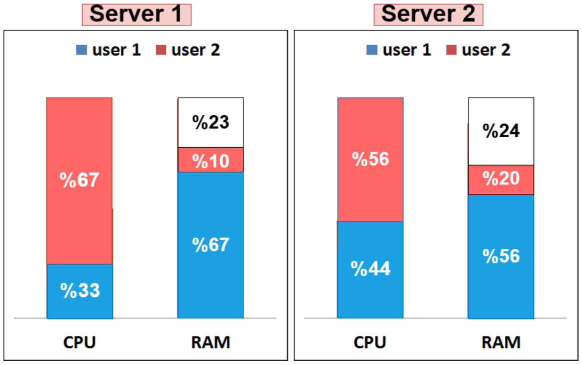


Fig. 2: The resource allocation under DRF mechanism

Assume there are two users let's say users 1, and 2 with demand vectors $(1CPU, 4GB)$, $(2CPU, 5GB)$, and $(3CPU, 1GB)$, $(4CPU, 3GB)$ respectively. Also, both users are eligible to run tasks in servers 1, and 2 with the capacity vectors $(9CPU, 18GB)$, and $(10CPU, 20GB)$. Accordingly, the dominant resources for user 1 in server 1, and 2 are memory $(4/9 > 1/18)$, and $(4/9, 3/18)$. For user 2, CPU is dominant in both servers as $(3/9, 1/18)$, and $(4/10 > 3/20)$. Intuitively, DRF tries to equalize both users' dominant resources in each server, assuming that DRF is naively employed. Therefore, under DRF policy user 1 receives $(3CPU, 12GB)$, and $(4.48CPU, 1.2GB)$ in server 2. On the other hand, user 2 schedules, $(6CPU, 2GB)$, and $(5.52CPU, 4.4GB)$ in servers 1, and 2 sequentially. Based on the figure, an inefficient allocation occurs in both servers, nearly 23% of wasted resources. Moreover, the distribution of resources among users corresponding to the dominant, and non-dominant shares is not the same percentage. The main assumption is to equalize all resources of users across multiple servers to achieve intuitive fairness. Therefore, according to the example in Fig. 3, users 1, and 2 benefit 69% of the entire resource pool in each server for dominant shares plus 31% equal percentage in all servers with regards to non-dominant shares.

A. System setting

Assume there are $|K|$ heterogeneous servers, $|K| = (s1, s2, ..., s_m)$ with capacity vector $C = (c_{1,s}, ..., c_{k,s})$. Also n users indicated by $U = (1, 2, ..., n)$ compete over k resource types as $R = (1, 2, ..., k)$ with resource request profiles, presented by $|R| = (r_{1,s}, r_{2,s}, ..., r_{n,s})$, where demands are always positive corresponding to each user. i.e, $r_{n,s} > 0, i \in U$.

Definition 1. *Consider resource k is the heaviest, and non-heaviest demand of submission r by user i over the maximum capacity of any server s. So, the dominant, and non-dominant resources for each user, specified by vectors $D = (d^k_{1,s}, ..., d^k_{i,s})$, and $\tilde{D} = (d^k_{1,s}, ..., d^k_{i,s})$ respectively, are determined as follows:*

$$d^k_{i,s} = max \frac{r_{i,s}}{C^{k_{max}}_s} \tag{1}$$

$$\tilde{d}^k_{i,s} = min \frac{r_{i,s}}{C^{k_{max}}_s} \tag{2}$$

Definition 2. *If the allocation for each user i,$(1 \leq i \leq n)$ in server s is specified by $\Pi_{i,s} = (\pi_{1,s}, ..., \pi_{n,s})$, with the utility $U_{i,s}$, then, the maximum number of tasks Ψ that a user can schedule in each server is the minimum allocation of a specific resource type k. So:*

$$\Psi_i(U_{i,s}){=}min \frac{\Pi_{i,s}}{r_{i,s}},U_{i,s}{=}max(\Pi_{i,s}) \implies \forall i{\in}U,\exists r_{i,s}{>}0. \tag{3}$$

