ORIGINAL RESEARCH **Open access**

Efficient Resource Allocation and Application Placement for UAV-Mounted Edge Computing in IoFT

David Osei[1*], Akua Afriyie[1], Kofi Adu[2], Adwoa Mensah[3]

Abstract

Unmanned Aerial Vehicle (UAV)-mounted edge computing (UMEC) offers a cost-effective, highly mobile solution for providing computation offloading to Internet of Forestry Things (IoFT) applications in forested environments that suffer from infrastructure deficits. Consolidating multiple IoFT applications onto fewer UAV-mounted servers can enhance resource efficiency and reduce deployment costs, provided all applications' Quality of Service (QoS) criteria are met. Nonetheless, prevailing application placement approaches in UMEC typically overlook the variability inherent in aggregated computing resource needs. This work examines resource allocation and application placement challenges in UMEC through the lens of fine-grained cooperative overbooking. First, a Two-tenant Cooperative Resource Overbooking (2CROB) method is developed for the two-tenant overbooking scenario, allowing tenants to absorb one another's resource demand violations (RDVs) within a shared cooperative overbooking area. Within 2CROB, a problem of minimizing aggregated resource demand is structured and solved by a tailored bisection search algorithm. Second, for multi-tenant overbooking scenarios, a Proportional Fairness-based Cooperative Resource Overbooking (PF-MCROB) scheme is developed, along with a bisection search algorithm to pinpoint the minimum aggregate demand. Subsequently, building on PF-MCROB, a First-Fit Decreasing-based Cooperative Application Placement (FFD-CAP) strategy is introduced to pack applications into the minimum feasible number of servers. Numerical evaluations demonstrate that the proposed cooperative overbooking mechanisms yield superior savings in computing resources when incorporating more tenants with higher or heterogeneous resource demand violation ratios (RDVRs); moreover, the FFD-ACP scheme slashes the number of deployed UAVs by roughly one-third relative to standard overbooking practices. Hence, deploying efficient cooperative overbooking for application placement can markedly lower both deployment and operational costs, elevate onboard resource utilization, and boost revenue streams in UMEC-supported IoFT deployments.

Keywords IoT, UAV, Overbooking, Application placement, Mobile edge computing

*Correspondence:
David Osei
david.osei@ug.edu.gh
[1] Department of AI for Cocoa and Cassava Disease Surveillance, Faculty of Agriculture, University of Ghana, Accra, Ghana
[2] Department of Smart Microclimatic Data Analytics for Rainfed Crops, Faculty of Agriculture, Kwame Nkrumah University of Science and Technology, Kumasi, Ghana
[3] Department of Machine Learning for Soil Carbon Monitoring, Faculty of Agriculture, University of Cape Coast, Cape Coast, Ghana

Introduction

As a paramount global ecosystem, forests play an irreplaceable role in preserving planetary ecological equilibrium, counteracting soil loss, capturing atmospheric CO2, addressing climate change, and furnishing an array of resources [1, 2]. The Forestry 4.0 concept has lately attracted considerable scholarly and practical attention [3]. By modernizing information and communication technology foundations, embedding intelligent sensing networks, and advancing

This is an open access article distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License (CC BY-NC-SA 4.0), which permits non-commercial use, distribution, reproduction, and adaptation in any medium or format, provided that appropriate credit is given to the original author(s) and source, a link to the license is provided, and any changes made are indicated. Adapted material must be distributed under the same license.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

digitization and smart capabilities across the entire forestry value chain, Forestry 4.0 aims to bolster forest productivity, sustainability, and resilience, thereby securing effective governance and the enduring development of forest assets [4, 5]. Within this framework, the Internet of Forestry Things (IoFT) and unmanned aerial vehicles (UAVs) fulfill critical functions across diverse domains, including live environmental surveillance, data harvesting and interpretation, wildfire alert and tracking, timber stock assessment, and airborne transport [6-9]. IoFT relies on a multitude of inexpensive, energy-frugal sensors or edge nodes dispersed throughout woodland expanses to capture assorted environmental metrics. However, the lack of communication infrastructure, such as cellular base stations, pronounced wireless signal degradation in forested landscapes, and finite onboard energy budgets pose substantial hurdles to delivering ubiquitous coverage, massive device connectivity, and dependable data aggregation for IoFT workloads. Consequently, the synergy between UAVs and IoFT has attracted growing interest, as versatile and economically viable UAVs can readily penetrate remote or inaccessible terrain, extend radio coverage, address service blackspots, improve energy efficiency, and prolong the battery lifespan of existing IoFT infrastructure [10, 11].

In a typical IoFT setup, distributed low-cost edge devices or sensors possess modest storage, battery reserves, and processing power, rendering local task execution unfeasible. Accordingly, raw measurement data are first amassed by these edge components before being dispatched to centralized cloud data centers for warehousing, advanced analytics, and decision support. Cloud platforms deliver extensive virtualized compute, storage, and networking resources under an elastic, usage-based cost model for IoFT workloads, easing the operational burden of deploying and sustaining varied IoFT applications [12]. Yet, major cloud facilities are frequently located in remote areas where electricity is abundant and cheap; thus, an IoFT task may traverse long-haul, congested communication paths linking edge devices to the remote data center, inducing substantial end-to-end latency [13]. Mobile edge computing (MEC) has lately surfaced as a potent paradigm for offloading computational workloads to proximate edge data centers, thereby shrinking end-to-end delay for a host of burgeoning applications with rigorous latency or deadline stipulations, including virtual/augmented reality, intelligent dwellings, smart urban systems, smart energy grids, and connected autonomous vehicles [14, 15]. Within cellular infrastructure, a base station (BS) can harvest data from mobile terminals and steer tasks toward the associated edge data center (EDC) [16]. In locales with sparse or absent infrastructure, UAV-aided MEC holds considerable appeal due to its inherent agility and low operational costs [17, 18].

Depending on the task offloading mechanism, extant UAV-aided MEC frameworks can be broadly categorized into UAV-relayed MEC (UREC) and UAV-mounted Edge Computing (UMEC). Under UREC, the UAV behaves as an aerial BS, ferrying gathered data to terrestrial MEC servers; with UMEC, the UAVs themselves host computing servers and execute tasks in-flight [19]. The UREC paradigm suffers from inefficient long-range offloading and elevated communication energy consumption due to the large spans between UAVs and ground gateways. Conversely, UMEC can trim end-to-end delay and pare communication energy. However, the compute capacity of an onboard server is constrained by the UAV's payload, dimensional constraints, and power budget, thereby limiting the number of offloadable tasks. Hence, computational tasks emanating from IoFT edge devices may be offloaded directly onto a UAV-mounted server or relayed to a ground-based server, with the offloading decision jointly factoring in service latency, energy expenditures of both UAVs and edge devices, flight trajectory optimization, task granularity, communication resource scheduling, and multi-node collaboration involving UAVs and fixed servers [17, 20].

Each IoFT application comprises numerous task requests arising from ground-level edge devices; these are processed inside virtual machines (VMs) hosted on a virtualization layer executing on the aerial or ground server [21]. A server's computational fabric is abstracted into VMs—logical enclaves for running software stacks—each configured to cater to a distinct application [22]. Hence, application placement constitutes an antecedent step before task offloading and resource distribution can occur: an application must first be sited on a server before its tasks can be directed there.

Owing to the scarcity of onboard computing assets, efficiently situating IoFT applications and provisioning services in UMEC is non-trivial. When a UAV-mounted server hosts multiple IoFT applications with time-varying resource consumption, adopting a resource overbooking policy can unlock higher utilization. Overbooking, a well-established resource governance tactic for handling volatile demand, is widely deployed across domains such as cloud data centers,

© 2022 The Author(s). | Published by Agro Intelligence Research Press

network slicing, and mobile edge computing [23-25]. Given the intrinsic dynamism of resource needs from heterogeneous services, overbooked assets include CPU cycles, memory capacity, power envelopes, and wired/wireless link bandwidth [26-28]. Broadly, resource overbooking is a policy in which service providers confirm more reservations than the capacity actually allocated to fulfil the service [29]—in simpler terms, the total resource claims of admitted tenants exceed the provider's physical resource ceiling. Across industries, overbooking can maximize resource utilization, boost provider profitability, and reduce tenant costs.

This paper marries the overbooking principle with resource allocation and application placement for UMEC-driven IoFT, devising an effective cooperative resource overbooking and application placement strategy that trims the necessary UAV fleet size while robustly safeguarding all applications' Quality of Service (QoS) constraints. The procedural flow underpinning the proposed cooperative resource overbooking and application placement methodologies appears in **Figure 1**, and the principal contributions are delineated below:

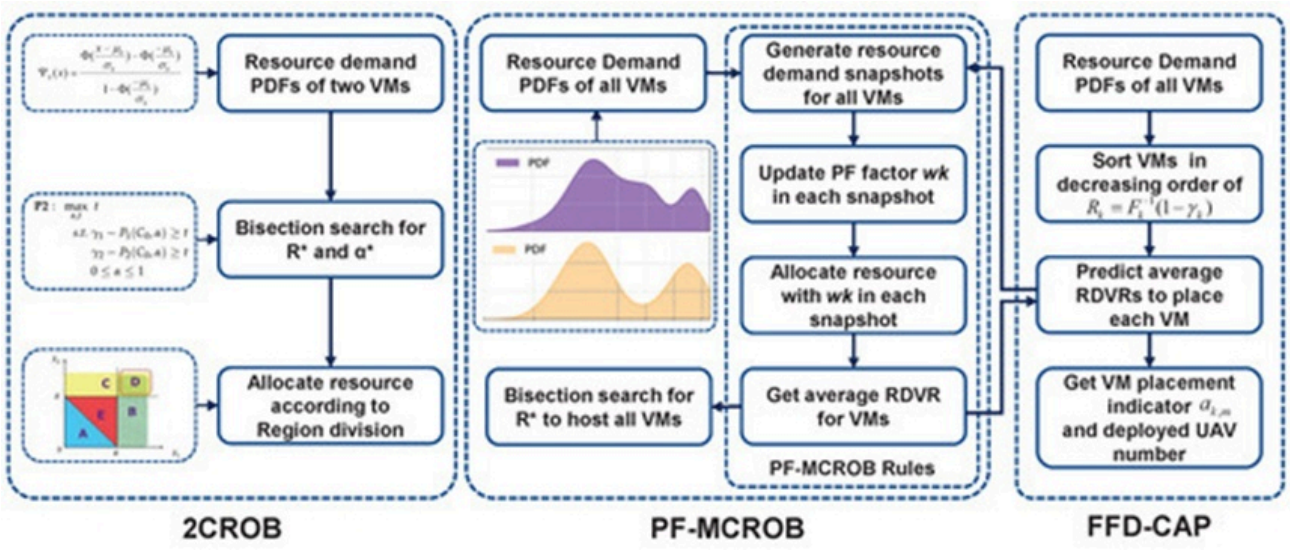

Figure 1. Flowchart of proposed methods.

• A cooperative resource overbooking architecture is introduced for the two-tenant overbooking context, accompanied by formalized resource allocation policies that enable two tenants to buffer resource demand violations jointly (RDVs) inside a cooperative overbooking zone.

• Drawing upon the Two-tenant Cooperative Resource Overbooking (2CROB) rules, a closed-form characterization of each tenant's RDVR is derived under truncated normally distributed resource demands, and a problem of minimizing the aggregate resource demand is posed subject to per-tenant RDVR constraints. This optimization is subsequently decomposed into a sequence of linear subproblems, and a bisection search procedure is engineered to determine the minimal aggregated resource demand within 2CROB.

• A Proportional Fairness-based Cooperative Resource Overbooking (PF-MCROB) framework is formulated for multi-tenant overbooking settings, complemented by a bisection search algorithm that computes the minimized aggregate resource demand.

• Leveraging PF-MCROB, a First Fit Decreasing-based Cooperative Application Placement (FFD-CAP) algorithm is conceived, which consolidates multiple VMs onto a single server solely when all VMs' RDVR guarantees remain intact under PF-MCROB stipulations.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

• Simulation evidence substantiates the 2CROB theoretical derivations and corroborates the practical effectiveness of the 2CROB, PF-MCROB, and FFD-CAP proposals.

The organizational structure of the remainder unfolds as follows. Section 2 surveys related scholarly contributions. Section 3 formulates the system model. The resource allocation mechanics and theoretical underpinnings of 2CROB are expounded in Section 4. Section 5 elaborates on the PF-MCROB and FFD-ACP schemes. Section 6 reports numerical experiments that validate the analytical results. Section 7 furnishes concluding remarks.