Definition 3. *A feasible allocation Π_i is existed, if the total allocation in each server is not greater than the total capacity of resources as:*

$$\sum_i^n \Pi^k_{i,s} \leq C^{k_{max}}_s, \forall i \in U \tag{4}$$

$$\sum_i^n r^k_{i,l} \leq C^{k_{max}}_s, \forall i \in U \tag{5}$$

Definition 4. *The allocation Π is non-wasteful, if any user i schedules at least one task in each particular server s, so that:*

$$\sum_{i=1}^n \Psi_i(U_{i,s}) \leq C_k, \forall r_{i,r}, i \in U. \tag{6}$$

Accordingly, for any allocation $\Pi_{i,s}$, and $\Pi^*_{i,s}$, the allocation meets non-wasteful if the following condition is satisfied:

$$\Psi_i(\Pi^*_{i,s}) < \Psi_i(\Pi_{i,s}) \tag{7}$$

IV. H-FFMRA

In this section, a fully fair allocation mechanism called H-FFMRA is proposed. H-FFMRA is a generalization of FFMRA in multiple servers. It captures both dominant and non-dominant resources with respect to each server and performs β-fairness [11] to evaluate how resources are evenly distributed among users across multiple servers. A two-level hierarchy approach in FFMRA is also applied to each server. H-FFMRA guarantees that users receive an equal share of resources depending on their demands.

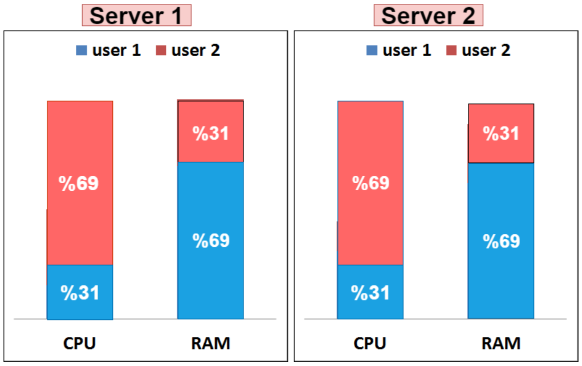


Fig. 3: The resource sharing under H-FFMRA mechanism, indicating how data center resources are distributed fairly among tasks in different servers.

A. Problem formulation

The allocation under H-FFRMA is initiated by defining the Global Aggregate Resource (GAR), considering both dominant and non-dominant based on (1) and (2). Fig. 3 illustrates how resources are allocated fully fair among users across multiple servers. As can been seen, CPU and RAM are dominant in both servers 1 and 2, while different for users 1 and 2. Therefore, H-FFMRA tries to balance the allocation of resources based on dominant and non-dominant shares based on GAR. Generally speaking, GAR aims to sum up all dominant and non-dominant resources of all servers as follows:

$$GAR(d^k_{i,s}) = \sum_{i=1}^n d^k_{i,s}, \forall r^k_{i,s}; r^k_{i,s} \in D \tag{8}$$

$$GAR(\tilde{d}^k_{i,s}) = \sum_{i=1}^n \tilde{d}^k_{i,s}, \forall r^k_{i,s}; r^k_{i,s} \in \tilde{D} \tag{9}$$

1) H-FFMRA allocation: According to Algorithm 1, in the first step of the hierarchy, the proportion of the total resource capacity of all servers which is indicated by $\rho(d^k_{i,T_s})$, and $\rho(\tilde{d}^k_{i,T_s})$ is calculated for dominant and non-dominant resources across all servers. (T_s stands for total servers) is determined for all users, sharing the entire resource pool with dominant, and non-dominant resources. Secondly, the proportions of $\rho(d^k_{i,T_s})$, and $\rho(\tilde{d}^k_{i,T_s})$ are specified for each server with respect to dominant, and non-dominant resources, represented by φ_{d_s}, and $\varphi_{\tilde{d}_s}$, respectively. If $U^k_{d^s}$, and $U^k_{\tilde{d}^s}$

denote a group of users with specific dominant, and non-dominant resources in each server sequentially. Then, the proportion of φ_d, and $\varphi_{\tilde{d}}$ are used to calculate final shares, employing MLF-DRS [14] based on the following optimization problem.

$$\begin{aligned} \text{maximize} \quad & (r_{1,s}, r_{2,s}, ..., r_{n,s}) \\ \text{subject to} \quad & \varphi_{d,s} \leq \rho_{d,s} \\ & \varphi_{\tilde{d},s} \leq \rho_{\tilde{d},s} \\ & \rho(U^k_{d,s}) \leq \varphi_{d,s} \\ & \rho(U^k_{\tilde{d},s}) \leq \varphi_{\tilde{d},s} \\ & \sum d^k_{i,s} + \sum \tilde{d}^k_{i,s} \leq C^k_s \end{aligned} \tag{10}$$

Definition 5. *An allocation under H-FFMRA is limited to only requested resources if any resource shortage happens in the system as the fewer amount of resources are provisioned to all servers. Consequently, the allocation can be determined as follows:*

$$\Pi(d^k_{i,s}) = r^k_{i,s}, \Pi(\tilde{d}^k_{i,s}) = r^k_{i,s} \tag{11}$$

Definition 6. *An allocation satisfies H-FFMRA, if for all servers*$(s \in S)$*, there is at least one user with submissions in which* $(r^k_{i,s}) > 0$*. Furthermore, H-FFMRA is feasible if any allocation* Π *corresponding to user* i *cannot be maximised without decreasing the allocation for any user* j*. i.e,* $\Pi_{i,s} \geq \Pi_{j,s}$.

V. Fairness

H-FFMRA is the generalization of FFMRA in multiple servers. It captures the dominant, and non-dominant resources concerning each server, and performs β-fairness to evaluate how resources are evenly distributed among users over multiple servers. The formulations in (12) and (13) indicate the difference between the allocated resources to dominant shares with respect to the capacity of each server. Accordingly, the value of β falls in a range (0,1). Hence, if it tends to 0, the allocation is fully fair.

$$\beta_d = \left| \left(\sum (\frac{\Pi_i(d^k_{i,1})}{C^k_s}) \right) - ... - \left(\sum (\frac{\Pi_i(d^k_{i,s})}{C^k_s}) \right) \right| \in (0,1) \tag{12}$$

$$\beta_{\tilde{d}} = \left| \left(\sum (\frac{\Pi_i(\tilde{d}^k_{i,1})}{C^k_s}) \right) - ... - \left(\sum (\frac{\Pi_i(\tilde{d}^k_{i,s})}{C^k_s}) \right) \right| \in (0,1) \tag{13}$$

The formulations (12), and (13) indicate that a group of users in each server receive an equal proportion of the resource pool. To measure resource allocation with fairness with regards to each user in each server, the *Jain's index* is applied for each server based on (14), and (15) for dominant and non-dominant shares.

$$J(d^{k_1}_{i,s}, ..., d^{k_m}_{i,s}) = \frac{(\sum d^{k_m}_{i,s})^2}{\Psi \sum {d^{k_m}_{i,s}}^2} \tag{14}$$

$$J(\tilde{d}^{k_1}_{i,s}, ..., \tilde{d}^{k_m}_{i,s}) = \frac{(\sum \tilde{d}^{k_m}_{i,s})^2}{\Psi \sum {\tilde{d}^{k_m}_{i,s}}^2} \tag{15}$$