Related Works

Resource allocation and application placement in cellular MEC and UMEC

In the MEC paradigm, the challenge of placing applications is also known as the VM Placement (VMP) problem or the service provisioning issue, which governs how individual services—comprising server-side code and associated data—are replicated and positioned within the storage capacity of each edge cloud [30]. Numerous researchers have tackled the VMP challenge within cellular MEC environments [30-34]. Wang *et al.* [31] explored strategies for situating VMs with minimal hardware footprint for a range of mobile cloud services. They cast the VMP task as a mixed-integer linear program to meet the distinct latency requirements of varied applications while minimizing total hardware consumption. Zhao and Liu [32] focused on VM distribution across a multi-MEC-server landscape, aiming to reduce average response time in edge networks constrained by server capacity ceilings. Zhang *et al.* [33] proposed a combined edge server location and service placement model to jointly select edge server sites and map each service to a suitable edge server. Their optimization sought to maximize overall profit under a suite of constraints, including server count, BS positions, request rates and fees, and per-server storage and compute limits [33]. The contributions by Wang *et al.* [31], Zhao and Liu [32], and Zhang *et al.* [33] excluded task-level compute or communication resource assignment and instead gauged QoS through user-experienced delay drawn from queuing formulations, an approach we likewise employ. Ouyang *et al.* [34] constructed an integrated VMP and request-scheduling optimization framework that accounts for terminal mobility and enables live VM relocation across MEC servers to meet latency targets. They harnessed Lyapunov optimization to curtail the long-term user-perceived delay while respecting a migration cost budget [34]. A shared supposition in these works is that VM resource needs are fixed, a premise at odds with the reality of time-varying aggregate demand per VM. Farhadi *et al.* [30] recognized the fluctuating resource consumption of data-heavy applications. They proposed a two-timescale structure for combined service placement and request scheduling, in which services are sited across edge clouds on a coarse-scale frame. Task requests are dispatched on a fine-grained slot basis. They disentangled the task-level resource allocation aspect from the application-level VMP aspect and crafted polynomial-time approximation algorithms for each separately.

Within cellular MEC, services or VMs can be shifted or duplicated across ground-based servers, with data flowing over a well-provisioned wired backhaul. In UMEC, however, the communication throughput and onboard compute muscle are tightly squeezed by UAV battery and payload restrictions, rendering in-flight VM migration or replication impractical for a UAV mid-mission. As a result, the application placement must stay static after the UAV launches. Before departure, a meaningful amount of application-specific data—spanning databases, software libraries, executables, and trained inference models—must be staged on the onboard server, and any application updates or swaps can happen only offline once the UAV returns to its ground depot [35]. Sound application placement can boost the compute service quality delivered from the air, uphold application QoS promises, trim the UAVs fleet required, and slash operating overhead in UMEC. To date, very few efforts have specifically probed the UMEC application placement problem [36]. Qu *et al.* [37] pioneered this area by jointly tuning service placement, UAV flight path, task scheduling, and computation resource assignment to reduce the total energy drain across all ground user devices. Li *et al.* [35] enable UAVs to revisit the depot to refresh application placement, jointly optimizing flight trajectories, energy top-ups, and placement to lift the multi-UAV fleet's long-term energy efficiency.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

While a rich body of research has examined application placement in cellular MEC or UMEC, the bulk of it still centers on resource allocation through the lens of individual task offloading and placement, or else presumes that a VM's aggregate compute demand across all tasks remains invariant. In truth, the application placement and task offloading problems ought to be handled on distinct timescales, and decoupling them drastically lowers complexity, mirroring the prevalent practice [30, 35, 37]. During application placement, providers should account for the volatile nature of an application's or a VM's combined resource appetite at a macro level. In UMEC, a VM is entrusted with processing tasks from edge nodes within the same application, and the compute resource granted to each newly arrived task—measured in CPU cycles—is determined by the offloaded data size and the task's deadline. Task arrivals follow a stochastic pattern, with the arrival rate influenced by the UAV's coverage footprint and movement path [38]. Hence, the aggregated compute demand of an IoFT application or its hosting VM can be portrayed as a stochastic process. At the onset of each resource allocation period, a VM presents a virtual resource ask to the provider—i.e., the UAV server—reflecting its present needs stemming from all active tasks. The provider then allocates virtual resources to each VM, guided by the QoS thresholds of the hosted applications and the physical server's available capacity. Even though Farhadi *et al.* [30], Li *et al.* [35], and Qu *et al.* [37] incorporated time-varying resource appetites of data-intensive applications, they remain fixated on per-task QoS indicators, such as user-experienced latency. Consequently, application-level QoS gauges—such as each VM's service-level agreement (SLA) violations (SLAVs)—should be adopted during application placement.

Resource overbooking

Scholarly work on resource overbooking can be broadly sorted into two streams: oversubscription driven by revenue management, and statistical multiplexing bounded by Quality of Service (QoS) constraints. The former harks back to airline reservation systems selling seats beyond physical capacity to buffer against passenger no-shows, thereby maximizing yield [39]. Within this revenue management-focused oversubscription thread, researchers typically focus on formulating reservation, overbooking, and cancellation policies, refining the trade-off between penalties and compensation for no-shows or cancellations, and ultimately increasing revenue [25, 40]. Tenant actions around reserving and canceling are commonly modeled as stochastic processes or forecast using various prediction mechanisms [41]. Individual tenant QoS is not rigidly enforced, provided sufficient restitution is offered [42].

Under QoS-constrained statistical multiplexing, the requirement is that tenants' QoS or resource-demand violation ratio (RDVR) must remain strictly below their agreed thresholds. Illustrating with bandwidth overbooking, production data center traffic traces show that a sizable share of dynamic traffic streams are generally weakly correlated and seldom peak in unison [43]. Through multiplexing, idle bandwidth shed by one tenant can be lent to a co-resident tenant facing a momentary spike in demand [44]. Hence, combining multiple tenants' resource requirements is more efficient, while each tenant's individual QoS remains intact by capping the combined demand's violation rate. Guerin *et al.* [45] delved into bandwidth overbooking in packet-switched networks. They coined the term equivalent bandwidth for single- and multiplexed connections, denoting the bandwidth required to achieve a link's desired buffer overflow probability (BOP) [45]. At a given BOP target, the equivalent bandwidth of multiplexed connections is far below the sum of the equivalent bandwidths of all isolated connections. Urgaonkar *et al.* [39] laid out two feasibility criteria for CPU and network resource overbooking on shared platforms, revealing that the combined demand's violation probability must fall below the smallest RDVR threshold among all tenants. Wang *et al.* [46] modeled the time-varying bandwidth requirements of VMs as Gaussian random variables. They formulated the VM bandwidth overbooking problem as a stochastic bin packing (SBP) problem to reduce the server count for energy savings [46]. Zhang *et al.* [47] and Liu *et al.* [48] employed the Gaussian bandwidth overbooking principle separately for virtual network embedding within data center fabrics and for deterministic routing across stochastic time-varying networks. Sexton *et al.* [49] split network slicing resources into two groups: assured resources and auxiliary resources. Should the assured portion dip below the instantaneous demand of certain slice tenants, the slice provider attempts to reallocate auxiliary resources to those tenants while respecting a system overload probability cap. The auxiliary pool can be allotted proportionally among tenants to lift revenue. They also furnished a quantitative dissection of the two-slice scenario with uniformly distributed resource demands.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

Recently, researchers have begun to incorporate revenue management-based oversubscription into the resource allocation fold of cellular MEC, aiming to increase utilization of both computing and communication assets [25, 50]. Yet, QoS-constrained statistical multiplexing remains rare in both cellular MEC and UMEC scholarship. Despite the breadth of published work on resource booking, the bulk of it privileges provider-side gains over tenants' differentiated needs. In oversubscription rooted in revenue management, tenants' RDVR limits may be sacrificed in exchange for compensation that swells the provider's bottom line. Under QoS-constrained statistical multiplexing, the RDVRs of individual tenants are safeguarded by keeping the aggregate demand's violation rate in check. However, this prescription is overly blunt for the demands of fine-grained resource management in overbooking, because a single breach of the combined demand does not necessarily translate into every tenant suffering its own resource shortfall. Thus, meeting the divergent QoS requirements across all applications becomes the linchpin for crafting new overbooking rules governing resource allocation and application placement in UMEC settings. **Table 1** contrasts the attributes of the previously discussed resource allocation and application placement approaches.

Table 1. Resource allocation and application placement methods surveyed.

Ref.	Task scheduling	Application placement	Scenario	Allocated resources	QoS metric	Overbooking
[30]	Supported	Supported	Cellular MEC	Computing and communication resources	Request deadline	No
[31]	Not considered	Supported	Cellular MEC	Computing resources	Service latency	No
[32]	Not considered	Supported	Cellular MEC	Computing resources	Service latency	No
[33]	Not considered	Supported	Cellular MEC	Computing resources	Service latency	No
[34]	Supported	Supported	Cellular MEC	Computing and communication resources	User-perceived latency	No
[37]	Supported	Supported	UMEC	Computing and communication resources	No specific QoS metric defined	No
[35]	Supported	Supported	UMEC	Computing and communication resources	User-experienced latency	No
[50]	Supported	Not considered	Cellular MEC	Computing resources	Risk	Yes
[25]	Supported	Not considered	Cellular MEC	Computing resources	Computation time	Yes
Ours	Supported	Supported	UMEC	Computing resources	RDVR	Yes

System Model and Problem Formulation

Depicted in **Figure 2** is a canonical UMEC-enabled IoFT landscape populated by M rotary-wing UAVs. Each UAV houses an onboard server, while a dense scattering of low-cost IoFT devices blankets the forest terrain. These edge nodes harvest data on behalf of K IoFT applications, spanning contamination sensing, environmental parameter logging, wildlife

© 2022 The Author(s). | Published by Agro Intelligence Research Press

monitoring, fire surveillance, and resource inventory tracking. Given the paucity of cellular infrastructure in this zone, the UMEC network shoulders the burden of providing computational services to edge devices.

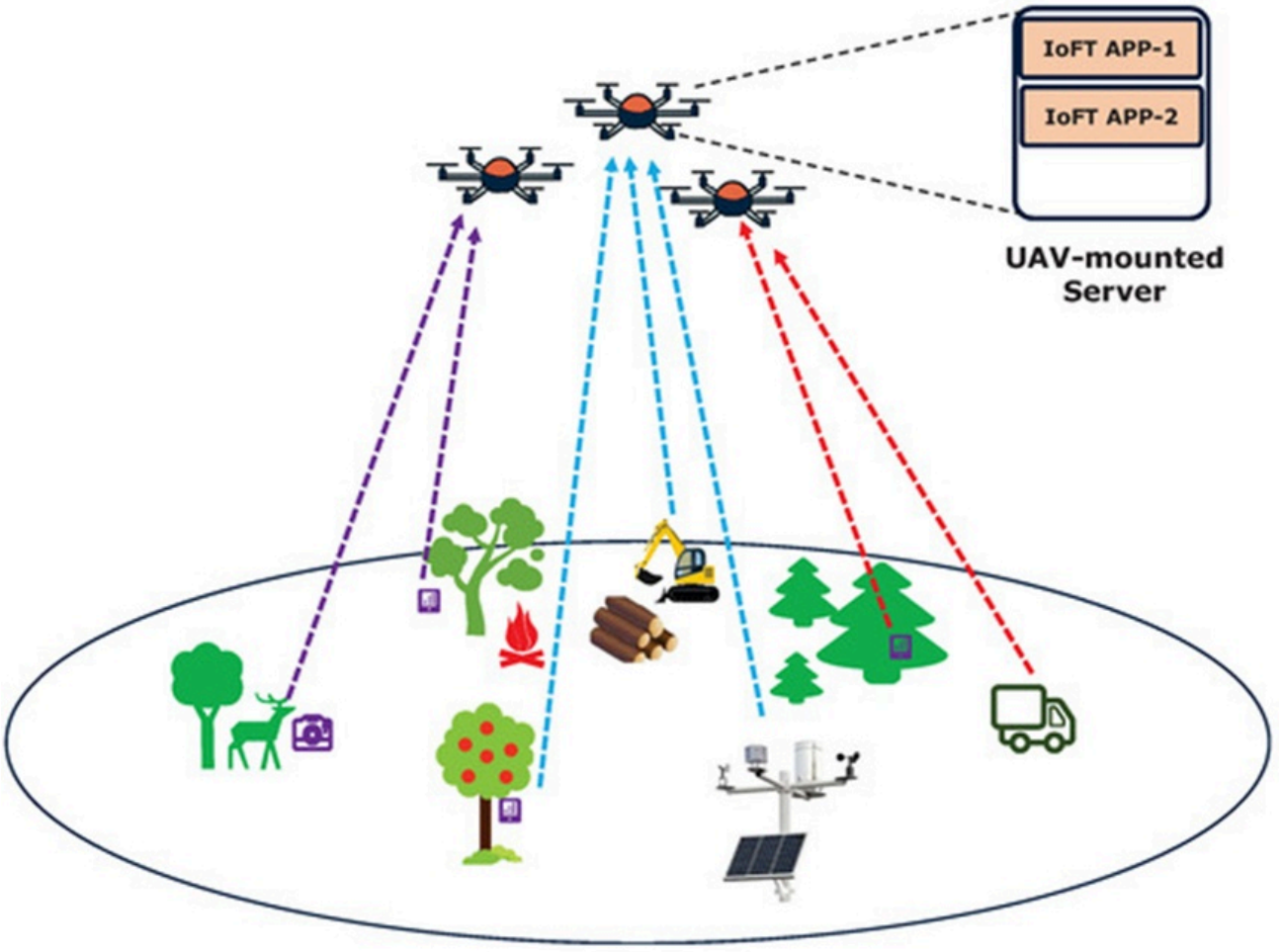

Figure 2. UAV-mounted edge computing for various IoFT applications.