Algorithm 1 FFMRA allocatin algorithm

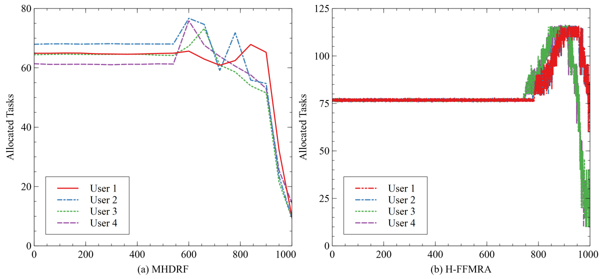
```
$K \leftarrow (1, 2, ..., k)$    ▷ Resource vector
$S \leftarrow (s_1, s_2, ..., s_m)$    ▷ The vector contains all servers
$U \leftarrow (1, 2, ..., n)$    ▷ total users in the system
$D \leftarrow (r^k_{1,s}, ..., r^k_{i,s})$    ▷ demand vector
$d^k_{i,s}, \tilde{d}^k_{i,s}$    ▷ Dominant, and non-dominant resources of each user in each server
$T_{c,s} \leftarrow \sum C^k_s$    ▷ Sum of total resources in entire system
$\sigma_d \leftarrow \sum d^k_{i,s}$    ▷ Sum of all dominant resources
$\sigma_{\tilde{d}} \leftarrow \sum \tilde{d}^k_{i,s}$    ▷ Sum of all non-dominant resources
$\rho_{d,s} \leftarrow (T_{c,s} * \sigma_d)/(\sigma_d + \sigma_{\tilde{d}})$ ▷ The proportion of resource pool for all dominant, and non-dominant resources
$\rho_{\tilde{d},s} \leftarrow (T_{c,s} * \sigma_{\tilde{d}})/(\sigma_d + \sigma_{\tilde{d}})$ ▷ The proportion of resource pool for all dominant, and non-dominant resources
for each s in S do
    $\varphi_{d,s} \leftarrow (\rho_{d,s} * C^k_s)/\sigma C^k_s$    ▷ The Proportion for each server in terms of dominant resources
    $\varphi_{\tilde{d},s} \leftarrow (\rho_{\tilde{d},s} * C^k_s)/\sigma C^k_s$    ▷ The Proportion for each server in terms of non-dominant resources
for each k in s do
    $\rho(G_{d,s}) \leftarrow (\varphi_{d,s} * C^k_s)/\sum C^k_s$    ▷ The proportion for each group of users in each server, dominated in each specific resource type
    $\rho(G_{\tilde{d},s}) \leftarrow (\varphi_{\tilde{d},s} * C^k_s)/\sum C^k_s$    ▷ The proportion for each group of users in each server with regards to non-dominant resources
for i in $G_{d,s}$ do
    Apply MLF-DRS
for i in $G_{\tilde{d},s}$ do
    Apply MLF-DRS
```

VI. Evaluations

In this section, we evaluate the performance of H-FFMRA, considering resource allocation, utilization, and fairness. The experiments are carried out using the CloudSim as a simulation framework, driven by randomly generated workloads. For simplicity, we use two types of resources, CPU and RAM. Servers configurations are arbitrarily selected to measure the actual functionality of H-FFMRA. Under random workload generation, users may represent various submissions based on dominant and non-dominant resources that could be varied depending on demands.

A. Resource allocation

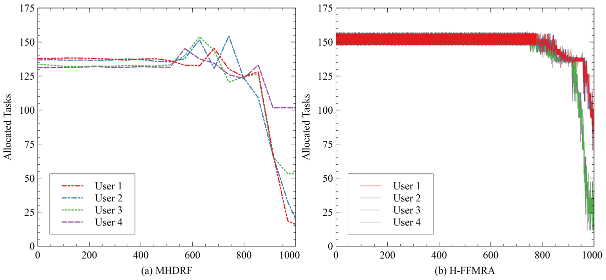
The resource allocation under H-FFMRA is measured taking into account four users and 90 servers. All users are eligible to submit their tasks in all servers in absence of any placement constraint. In each iteration, users submit a variety of tasks, dominated either on CPU or memory. Moreover, the allocation is examined over 1000 iterations. Fig. 4 and 5, compare total allocated tasks in Multi-Host DRF and H-FFMRA for four users, taking into account CPU and RAM. As can be seen in the figures, under H-FFMRA all users approximately get a total equal allocated task as all resources are distributed fairly among groups of users in each server. Surprisingly, according to lines 15, and 16 in Algorithm 1, an absolute correlation is maintained for both dominant and non-dominant resources in the entire system, as it also affects allocated tasks for all users. Fig 4(a) is the best representation of CPU allocation under H-FFMRA compared to Multi-Host DRF as all users in H-FFMRA get more resources than Multi-Host DRF. Furthermore, by iteration 800, all users get nearly the same allocation under H-FFMRA. Nonetheless, after this point, as the dominant resources of submitted tasks are greater than the *fair-share* for some users, the allocation is slightly different. Therefore, some users get more resources than others. Typically, H-FFMRA attempts to capture an equal allocation for all users under such a condition where dominant resources of all users are less than the *fair-share*. Fig. 5(b) also illustrates RAM allocation in both approaches. The allocation under HFFMRA is significantly higher than Multi-Host DRF. After iteration 700, the allocated tasks for users in both algorithms start to go down. This is because of the happening resource shortage in the system as at this point no more resources are provisioned to servers. Accordingly, both algorithms try to allocate resources to users at least based on users' requested resources.



(a) MHDRF

(b) H-FFMRA

Fig. 4: Allocated CPU to all users



(a) MHDRF

(b) H-FFMRA

Fig. 5: Allocated RAM to all users

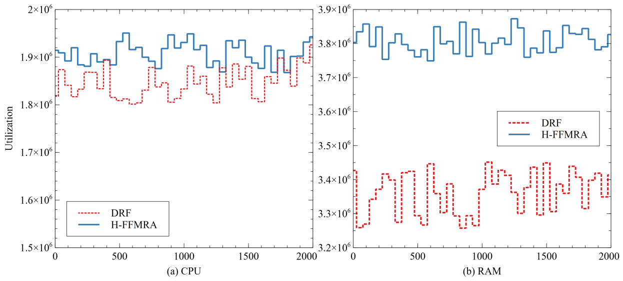
B. Resource utilization

In this section, we compare resource utilization under H-FFMRA, and Multi-Host DRF policies, taking into account 2000 servers, and 200 users. The configuration of servers is randomly selected so that the total capacities of CPU and RAM in the entire system are: 2×10^6, and 4×10^6, respectively. Then, the average resource utilization is determined in all servers in each iteration based on the following equation:

$$U_{s,k}(t) = \sum_{s=1}^{m} \sum_{i=1}^{n} \Pi_{i,s}^{k} / \Psi_s, s \in S \qquad (16)$$

Where Ψ_s denotes the total number of servers in the system. The average utilization is a time-dependent function that captures the utilization in a specific range of iterations. So, if we assume time series in a range (t_0, t_{h+1}), then $U_{s,k}(0)$ gives utilization in average at iteration 0.

Fig. 6 represents the resource utilization in both approaches in a time series experiment over 2000 iterations. Fig. 6(a) depicts CPU utilization. As can be seen in the figure, H-FFMRA shows better CPU utilization than DRF in a multi-host setting. This is clear in allocated tasks to users in Fig. 4 and Fig. 5, as all users under H-FFMRA schedule significantly more tasks than DRF. Concerning RAM utilization, the superiority of H-FFMRA is seen under H-FFMRA in Fig. 6(b). Typically, the original DRF algorithm fails to satisfy the full utilization of resources, at least on a specific resource type in the heterogeneous resource profile. Accordingly, the DRF in a multi-servers perspective is unable to capture the full utilization of resources. It is worth mentioning that under multi-host DRF, the CPU utilization is near to optimum, however, other resource types such as RAM and disk may not have been fully utilized. Therefore, Fig.6(b) shows that H-FFMRA performs extremely better than DRF in consuming system resources. Except for resource shortage in some iterations, H-FFMRA obtains approximately full utilization of resources.



(a) CPU

(b) RAM

Fig. 6: resource utilization in H-FFMRA, and MHDRF

C. Fairness

In this section, the performance of H-FFMRA is analyzed, considering fairness in servers and user levels. For the first set of experiments, the fairness is measured using the $\beta - fairness$ index, then the $Jain's$ index is applied to evaluate the fairness for all users in the entire system. The experiments are conducted using time series over 1000 iterations, and 2000 servers so that the fairness becomes meaningful. Similar to the previous experiments, 200 users are considered with various task submissions. Then, the average fairness in each iteration is reported, perceived by users in both hierarchical levels in FFMRA.