Let the UAV collection be denoted by $\mathcal{U}$={$U1,U2,\ldots,UM$}. All of these UAV-mounted servers share identical computational capacity, each offering Rc computing resources. As every application resides on a server as a VM, our model uses Vk to denote both the k-th application and its corresponding VM, with the application/VM ensemble denoted by $\mathcal{V}$ = y $\mathcal{V}$= {$V1,V2,\ldots,VK$}. Each UAV-mounted server can run multiple VMs, while each VM processes exclusively the computing tasks offloaded from ground devices affiliated with a single IoFT application. This paper focuses on application/VM-level computing resource allocation, assuming that computing resources are the performance bottleneck during UMEC offloading. For simplicity, the communication segment of data offloading is omitted here; instead, we assume that each device's collected data can be uploaded to its intended UAV without delay. The migration or replication of applications/VMs among UAVs is not permitted.

The application placement indicator variables are set out as follows:

© 2022 The Author(s). | Published by Agro Intelligence Research Press

$$a_{k,m} = \begin{cases} 1, if\ V_k\ is\ placed\ in\ U_m, \\ 0, otherwise, \end{cases} \forall k, m. \quad (1)$$

Should ak,m=1 hold, VM Vk is situated on UAV Um, requiring Um to pre-load the application data of Vk ahead of mission launch. To enforce that each VM is assigned to exactly one UAV, ak, must obey:

$$\sum_{m=1}^{M} a_{k,m} = 1, \forall k. \quad (2)$$

The mission allocation indicator variable for Um is written as:

$$b_m = \begin{cases} 1, if \sum_{n=1}^{K} a_{k,m} \geq 1 \\ 0, otherwise \end{cases}, \forall m. \quad (3)$$

When bm=0, UAV Um is spared any offloading duties. Hence, the count of UAVs necessary to deliver all K IoFT application services across the forest works out to $\sum_{m=1}^{M} bm$.

The virtual computing resource overbooking construct features two stakeholders: the resource provider and the resource tenants. Within the proposed UMEC arrangement, the onboard physical server and the VMs are respectively cast as the computing resource provider and the tenants. A slot is treated as the smallest time grain for executing a full cycle of computing resource allocation. In each slot, each VM first tallies the cumulative resource demand from all its active computing tasks offloaded from ground devices and relays it to the server. The server then allocates computing resources to each VM based on incoming demand and available capacity. Finally, each VM assesses whether its resource demand ends up violated or satisfied. The resource demand quantity of VM Vk during the t-th slot is captured by (t), and the resource amount apportioned to Vk is marked as Rk,(t) when Vk is hosted on Um. Consequently, at the t-th slot, the total of resources handed out to all VMs on Um must stay within Rc, i.e.,

$$\sum_{k \in \mathcal{U}_m} R_{k,m}(t) \leq R_c, \forall m, \quad (4)$$

Where $\mathcal{U}m$ stands for the index set of VMs placed on Um, meaning $\mathcal{U}m$={k|ak,m=1}. The VM's RDVR serves as the QoS performance yardstick for each application. Accordingly, the average RDVR of VM Vk across the whole span of T slots, labeled Pk, must not exceed the stipulated cap, formulated as:

© 2022 The Author(s). | Published by Agro Intelligence Research Press

$$P_k = \frac{1}{T}\sum_{t=1}^{T} \mathcal{I}\left(x_k\left(t\right) > R_{k,m}\left(t\right)\right) \leq \gamma_k, \forall k, \quad (5)$$

Where $\mathcal{I}(y)$=1 if y proves true, and γk is the RDVR ceiling set for Vk. We further assume that (t) is tendered independently in each slot, and the resource demand of Vk can be captured by a random variable Xk with probability density function (PDF) (x) and cumulative density function (CDF) (x). It is likewise presumed that the resource demands spanning all VMs, $\{Xk,1\leq k\leq K\}$, are K independent yet non-identically distributed random variables. Moreover, the PDF and CDF of each Xk are known in advance of application placement, having been extracted, for instance, from historical data traces [24].

The attendant application placement and computing resource allocation problem in UMEC can be formalized as an optimization problem **P0**, whose objective is to minimize the number of deployed UAVs. Inside **P0**, the placement indicator variables ak, and the resource allocation variables $Rk,(t)$ for each VM act as the decision variables.

$$\mathbf{P0}: \min_{a_{k,m},\ R_{k,m}(t)} \sum_{m=1}^{M} b_m \quad (6)$$
$$s.t. \quad Constraint\ (1)-(5).$$

Problem **P0** resists direct solution owing to the split between placement indicators ak, and resource allocation variables $Rk,(t)$. On top of that, $Rk,(t)$ must be settled online, seeing that the resource demand $xk(t)$ materializes only within the t-th slot and stays unknown beforehand. In this paper, cooperative overbooking-based resource allocation and application placement schemes are architected by pursuing two elementary steps: first, work out the smallest aggregate resource amount capable of housing a given batch of VMs so that all of their RDVR properties remain intact; second, pinpoint the minimal server count needed to house all VMs.

Two-Tenant Cooperative Resource Overbooking

In this section, the resource allocation rules grounded in cooperative overbooking are laid out for the basic two-tenant overbooking scenario, followed by the associated minimal aggregate resource quantity.

Resource isolation and overbooking

If every VM commands a private resource pool and these pools are walled off from one another, overbooking is ruled out, and a fixed resource volume must be earmarked for each VM to preserve its RDVR threshold. Should VM Vk be granted a steady resource quantity Rk for the duration of T slots, **Eq. 5** can be recast as:

$$P_k = 1 - F_k\left(R_k\right) \leq \gamma_k, \forall k. \quad (7)$$

Thus, the resource volume to accommodate VM Vk must fulfill $R_k \geq F^{-1}(1-\gamma k)$. To illustrate with a two-VM placement instance, suppose $X1$ and $X2$ each obey independent Truncated Normal Distributions, designated as $Xk \sim TN(\mu_k, \sigma^2{}_k; a_k, b_k)$, whose PDFs take the form:

© 2022 The Author(s). | Published by Agro Intelligence Research Press

$$f_{X_k}(x) = \frac{f_g(x)}{F_g(b_k) - F_g(a_k)} = \frac{e^{-\frac{(x-\mu_k)^2}{2\sigma_k^2}}}{\sqrt{2\pi}\sigma_k\left[\Phi\left(\frac{b_k-\mu_k}{\sigma_k}\right) - \Phi\left(\frac{a_k-\mu_k}{\sigma_k}\right)\right]},\ a_k \le x \le b_k,\ k = 1,2. \quad (8)$$

In (8), ak and bk demarcate the truncation boundaries of Xk, $\Phi(x)$ is the standard normal distribution function, and $fg(x)$, along with $Fg(x)$, separately represent the PDF and CDF of the parent general normal distribution characterized by mean μk and variance $\sigma 2k$. For this paper, we fix ak=0 and bk=∞. The corresponding CDF of the zero-truncated normal variable Xk emerges as:

$$\Psi_k(x) = \int_0^x f_{X_k}(t)\,dt = \frac{\Phi\left(\frac{x-\mu_k}{\sigma_k}\right) - \Phi\left(-\mu_k/\sigma_k\right)}{1 - \Phi(-\mu_k/\sigma_k)},\ k = 1,2. \quad (9)$$

Using the two-VM placement case as an example, with the CDF from (9) and the parameters $\mu 1$=$\mu 2$=0, $\sigma 1$=$\sigma 2$=1, and $\gamma 1$=$\gamma 2$=0.01, the least amount of dedicated resource for each VM calculates to:

Rk=$\Psi^{-1}(1-\gamma k)$=$\sigma x\Phi^{-1}(1-\gamma_k/2)$= 2.58. So, the minimum aggregate resource volume under resource isolation for this two-VM arrangement comes to 5.16.

The essence of overbooking lies in resource sharing. Hence, several VMs can tap into a single resource pool. In a traditional overbooking approach, the collective resource demand across all VMs, Y=∑, is treated as a novel random variable, and its RDVR must sit beneath the lowest γk [39, 46], a condition written as:

$$P(Y > R^*) = 1 - F_Y(R^*) \le min\{\gamma_k, \forall k\}, \quad (10)$$

Where (x) denotes the CDF of the combined demand, and $R*$ signifies the minimum necessary resource amount. One can verify that within the very same two-VM placement case, the minimum aggregate resource volume called for is R=$F^{-1}(1-\gamma 1)$=3.97. Put differently, the conventional overbooking policy yields a 23% reduction in computing resource consumption.

Resource allocation for two-tenant cooperative overbooking

Although traditional overbooking can deliver substantial resource savings compared with non-overbooking, it focuses on the violation probability of the pooled resource demand rather than the distinct RDVR targets of individual tenants. In truth, this is a rather blunt instrument and yields imprecise resource governance. Revisiting the two-tenant example, whenever $X1$+$X2$ > R, an aggregate demand violation is registered, yet that does not mean both tenants simultaneously suffer a shortfall. The compound event $X1$+$X2$ > R can be decomposed into multiple sub-events to enable finer-grained resource distribution. If $X1$ > R while $X2$ < R, the provider can allocate sufficient resources to $V2$ and channel the leftovers to $V1$. In that situation, only the demand of $V1$ is breached. Analogously, if $X1$+$X2$ > R while both $X1$ and $X2$ lie below R, the provider retains the leeway to pick which tenant receives a full allocation and which receives a best-effort supplement. Consequently, the RDVR behaviour of each tenant depends jointly on the statistical properties of their resource demands and on the overbooking allocation rules in effect.

The precise cooperative overbooking resource allocation rules are introduced below, following the regional partitioning sketched in **Figure 3**. The two-dimensional plane spanned by the resource demands of the two tenants is carved into

© 2022 The Author(s). | Published by Agro Intelligence Research Press

several regions.