1) $\beta - fairness$: Fig. 7 illustrates the fair distribution of CPU and RAM under H-FFMRA among all servers as well as groups of users with dominant resources. As an example, in a range of iterations $(20, 140)$, when 0.59 of RAM is considered on average to all servers with a set of dominant resources, the same amount for CPU is considered as well. The figure clearly states that a fully fair distribution is achieved in almost all iterations. Nonetheless, in some points, the distribution is not the same for CPU and RAM in some specific ranges such as $170 - 270$, $600 - 830$, and $840 - 900$. However as there is a small deviation between $(0.62, 0.64)$, it is trivial in such a large-scale scenario. Overall, H-FFMRA achieves a fully fair distribution of resources among servers.

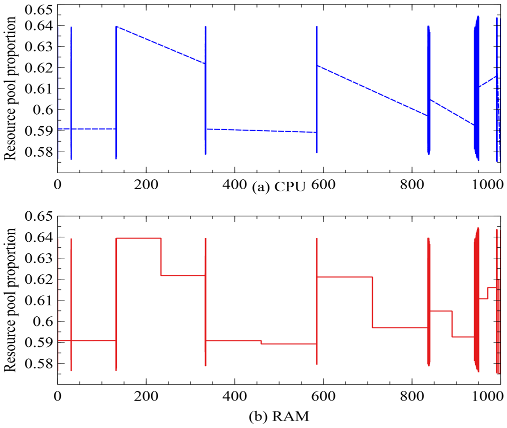


Fig. 7: The fair distribution of resources among servers, considering dominant resources

In Fig. 8(a)(b), the equal distribution of resources among servers is examined, considering non-dominant shares, the figures strongly confirm that the fair distribution is well-achieved in H-FFMRA. Therefore, the values for β are exactly the same for both types of resources as the difference between the CPU and RAM shares are about 0 in each iteration according to (11) and (12). Therefore, according to the results in this section, we capture the fairness in the first hierarchy of H-FFMRA. In the next section, we will go through the second hierarchy by evaluating fairness using $Jain'sindex$.

Given the results in Fig. 7, and Fig. 8, the beta fairness for dominant and non-dominant resources is illustrated in Fig. 9. Accordingly, it is the difference between the proportional allocation for groups of users with dominant and non-dominant shares for CPU and RAM.

As can be seen in the figure, the $\beta - fairness$ is satisfied for dominant and non-dominant shares. Although there is a small variation for dominant shares in some iterations, it is trivial to consider the difference in the range $(0, 1)$. For example, if we consider the time series in a range $(150, 300)$, the difference is $0.64 - 0.62 = 0.02$ which is a very small value.

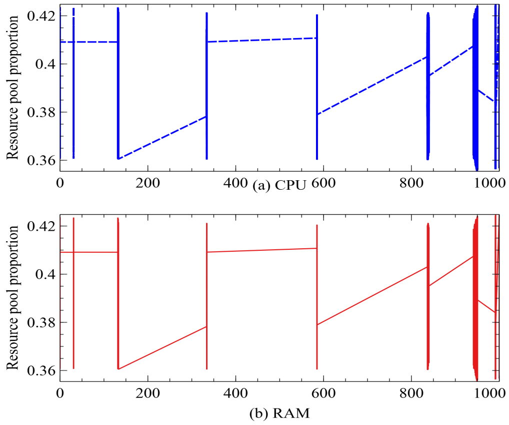


Fig. 8: The fair distribution of resources among servers, considering non-dominant resources