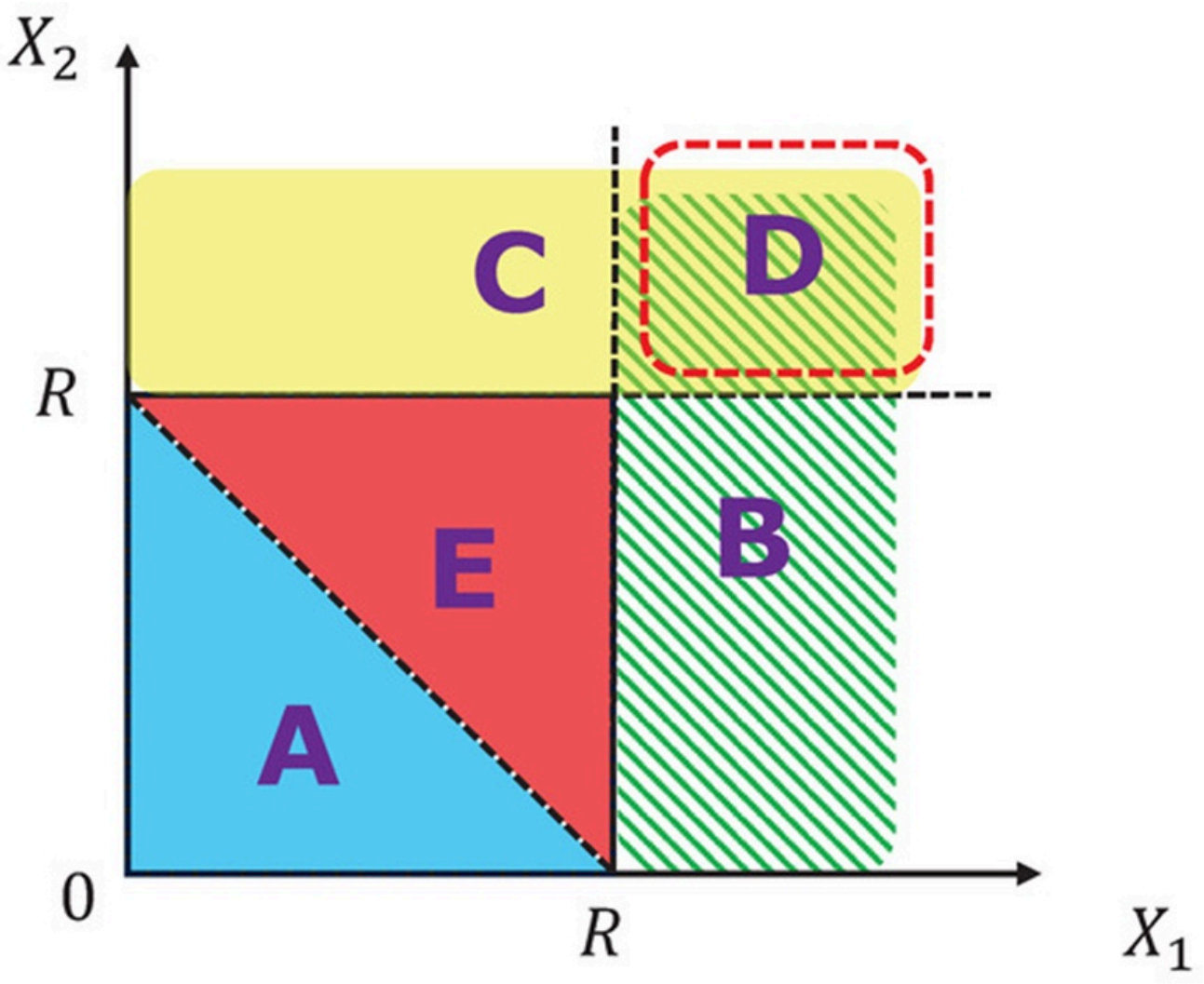

Figure 3. Region division for two-tenant cooperative overbooking.

· Region A (blue) captures the event $X1+X2 < R$, where the demands of both tenants can be met entirely.

· Region B (green diagonal lines) captures the event $X1 > R$, where $V1$ will experience an RDV.

· Region C (yellow) captures the event $X2 > R$, where $V2$ will experience an RDV.

· Region D (red box outline) is the overlap of Regions B and C, denoting simultaneous violation of the demands of $V1$ and $V2$.

· Region E (red) captures the event where $X1+X2 > C$ yet $X1$ and $X2$ are both below C. Within this zone, one tenant can be fully serviced while the other incurs an RDV. The parameter α codifies this cooperative overbooking policy: inside Region E, the demands of $V1$ and $V2$ are violated with probabilities α and $1-\alpha$, respectively. Region E is thus termed the cooperative overbooking region.

Theoretical analysis for RDVR

© 2022 The Author(s). | Published by Agro Intelligence Research Press

Under the cooperative overbooking rules, the RDVR of $V1$ is a function of R and α, and it can be written as:

$$P_1(R,\alpha) = P_B(R) + \alpha P_E(R), \quad (11)$$

Where (R) denotes the probability that $X1$ falls within Region B, and $PE(R)$ denotes the probability that both $X1$ and $X2$ land in the cooperative overbooking region. Using the PDF from (9), (R) and (R) are derived as follows.

$$P_B(R) = P(X_1 > R) = 1 - \Psi_1(R) = 1 - \frac{\Phi\left(\frac{R-\mu_1}{\sigma_1}\right) - \Phi\left(-\frac{\mu_1}{\sigma_1}\right)}{1 - \Phi\left(-\frac{\mu_1}{\sigma_1}\right)} \quad (12)$$

$$\begin{aligned} P_E(R) &= P(X_1 + X_2 > R, X_1 < R, X_1 < R) \\ &= \iint_E f_{X_1}(x) f_{X_2}(y)\, dxdy = \int_0^R f_{X_1}(x)\, dx \int_{R-x}^{V} f_{X_2}(y)\, dy \\ &= \int_0^R f_{X_1}(x) \left[\Psi_2(R) - \Psi_2(R-x)\right] dx \\ &= \Psi_1(R)\Psi_2(R) - \int_0^R \Psi_2(R-x) f_{X_1}(x)\, dx \end{aligned} \quad (13)$$

With $T = \int_0^R \Psi_2(R-x) f_{X_1}(x)\, dx$, we have

$$\begin{aligned} T &= \int_0^R \frac{\Phi\left(\frac{R-x-\mu_2}{\sigma_2}\right) - \Phi\left(-\mu_2/\sigma_2\right)}{1 - \Phi(-\mu_2/\sigma_2)} \frac{e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}}{\sqrt{2\pi}\sigma_1\left[1 - \Phi\left(-\mu_1/\sigma_1\right)\right]} dx \\ &= \frac{\int_0^R \Phi\left(\frac{V-x-\mu_2}{\sigma_2}\right) e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} dx - \Phi\left(-\mu_2/\sigma_2\right) \int_0^R e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} dx}{\sqrt{2\pi}\sigma_1\left[1 - \Phi\left(-\mu_1/\sigma_1\right)\right]\left[1 - \Phi\left(-\mu_2/\sigma_2\right)\right]} \\ &= \frac{\int_0^R \Phi\left(\frac{R-x-\mu_2}{\sigma_2}\right) \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} dx - \Phi\left(-\mu_2/\sigma_2\right)\left[\Phi\left(\frac{R-\mu_1}{\sigma_1}\right) - \Phi\left(-\mu_1/\sigma_1\right)\right]}{\left[1 - \Phi\left(-\mu_1/\sigma_1\right)\right]\left[1 - \Phi\left(-\mu_2/\sigma_2\right)\right]}. \end{aligned} \quad (14)$$

Thus, (R) can be expressed as:

$$\begin{aligned} P_E(R) &= \frac{\left[\Phi\left(\frac{R-\mu_1}{\sigma_1}\right) - \Phi\left(-\mu_1/\sigma_1\right)\right]\left[\Phi\left(\frac{R-\mu_2}{\sigma_2}\right) - \Phi\left(-\mu_2/\sigma_2\right)\right]}{\left[1 - \Phi\left(-\mu_1/\sigma_1\right)\right]\left[1 - \Phi\left(-\mu_2/\sigma_2\right)\right]} - T \\ &= \frac{\left[\Phi\left(\frac{R-\mu_1}{\sigma_1}\right) - \Phi\left(-\frac{\mu_1}{\sigma_1}\right)\right]\Phi\left(\frac{R-\mu_2}{\sigma_2}\right) - \int_0^V \Phi\left(\frac{R-x-\mu_2}{\sigma_2}\right) \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} dx}{\left[1 - \Phi\left(-\frac{\mu_1}{\sigma_1}\right)\right]\left[1 - \Phi\left(-\frac{\mu_2}{\sigma_2}\right)\right]}. \end{aligned} \quad (15)$$

For a given value of V, the integral in (15) can be evaluated numerically.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

Similarly, $P2(R,\alpha)$ can be expressed as:

$$P_2(R,\alpha) = P_C(R) + (1-\alpha)\,P_E(R), \quad (16)$$

Where:

$$P_C(R) = P(X_2 > R) = 1 - \Psi_2(R) = 1 - \frac{\Phi(\frac{\quad}{\sigma_2}) - \Phi(-\frac{\quad}{\sigma_2})}{1 - \Phi(-\frac{\mu_2}{\sigma_2})}$$

According to $1-\Phi(-x) = \Phi(x)$, $\Phi(x) = [1 + erf(\frac{x}{\sqrt{2}})]/2$, and $erf(-x) = -erf(x)$, $P_1(R,\alpha)$ (17)

And $P2(R,\alpha)$ can be separately restated as follows.

$$P_1(R,\alpha) = \frac{1 - erf(\frac{R-\mu_1}{\sqrt{2}\sigma_1})}{1 + erf(\frac{\mu_1}{\sqrt{2}\sigma_1})} + \alpha \frac{[1 + erf(\frac{R-\mu_2}{\sqrt{2}\sigma_2})]\,[erf(\frac{R-\mu_1}{\sqrt{2}\sigma_1}) + erf(\frac{\mu_1}{\sqrt{2}\sigma_1})] - \int_0^R [1 + erf(\frac{R-x-\mu_2}{\sqrt{2}\sigma_2})]\,\frac{2e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}}{\sqrt{2\pi}\sigma_1}dx}{[1 + erf(\frac{\mu_1}{\sqrt{2}\sigma_1})]\,[1 + erf(\frac{\mu_2}{\sqrt{2}\sigma_2})]} \quad (18)$$

$$P_2(R,\alpha) = \frac{1 - erf(\frac{R-\mu_2}{\sqrt{2}\sigma_2})}{1 + erf(\frac{\mu_2}{\sqrt{2}\sigma_2})} + (1-\alpha) \frac{[1 + erf(\frac{R-\mu_2}{\sqrt{2}\sigma_2})]\,[erf(\frac{R-\mu_1}{\sqrt{2}\sigma_1}) + erf(\frac{\mu_1}{\sqrt{2}\sigma_1})] - \int_0^R [1 + erf(\frac{R-x-\mu_2}{\sqrt{2}\sigma_2})]\,\frac{2e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}}{\sqrt{2\pi}\sigma_1}dx}{[1 + erf(\frac{\mu_1}{\sqrt{2}\sigma_1})]\,[1 + erf(\frac{\mu_2}{\sqrt{2}\sigma_2})]} \quad (19)$$

Two special cases merit attention.

(1) If both $\underset{\approx 0}{\Phi\left(-\frac{\mu 1}{\sigma 1}\right)}$ and $\underset{\approx 0}{\Phi\left(-\frac{\mu 2}{\sigma 2}\right)}$ hold—i.e., the probability of each tenant's resource demand falling below zero can be disregarded—the zero-truncated normal distribution can be approximated by its parent general normal distribution. Then, $\begin{matrix} erf\left(\frac{\mu 1}{\sqrt{2}\sigma 1}\right) \\ = erf\left(\frac{\mu 2}{\sqrt{2}\sigma 2}\right) \\ = 1 \end{matrix}$ holds, and $P1(R,\alpha)$ together with $P2(R,\alpha)$ simplify respectively to:

© 2022 The Author(s). | Published by Agro Intelligence Research Press

$$P_1(R,\alpha) = \frac{1 - erf(\frac{R-\mu_1}{\sqrt{2}\sigma_1})}{2} + \frac{\alpha}{4}\left\{[1 + erf(\frac{R-\mu_1}{\sqrt{2}\sigma_1})][1 + erf(\frac{R-\mu_2}{\sqrt{2}\sigma_2})] - \int_0^R [1 + erf(\frac{R-x-\mu_2}{\sqrt{2}\sigma_2})] \frac{2e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}}{\sqrt{2\pi}\sigma_1} dx\right\} \quad (20)$$

And:

$$P_2(R,\alpha) = \frac{1 - erf(\frac{R-\mu_2}{\sqrt{2}\sigma_2})}{2} + \frac{1-\alpha}{4}\left\{[1 + erf(\frac{R-\mu_1}{\sqrt{2}\sigma_1})][1 + erf(\frac{R-\mu_2}{\sqrt{2}\sigma_2})] - \int_0^R [1 + erf(\frac{R-x-\mu_2}{\sqrt{2}\sigma_2})] \frac{2e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}}{\sqrt{2\pi}\sigma_1} dx\right\}. \quad (21)$$

(2) If $\mu 1=\mu 2=0$ under the zero-truncated normal distribution, $X1$ and $X2$ become the absolute values of two independent zero-mean Gaussian variables with variances $\sigma^2{}_1$ and $\sigma^2{}_2$, respectively. Given $er(0) = 0$, $P1(R,\alpha)$ and $P2(R,\alpha)$ likewise simplify respectively to:

$$P_1(R,\alpha) = 1 - erf(\frac{R}{\sqrt{2}\sigma_1}) + \alpha\left\{erf(\frac{R}{\sqrt{2}\sigma_1})[1 + erf(\frac{R}{\sqrt{2}\sigma_2})] - \int_0^R [1 + erf(\frac{R-x}{\sqrt{2}\sigma_2})] \frac{2e^{-\frac{x^2}{2\sigma_1^2}}}{\sqrt{2\pi}\sigma_1} dx\right\} \quad (22)$$

And:

$$P_2(R,\alpha) = 1 - erf(\frac{R}{\sqrt{2}\sigma_2}) + (1-\alpha)\left\{erf(\frac{R}{\sqrt{2}\sigma_1})[1 + erf(\frac{R}{\sqrt{2}\sigma_2})] - \int_0^R [1 + erf(\frac{R-x}{\sqrt{2}\sigma_2})] \frac{2e^{-\frac{x^2}{2\sigma_1^2}}}{\sqrt{2\pi}\sigma_1} dx\right\}. \quad (23)$$

Referring once more to the two-VM placement case from Section 4.1, the expressions (22) and (23) give $P1(3.64,0.5) = P2(3.64,0.5) = 0.01$. Hence, the minimum aggregate resource volume required with 2CROB in this two-VM scenario is 3.64, yielding an additional 8.3% reduction in total resource consumption beyond traditional overbooking.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

Bisection search algorithm for optimal R_*

Under the overbooking rules, the goal is to determine the optimal α_* and the minimal aggregate resource volume R_* such that $(R_*,_*) \leq \gamma k$ is satisfied for all tenants. The resulting optimization problem **P1** can be formulated as follows.

$$\begin{aligned} \mathbf{P1}: \quad & \min_{\alpha,R} \quad R \\ & s.t. \quad P_1(R,\alpha) \leq \gamma_1 \\ & \qquad P_2(R,\alpha) \leq \gamma_2 \\ & \qquad 0 \leq \alpha \leq 1. \end{aligned} \tag{24}$$

Problem **P1** is non-convex, meaning its optimal R_* cannot be extracted by solving **P1** directly. Introduce an auxiliary variable t defined by:

$$t = min\{\gamma_1 - P_1(R,\alpha), \gamma_2 - P_2(R,\alpha)\}. \tag{25}$$

Treating $R0$ as a fixed constant, if $R0 < R_*$, the first two constraints of problem **P1** cannot be met simultaneously, resulting in $t < 0$. On the other hand, if $R0 \geq R_*$, it should be possible to find an α such that $t \geq 0$ holds. Thus, problem **P1** can be decomposed into a succession of sub-problems **P2**, each associated with a different value of $R0$; the optimal t_* obtained from each sub-problem **P2** reveals whether the supplied $R0$ falls short of or exceeds the optimal R_* of the original problem **P1**. Problem **P2** is a linear optimization program, readily solvable by tools such as CVX. A bisection search algorithm is constructed to converge toward the optimal minimum aggregate resource quantity R_* in two-tenant cooperative overbooking; during every iteration, problem **P2** is solved to narrow the search bracket. The pseudocode is presented in **Algorithm 1** below. Once the iterations terminate, the right endpoint of the concluding search interval is taken as R_*, and the optimal α_* retained corresponds to this R_*.

The iteration count of the bisection search algorithm is on the order of (log2(Rmax/Δ)). In each iteration, $P1(R0,\alpha)$ and $P2(R0,\alpha)$ are computed for the given $R0$, and the linear optimization problem **P2** is solved. The expression for $P1(R0,\alpha)$ in (18) contains a numerical integration step with complexity $\mathcal{O}(N0)$, where $N0$ is the number of discretization points spanning the integration range. The complexity of a standard linear optimization problem is $(a2b)$, where a is the number of variables and b the number of constraints. Therefore, the overall computational complexity of the bisection search algorithm for the minimum aggregate resource quantity R_* within 2CROB amounts to ($N0$log2(Rmax/Δ)).

$$\begin{aligned} \mathbf{P2}: \quad & \max_{\alpha,t} \quad t \\ & s.t. \quad \gamma_1 - P_1(R,\alpha) \geq t \\ & \qquad \gamma_2 - P_2(R,\alpha) \geq t \\ & \qquad 0 \leq \alpha \leq 1 \end{aligned} \tag{26}$$

© 2022 The Author(s). | Published by Agro Intelligence Research Press

Algorithm 1

Require: $Rmin$=0, $Rmax$, $R0$=($Rmin$+$Rmax$)/2, n=0;
Ensure: $\alpha*$, $R*$, n;

while $|Rmax-Rmin|>\Delta$ **do**
  Solve problem **P2** with $R0$ for $t*$ and $\alpha*$;
  if $t*\geq 0$ **then**
    $Rmax=R0$
  else
    $Rmin=CR0$
  end if
  $n=n+1$;
  $R0=(Rmin+Rmax)/2$;
end while
$R*=Rmax$
if $t*<0$ **then**
  Solve problem **P2** with $R*$ for $\alpha*$;
end if

Algorithm 1. Bisection Search Algorithm for Minimum Aggregated Resource Amount in 2CROB

Multiple-Tenant Cooperative Resource Overbooking and Application Placement

Proportional fairness-based resource allocation in multiple-tenant cooperative overbooking

The cooperative overbooking resource-allocation approach introduced in Section 4 recovers the minimum aggregate resource volume while preserving each tenant's RDVR guarantee in the two-tenant setting. That said, the minimum aggregate resource volume hinges on both tenants' resource demands being expressible through closed-form PDFs; failing that, the RDVR derivations and the optimization problem cannot be formulated. Moreover, the region-partitioning logic grows combinatorially with the number of tenants, rendering those allocation rules unsuitable for multi-tenant overbooking. Hence, a broadly applicable cooperative resource overbooking framework tailored to general multi-tenant cases must be designed, on top of which the application placement strategy can then be erected.

A central takeaway from Section 4 is that the spirit of cooperative resource overbooking should let tenants share the burden of resource demand violations. In contrast, each tenant's RDVR remains strictly bounded by its threshold. Thus, in multi-tenant cooperative overbooking, whenever the instantaneous total resource demand outstrips the available resource pool R, a priority ordering can be established by comparing each tenant's current RDVR against its RDVR threshold. The

© 2022 The Author(s). | Published by Agro Intelligence Research Press

tenant whose present RDVR sits furthest below its cap is assigned the highest interrupt priority, i.e., the first to be denied resources. After denying a sufficient set of high-priority tenants, the aggregate demand of the survivors can be driven below R. Within the proposed PF-MCROB scheme, the interrupt priority assigned to VM vk during the t-th slot is captured by the PF factor (t), and computing resources are distributed to each VM in the t-th slot according to the rules below.

(1) Should the sum of all VMs' resource demands fall below R, every VM receives its full requested allocation.

(2) When the total demand exceeds R, the PF factor for each VM is computed, given by:

$$w_k(t) = p_k(t)/\gamma_k, \forall k, \quad (27)$$

Where $pk(t)=NRDVk(t)/(t-1)$ represents the running average RDVR of the k-th tenant as the t-th slot begins, and $NRDVk(t)$ counts the RDV incidents accumulated by the k-th VM up to that point. Resources are preferentially granted to VMs with smaller values of (t). To implement this, VMs are ranked in descending order of (t), and the $k*$-th VM is chosen to forfeit its allocation, where:

$$k^* = \arg \min w_k(t), 1 \le k \le K. \quad (28)$$

If the remaining VMs' total demand still overshoots R, the VM with the next-smallest PF factor is suspended, and the procedure iterates until the residual aggregate demand drops below R.

Bisection search algorithm for minimum aggregated resource amount $R*$

Under the PF-MCROB scheme, for a multi-tenant cooperative overbooking case where the resource demand distributions and RDVR caps γk of all VMs are known beforehand, a minimum aggregate resource demand $R*$ exists such that every VM's average RDVR can be individually kept beneath its respective ceiling, i.e.:

$$\lim_{t\to\infty} p_k(t) \le \gamma_k, 1 \le k \le K. \quad (29)$$

Manifestly, if the allocated resource amount dips under $R*$, the average RDVR of at least one VM will overshoot its permissible limit. Note that (t) is the average RDVR estimate obtained from a sufficiently large sample. Therefore, once the resource demand distributions (x) and RDVR caps γk are specified, the provider can discern whether a candidate resource amount $R0$ is greater or smaller than $R*$. The provider may generate a large number of random resource-demand snapshots per VM via Monte Carlo sampling, execute resource allocation under the PF-MCROB rules for each snapshot, and compile the final average RDVR for each VM. If condition (29) is satisfied across all VMs, then $R0 \ge R*$; otherwise, $R0 < R*$. Anchored in this rationale, a bisection search algorithm is formulated to identify the minimum aggregate resource demand $R*$ within PF-MCROB. The corresponding pseudocode is laid out in **Algorithm 2**. For the two-VM placement example from Section 4.1, the bisection search under PF-MCROB yields a minimum aggregate resource demand of 3.64, which is practically identical to the 2CROB outcome.

The iteration count of the bisection search algorithm remains on the order of ($\log2(R\max/\Delta)$). Inside each iteration, $KN1$ resource demand samples are drawn, where K and $N1$ stand for the number of VMs and the number of snapshots, respectively. For each snapshot, a Boolean check is executed. Whenever the cumulative demand across all VMs surpasses $R0$, K PF factors must be refreshed and sorted in descending sequence, after which suitable VMs are selected

© 2022 The Author(s). | Published by Agro Intelligence Research Press

for suspension. Sorting K PF factors carries complexity ($K\log 2K$), and picking the VMs to suspend entails (K) effort. Consequently, the computational complexity of the bisection search algorithm for the minimum aggregate resource quantity $R*$ under PF-MCROB works out to ($N1K\log 2K\log 2(R\max/\Delta)$).

Algorithm 2

Require: $Rmin$=0, $Rmax$, γk,$\forall k$;
Ensure: $R*$, n;

Initialize $R0=(Rmin+Rmax)/2$, $n=0$, pe,$k=0$;
while $|Rmax-Rmin|>\Delta$ **do**
  Generate enough resource demand samples and calculate the average RDVR pk for each VM with capacity $R0$;
  If $pk\leq\gamma k$ holds for all VMs, **then**
    $Rmax=R0$
  else
    $Rmin=R0$
  end if
  $n=n+1$;
  $R0=(Rmin+Rmax)/2$;
end while
$R*=Rmax$

Algorithm 2. Bisection Search Algorithm for Minimum Aggregated Resource Amount in PF-MCROB

First fit decreasing-based cooperative application placement

Within the application placement optimization problem **P0**, the aim is to field as few UAVs as possible while hosting as many applications/VMs as feasible, all without compromising each VM's RDVR performance. Broadly, for a given VM ensemble, a smaller minimum aggregate resource demand $R*$ translates into fewer physical servers needed to house those VMs. Hence, calculating the minimum aggregate resource demand for a group of VMs and trimming the server count are fundamentally two sides of the same coin.

In the cloud computing domain, the VMP problem under dynamic resource demands is often mapped to the SBP problem. In SBP, a collection of independent random variables is supplied, and the goal is to ascertain the fewest unit-capacity bins necessary to pack all items while keeping each bin's overflow probability—the likelihood of exceeding its capacity—under a predefined constant [46]. The SBP formulation is rooted in the conventional overbooking philosophy and likewise addresses the RDVR of the aggregated demand rather than the distinct RDVR requirements of individual VMs. SBP is NP-hard and is typically addressed using heuristics such as First Fit (FF) and FFD [51]. Under FF, a VM is inserted into the earliest server that reports sufficient available capacity; under FFD, VMs are first arranged in descending order before FF is applied.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

The cornerstone of effective application placement is the ability to judge whether the RDVR requirements of all co-located VMs can be simultaneously honored when they share a server with resource capacity Rc. This determination is precisely what the proposed PF-MCROB scheme supplies. Accordingly, by building upon the classical FFD-based Stochastic Bin Packing algorithm, an FFD-CAP algorithm is devised. The core concept of FFD-CAP is to sequence VMs in descending order and assign each VM to the first server that meets its RDVR threshold. The pseudocode appears in **Algorithm 3**. To begin, all VMs are sorted in descending order of their minimum resource demands Rk = F_k^(-1) (1-γk),∀k. The algorithm then attempts to place VM vk inon the active servers, inspecting them in ascending index order; it assigns VM vk to the earliest active server that can accommodate it without violating any VM's RDVR threshold. The assessment of whether VM vk can be packed into server Un involves predicting the average RDVRs of all VMs residing in Un—now including vk—via the PF-MCROB scheme. Should no active server be able to admit vk, a fresh, empty server is brought online to host vi. The procedure concludes once every VM has been successfully placed, at which point the VM placement indicators ai, ators ai,1 ≤ i≤ K are output, with ai=m indicating that VM Vi has been packed into server Um.