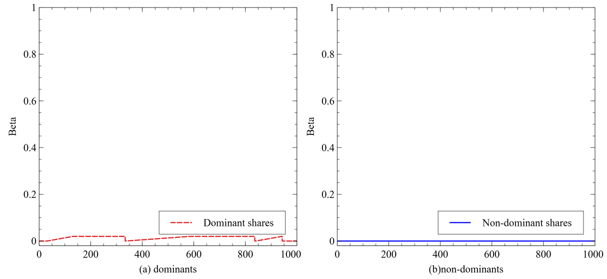


Fig. 9: Beta fairness for dominant, and non-dominant shares

Consequently, the results confirm that H-FFMRA satisfies a fully fair distribution of resources in the first level of hierarchy. In the next section, the fairness will be evaluated by applying $Jain'sindex$ to see how H-FFRMA behaves fairly among users with regards to each specific server.

2) Jain's fairness index: In the second hierarchy of H-FFMRA, the fair allocation is examined with respect to each server, and for each user in a specific group, considering dominant and non-dominant shares. In this stage, and for evaluating fairness, a series of experiments are conducted, taking into account 400 users and 3000 servers over 1000 iterations. Fig. 8, depicts fair CPU and RAM allocations to users, respecting dominant resources under H-FFMRA and multi-host DRF. As the requested resources by users with tasks dominated on CPU and RAM could be greater than the fair-share, the value for the Jain could be oscillated, and far away from 1. Correspondingly, based on the randomly generated workloads, in some cases, the fairness is not the same for users with dominant resources. Due to this, as can be seen in Fig. 10(a), the fairness in CPU goes beyond 1. In terms of RAM, according to Fig. 10(b), the fairness index oscillates between 0.95, and 0.96 as the requested RAM by users is greater than the fair-share in a certain time series. On the other hand, in some iterations, Jain's index is 1 which denotes that almost all requested resources are less than the fair share.

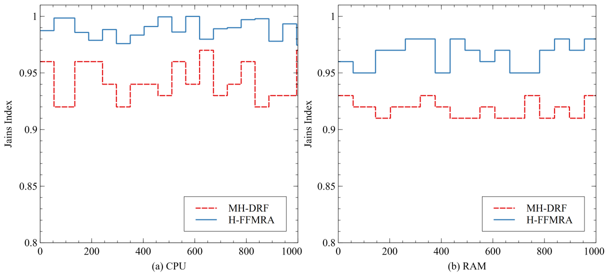


Fig. 10: The Jain's index for users with dominant resources

In addition to dominant shares, we also measure Jain's index for non-dominants. According to Fig. 11, H-FFMRA performs remarkably fairer than DRF in multiple hosts, as it equalizes only dominant resources. Consequently, under DRF, non-dominant resources do not receive an equal share of the resource pool. Hence, as can be seen in the figure, the fairness index for H-FFMRA is near to the optimal value of 1 for CPU, and RAM compared to DRF. In a certain range of time series, the value of Jain's index is approximately 0.95. This is due to that in some cases the dominated demand either on CPU and RAM is greater or less than the *fair-share*. Therefore, similar to dominant resources, the allocation could not be the same for all users. Overall, the results verify that H-FFMRA is a fully fair allocation mechanism. This is worth mentioning that by increasing resource capacities for all servers in a standard condition, and without having resource shortage, the chance of reaching a fully fair allocation is considerably high.