During initialization, the minimum resource demands Rk = F_k^(-1)(1 - γk) are computed for all VMs. When the inverse functions F_k^(-1) (x) are unavailable in closed form, each Rk can be recovered through bisection search at a complexity of (log2(1/Δ)). Executing FFD on K constant-size VMs incurs complexity (Klog2K). Each time a VM is evaluated for placement on a server, FFD-MCROB is invoked with N1 snapshots. Therefore, the computational complexity of the FFD-CAP algorithm sums to (Klog2(1/Δ)+N1Klog2K).

© 2022 The Author(s). | Published by Agro Intelligence Research Press

Algorithm 3

Require: $Fk(x)$, $\gamma k, \forall k$, and Rc;
Ensure: VM placement indicator $ai, 1 \le i \le K$;

Initialize m=0;
Sort VMs in decreasing order of $Rk=F-1(1-\gamma k), \forall k$, and renumber the VM indexes with $R1 \ge R2 \ge \ldots \ge RK$.
for i = 1 to K **do**
  if m=0 **then**
    Activate a new server to place Vi, $m=m+1$, $ai=m$;
  else
    for j = 1 to m **do**
      Predict the average RDVRs of all VMs in server Um, including vi, by the PF-MCROB scheme.
      If RDVRs of all VMs in Um, including vi are less than their thresholds, **then**
        VM Vi is placed in server Uj, $ai=j$;
        Break
      end if
    end for
  end if
  if $m>0$ & $ai=0$ **then**
    Activate a new server to place Vi, $m=m+1$, $ai=m$;
  end if
end for

Algorithm 3. FFD-based Cooperative Application Placement Algorithm

Simulation Results

Two-tenant cooperative resource overbooking

Figures 4 and 5 display the smallest aggregate resource pools necessary to satisfy the RDVR limits of both VMs when employing non-overbooking (NonOB), traditional overbooking (TradOB), and the 2CROB strategy introduced in this work. The stochastic demands of the VMs follow two mutually independent truncated normal distributions across three illustrative two-VM configuration scenarios: Case 1 sets $\mu 1 = \mu 2 = 0$, $\sigma 1 = \sigma 2 = 1$; Case 2 uses $\mu 1 = \mu 2 = 0$, $\sigma 1 = 1$, $\sigma 2 = 3$; and Case 3 adopts $\mu 1 = \sigma 1 = 2$, $\mu 2 = \sigma 2 = 3$. The demand patterns in the first two cases correspond to zero-truncated normal laws with means fixed at zero, meaning the demand realizations effectively represent the magnitudes of draws

© 2022 The Author(s). | Published by Agro Intelligence Research Press

from zero-mean Gaussian populations. These two cases capture situations in which the VM demands are governed by matching and mismatched distributions. In the third case, the demands are drawn from non-identical general zero-truncated normal distributions possessing strictly positive means. The chosen parameter settings and RDVR bounds are intended to reflect representative two-VM co-location conditions, allowing a comprehensive dissection and side-by-side appraisal of the alternative overbooking mechanisms.

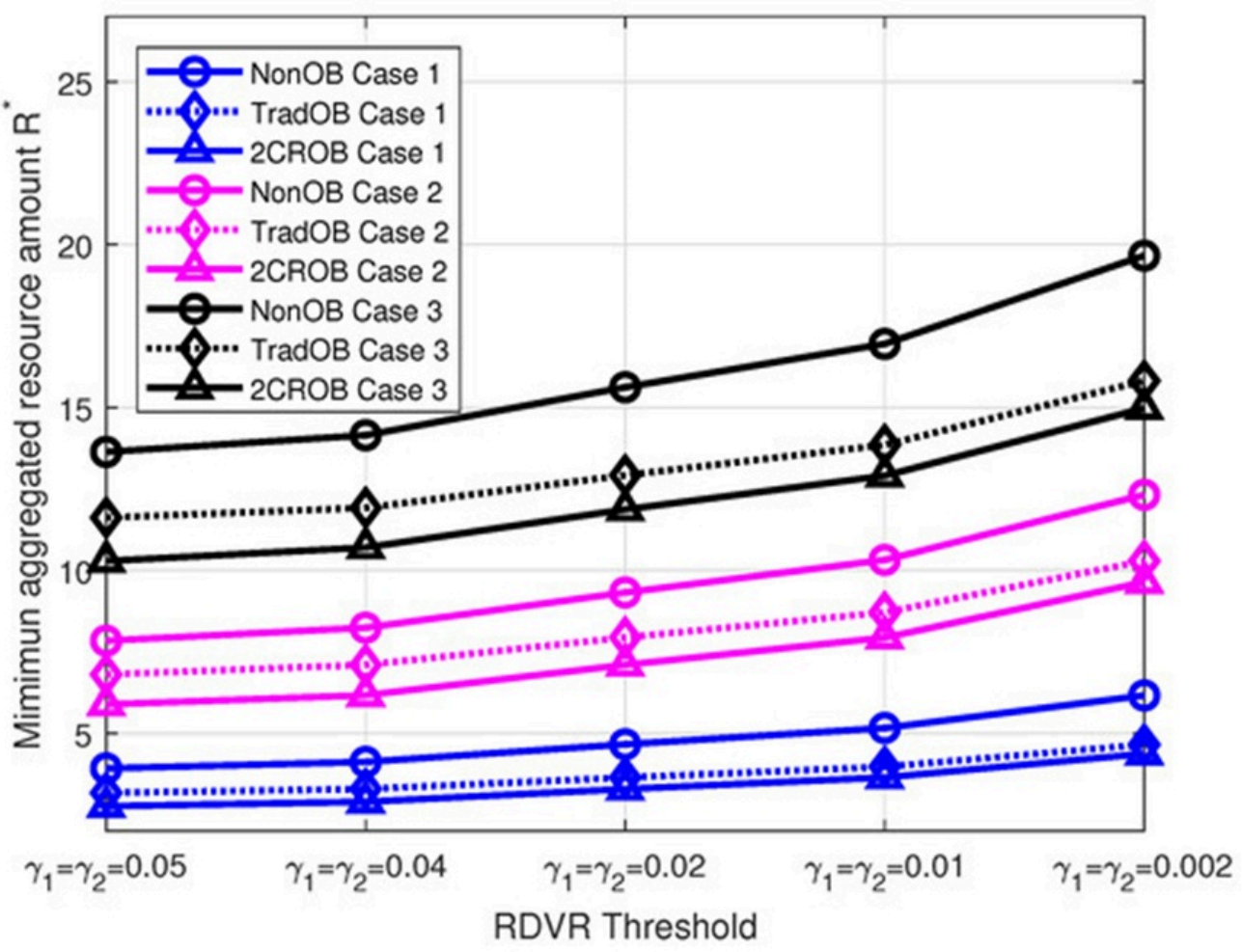

Figure 4. Minimum aggregated resource amounts of different overbooking strategies with identical RDVR thresholds.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

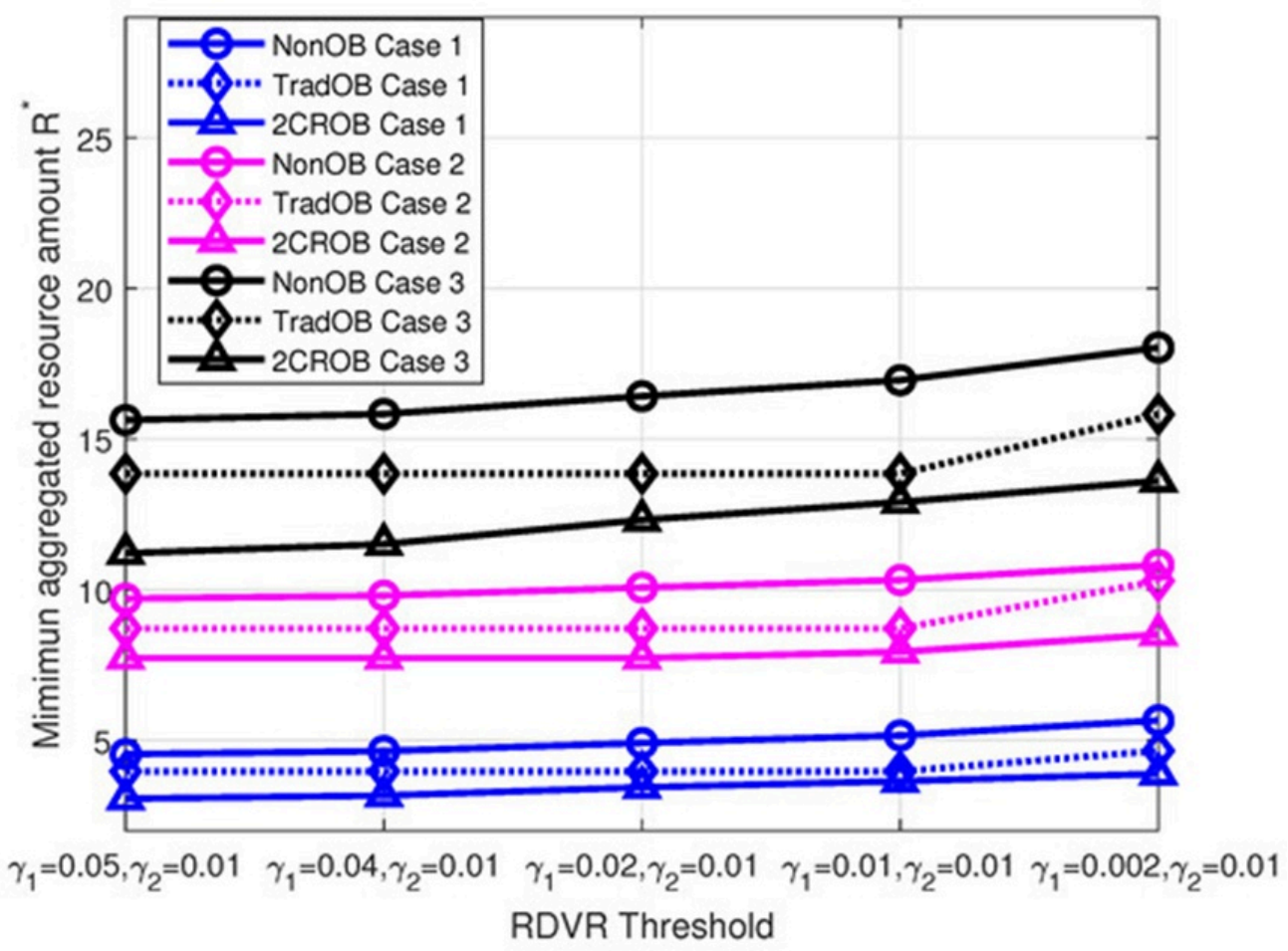

Figure 5. Minimum aggregated resource amounts of different overbooking strategies with differentiated RDVR thresholds.

The minimal aggregate resource volumes for the trio of schemes shown in **Figures 4 and 5** are derived from separate computational pathways. The 2CROB results are obtained by executing the bisection search procedure described in Section 4.4. The NonOB baseline implements strict resource partitioning, where VM Vk receives a dedicated, invariant resource block $Rk = F_k^{-1}(1 - \gamma k)$ to keep its RDVR within bounds; consequently, the NonOB aggregate figure is simply the arithmetic sum of these two static blocks. The TradOB method complies with the criterion in (10), requiring the aggregate demand's RDVR to stay beneath the more demanding of the two thresholds, i.e., $R^* = F_Y^{-1}(1 - min\{\gamma 1, \gamma 2\})$. Owing to the absence of tractable closed-form PDFs for the combined demand (x) across all three cases, the TradOB minimal volumes are likewise extracted from numerical experiments. The procedure unfolds as follows: a sufficiently rich collection of demand sample pairs, designated $X1(n)$ and $X2(n)$ for $1 \leq n \leq N$, is synthesized; these paired observations are then combined to form aggregate demand samples $Y(n) = X1(n)+X2(n)$; finally, a bisection procedure identifies the threshold R_* at which the empirical violation rate matches the tighter RDVR limit, satisfying $\sum_{n=1}^{N} \mathcal{J}(Y(n) > R^*)/N = min\{\gamma 1, \gamma 2\}$.

In every scenario plotted in **Figure 4**, the two VMs operate under an equal RDVR bound, whereas the limits diverge between the VMs for the scenarios in **Figure 5**. A clear pattern visible across both figures is that 2CROB consistently demands the smallest aggregate resource footprint, regardless of whether the underlying demand distributions or the RDVR limits are symmetric or asymmetric. The resource saving ratio (RSR), formulated as $(RTradOB - R2CROB)/RTradOB$

© 2022 The Author(s). | Published by Agro Intelligence Research Press

and computed from the data in **Figures 4 and 5**, is depicted in **Figure 6**. Broadly speaking, 2CROB extracts larger fractional savings in configurations featuring elevated and unequal RDVR caps. With 2CROB, the total resource commitment can contract by upwards of 6% under equal thresholds and by more than 11% under differentiated thresholds.

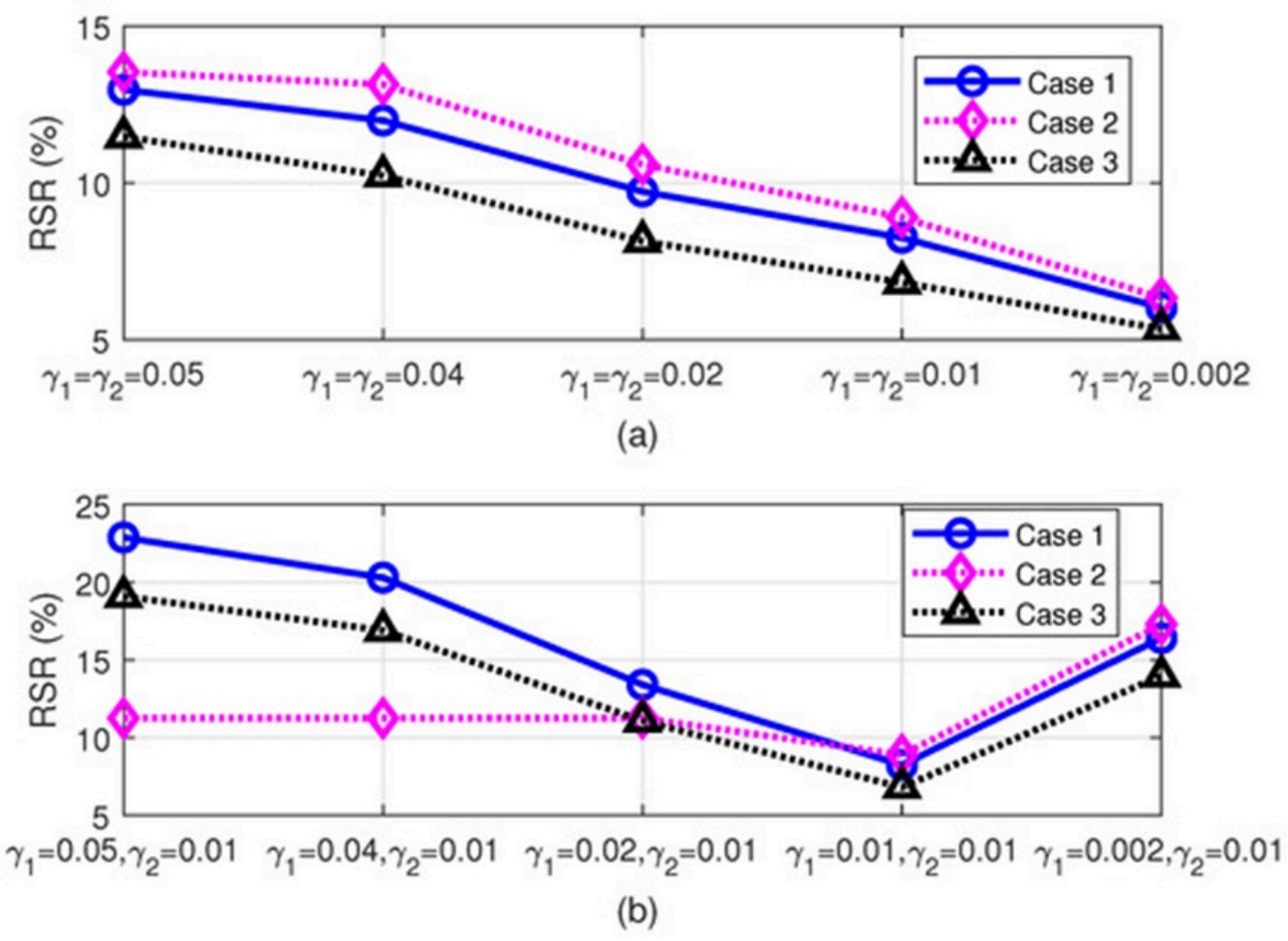

Figure 6. Resource-saving ratios for 2CROB across different cases: (a) Identical RDVR thresholds, and (b) Differentiated RDVR thresholds.

The best-fit $\alpha*$ values recovered from the scenarios in **Figures 4 and 5** are shown in **Figure 7**. In perfectly symmetric settings where the means, variances, and RDVR limits coincide, the optimum settles at $\alpha*$ = 0.5. A noteworthy observation for Case 2 with distinct RDVR limits is that $\alpha*$ jumps to 1 once $\gamma 1$ reaches or exceeds 0.02. This signals that the entirety of RDV occurrences originating within the cooperative overbooking zone are charged to $u1$, and the bottleneck that floors the aggregate resource volume is the RDVR of $u2$, driven solely by Region C events, (C). Accordingly, the aggregate volume plateaus at 7.73 when α = 1, consistent with the relation (7.73) = $\gamma 2$, and this accounts for the flat trajectory of the magenta triangle-marked curve in **Figure 5** beyond $\gamma 1$ = 0.02. Within Case 2, the tenant $u2$—characterised by a wider spread and a tighter RDVR allowance—forms the binding constraint on further shrinkage of the total resource pool, as its RDVR ceiling is reached earlier than that of its counterpart.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

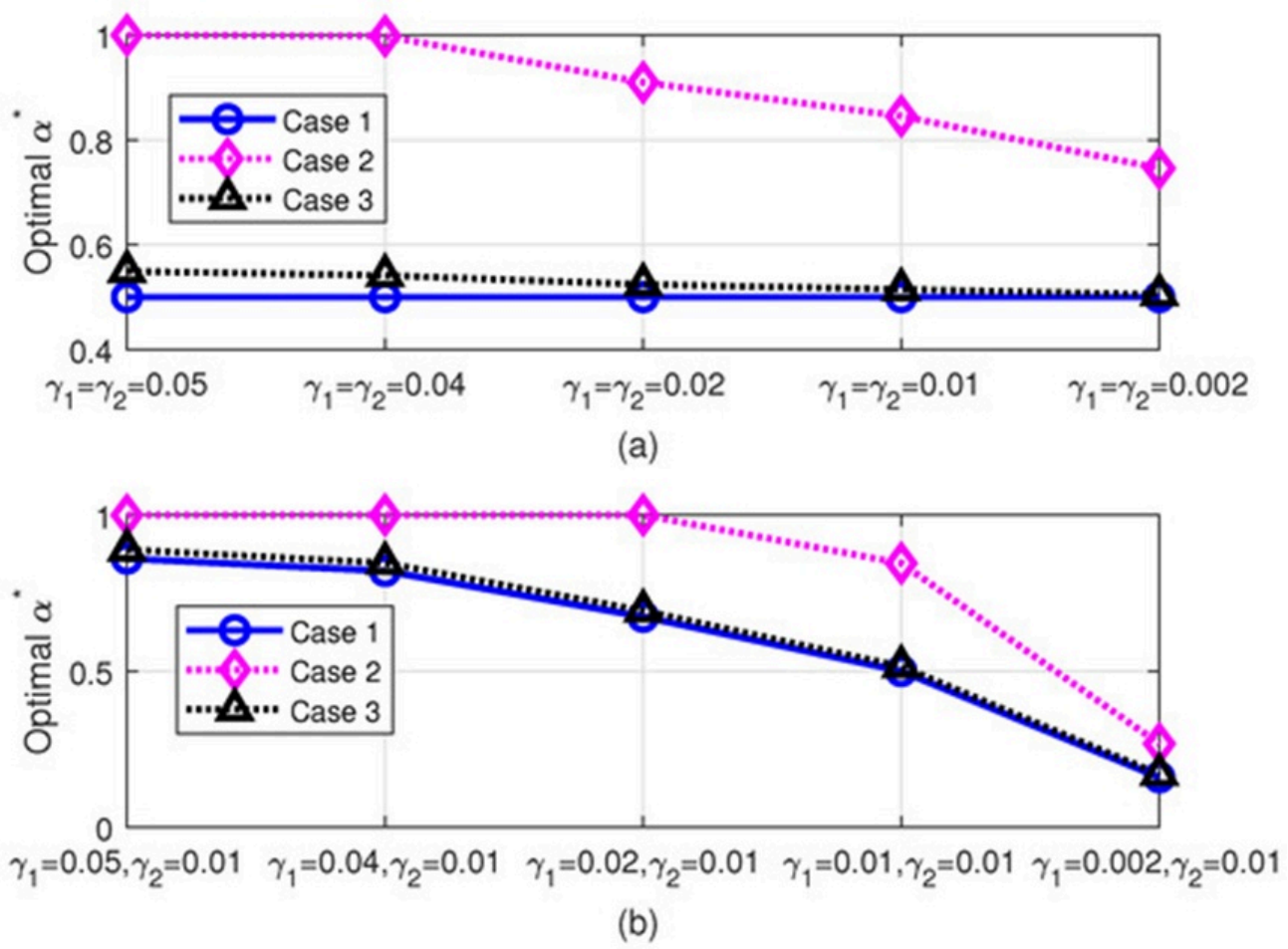

Figure 7. Optimal $\alpha*$ of 2CROB in different cases: (a) Identical RDVR thresholds, and (b) Differentiated RDVR thresholds.

Figure 8 presents the simulated RDVR outcomes for $u1$ and $u2$ alongside the matching $R*$ data from **Figure 5** and $\alpha*$ data from **Figure 7**. The simulation workflow generates abundant demand samples for the two tenants, distributes available computing capacity according to the 2CROB protocol, and tabulates the long-term average RDVRs for each tenant. When the optimal sharing parameter sits below unity, the empirical average RDVRs of both tenants can closely track their respective limits; when $\alpha*$ equals 1, the mean RDVR realised by $u1$ undershoots its permitted ceiling $\gamma1$. This phenomenon, too, follows from assigning every Region E violation to $u1$, leaving no further room for $u1$ to absorb additional violationsr to relieve pressure on $u2$'s RDVR or to enable a lower aggregate resource volume.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

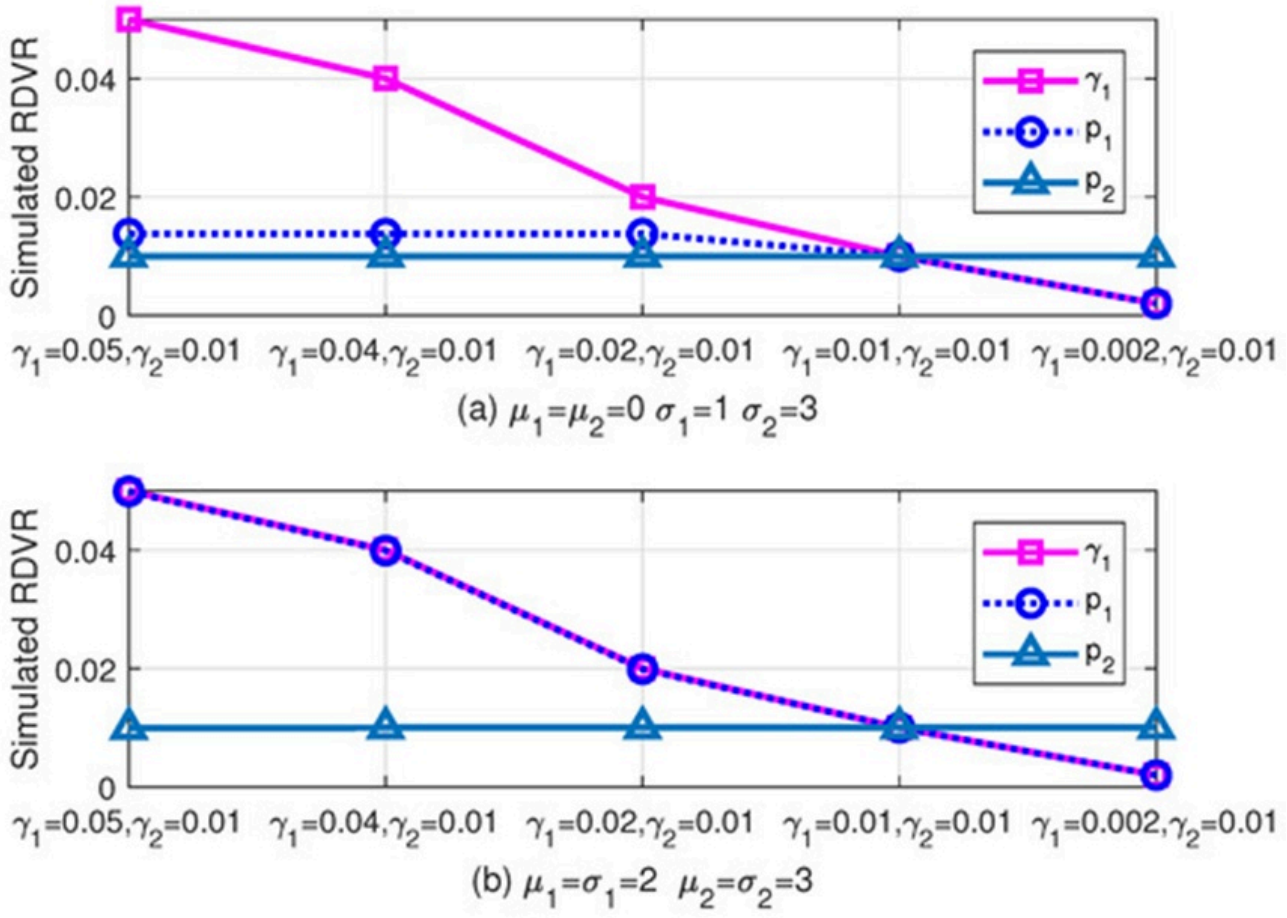

Figure 8. Optimal $\alpha*$ of 2CROB in different cases: (a) Case 2 with differentiated RDVR thresholds, and (b) Case 3 with differentiated RDVR thresholds.