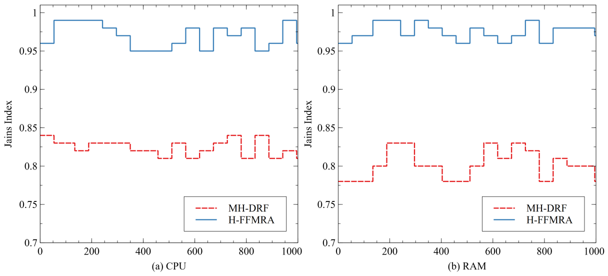


Fig. 11: The Jain's index for users with non-dominant resources

VII. Conclusion and Future Work

In this paper, we propose H-FFMRA as a multi-resource, multi-server fair allocation mechanism in cloud computing. We determine Global Aggregate Resource(GAR), respecting each server to establish an equal distribution of resources across all servers. We systematically analyze the functionality of H-FFMRA against DRF in a multi-server setting through experiments conducted in the CloudSim and driven by randomly generated workloads. The results from experiments show that H-FFMRA captures intuitive fairness, fully fair resource distribution across all servers and users as well as full utilization of resources. Furthermore, a new fairness indicator called β is introduced which indicates how resources in the entire resource pool are fairly distributed among servers. H-FFMRA is expected to be applicable in a wide range of application domains such as IoT, edge computing, and cloud-native solutions like Kubernetes. In future work, we will investigate the implementation of H-FFMRA in these areas by leveraging machine learning, and autonomic computing techniques to make it adaptable to these environments.

References

[1] F. L. Pires and B. Barán, *Cloud computing resource allocation taxonomies*, International Journal of Cloud Computing.

[2] W. Wang, B. Li, and B. Liang, *Dominant resource fairness in cloud computing systems with heterogeneous servers*, IEEE INFOCOM 2014 - IEEE Conference on Computer Communications, 2014.

[3] J. V. Gautam, H. B. Prajapati, V. K. Dabhi and S. Chaudhary, "A survey on job scheduling algorithms in Big data processing," 2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, 2015, pp. 1-11.

[4] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, *"Dominant resource fairness: Fair allocation of multiple resource types.*inProc. USENIX NSDI , vol. 11, 2011, pp. 24–24.

[5] D. Nace and M. Pioro, "Max-min fairness and its applications to routing and load-balancing in communication networks: a tutorial," in IEEE Communications Surveys Tutorials, vol. 10, no. 4, pp. 5-17, Fourth Quarter 2008.

[6] M. A. Mollah, X. Yuan, S. Pakin and M. Lang, "Rapid Calculation of Max-Min Fair Rates for Multi-Commodity Flows in Fat-Tree Networks," in IEEE Transactions on Parallel and Distributed Systems, vol. 29, no. 1, pp. 156-168, 1 Jan. 2018.

[7] W. Wang, B. Li, B. Liang and J. Li, "Multi-resource Fair Sharing for Datacenter Jobs with Placement Constraints," SC '16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, Salt Lake City, UT, 2016, pp. 1003-1014.

[8] J. Khamse-Ashari, I. Lambadaris, G. Kesidis, B. Urgaonkar and Y. Zhao, "An Efficient and Fair Multi-Resource Allocation Mechanism for Heterogeneous Servers," in IEEE Transactions on Parallel and Distributed Systems, vol. 29, no. 12, pp. 2686-2699, 1 Dec. 2018.

[9] W. Wang, C. Feng, B. Li, and B. Liang. "On the Fairness-Efficiency Tradeoff for Packet Processing with Multiple Resources". In Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies (CoNEXT '14). ACM, New York, NY, USA, 235-248.

[10] P. Poullie, T. Bocek and B. Stiller, *"A Survey of the State-of-the-Art in Fair Multi-Resource Allocations for Data Centers*," in IEEE Transactions on Network and Service Management, vol. 15, no. 1, pp. 169-183, March 2018.

[11] H. Hamzeh, S. Meacham, K. Khan, K. Phalp, and A. Stefanidis. FFMRA: *"Fully Fair Multi-Resource Allocation Algorithm in Cloud Environments"*, 2019.

[12] C. Joe-Wong, S. Sen, T. Lan and M. Chiang, "Multi-resource allocation: Fairness-efficiency tradeoffs in a unifying framework," 2012 Proceedings IEEE INFOCOM, Orlando, FL, 2012, pp. 1206-1214.

[13] Y. Tahir, S. Yang, A. Koliousis and J. McCann, "UDRF: Multi-Resource Fairness for Complex Jobs with Placement Constraints," 2015 IEEE Global Communications Conference (GLOBECOM), San Diego, CA, 2015, pp. 1-7.

[14] H. Hamzeh, S. Meacham, B. Virginas, K. Khan and K. Phalp, "MLF-DRS: A Multi-level Fair Resource Allocation Algorithm in Heterogeneous Cloud Computing Systems," 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), 2019.