Proportional fairness-based multiple-tenant cooperative resource overbooking

A cross-validation exercise comparing 2CROB against PF-MCROB is conducted using two-VM overbooking instances, with the resultant minimum aggregate volumes $R*$ and attained RDVRs plotted in **Figure 9**. Across all tested configurations, the final aggregate volumes required by PF-MCROB register slightly above those of 2CROB. In the regime where $\alpha*$ < 1, the mean RDVRs delivered by the two schemes coincide and also approach the stipulated thresholds. For Case 1 under $\alpha*$ = 1, the RDVR of $u2$ under PF-MCROB matches the 2CROB benchmark, whereas the RDVR realised by $u1$ through PF-MCROB is incrementally lower. The evidence in **Figure 9** confirms that the PF-MCROB scheme can nearly replicate the optimal minimal aggregate volume attained by 2CROB, while 2CROB squeezes out marginally higher resource utilisation whenever $\alpha*$ < 1.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

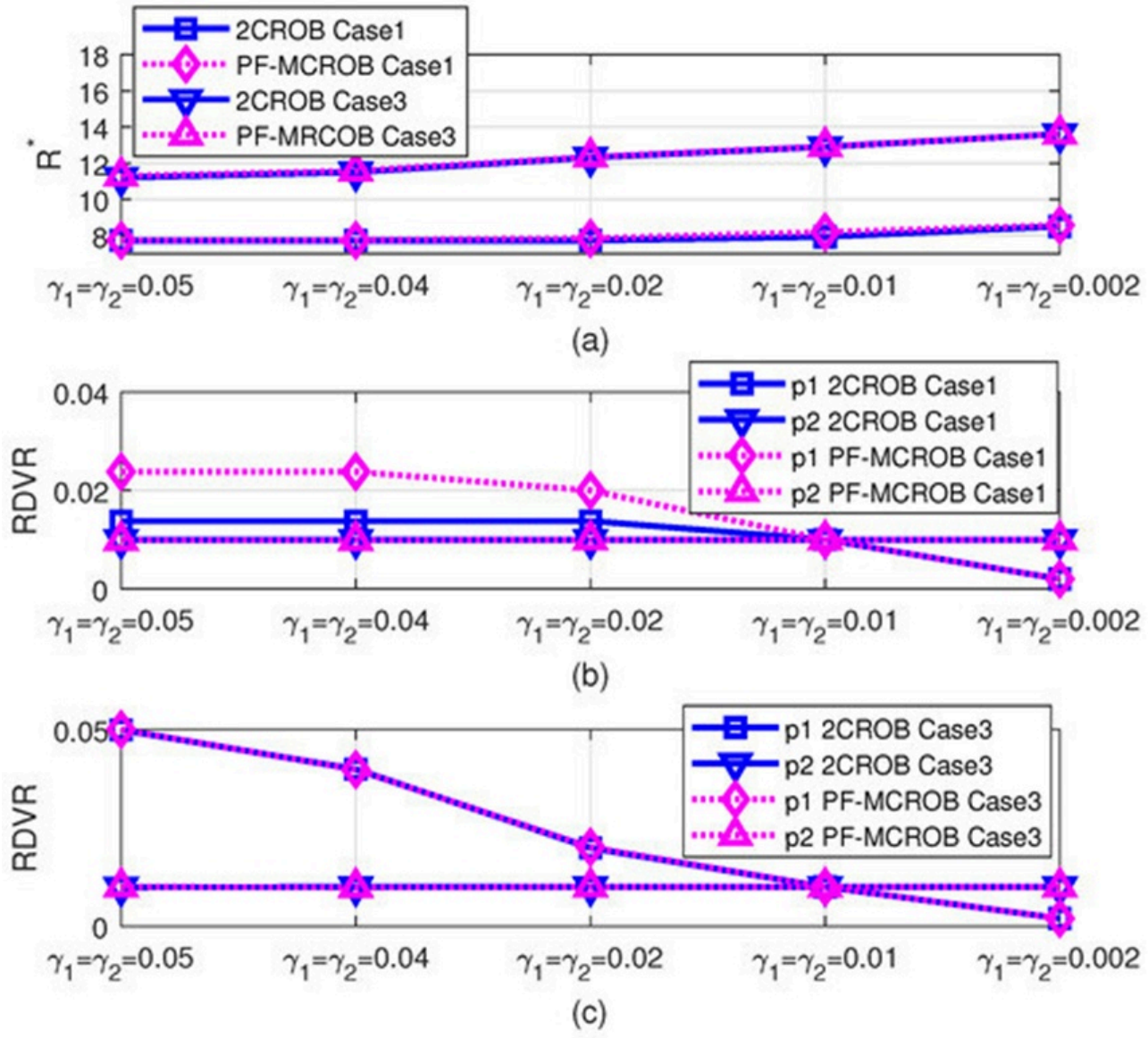

Figure 9. Performance comparison of 2CROB and PF-MCROB in two-VM overbooking cases: (a) Minimum aggregated resource amounts R_* in Case 1 and Case 3, (b) Achieved RDVRs in Case 1, and (c) Achieved RDVRs in Case 3.

The natural habitat for PF-MCROB is the multi-VM overbooking regime, a territory where 2CROB loses direct applicability. **Figure 10** compares the minimal aggregate resource pools produced by NonOB, TradOB, 2CROB, and PF-MCROB across four-VM placement scenarios. The curves for NonOB, TradOB, and 2CROB are constructed using the identical methodology outlined in Section 6.1. The per-VM demand distribution parameters and RDVR limits for each scenario are catalogued in **Table 2**. Extending 2CROB to the four-VM context involves dividing the VMs into two independent pairs, enforcing the 2CROB discipline within each pair, and walling off the resource pools of the two pairs from one another. The PF-MCROB scheme presented here yields the lowest aggregate resource volumes across all four contenders in every tested case, regardless of distribution homogeneity or heterogeneity or RDVR bounds. The margin of improvement over traditional overbooking is especially pronounced when the VMs operate under non-uniform RDVR limits. The root cause is that traditional overbooking forces the aggregate demand to comply with the single most stringent RDVR cap across all co-located VMs.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

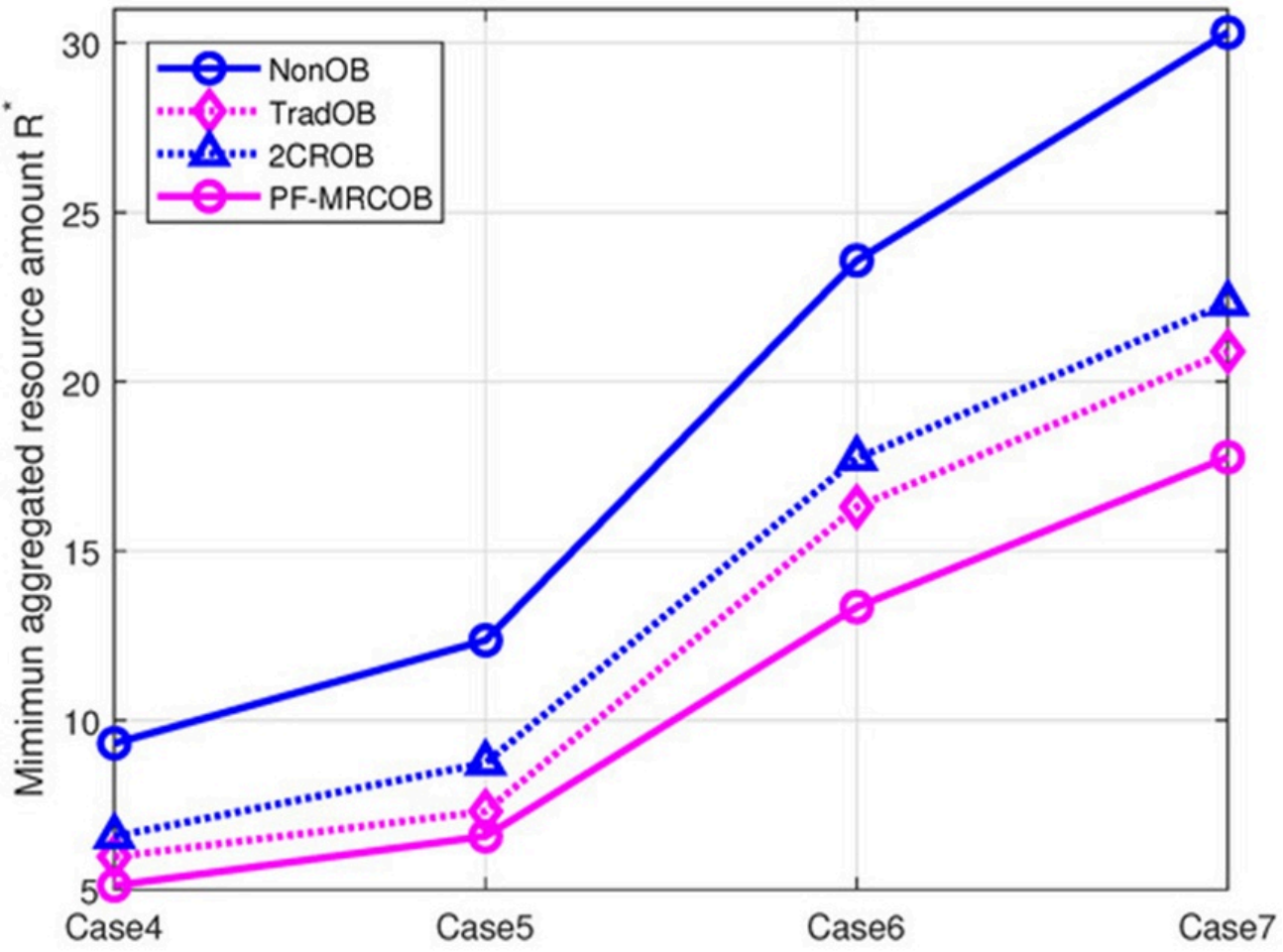

Figure 10. Minimum aggregated resource amounts of different overbooking strategies in four-VM placement cases.

Table 2. Four-VM placement cases in **Figure 10**.

	Resource demand distribution parameter	RDVR threshold
Case 4	$\mu1 = \mu2 = \mu3 = \mu4 = 0$ $\sigma1 = \sigma2 = \sigma3 = \sigma4 = 1$	$\gamma1 = \gamma2 = \gamma3 = \gamma4 = 0.02$
Case 5	$\mu1 = \mu2 = \mu3 = \mu4 = 0$ $\sigma1 = \sigma2 = \sigma3 = \sigma4 = 1$	$\gamma1 = \gamma2 = \gamma3 = \gamma4 = 0.002$
Case 6	$\mu1 = \mu2 = 0\mu3 = \mu4 = 1$ $\sigma1 = 1\sigma2 = 3\sigma3 = 2\sigma4 = 3$	$\gamma1 = 0.05$, $\gamma2 = 0.01$ $\gamma3 = 0.02$, $\gamma4 = 0.01$
Case 7	$\mu1 = \mu2 = 1\mu3 = 2\mu4 = 3$ $\sigma1 = 2\sigma2 = 3\sigma3 = 2\sigma4 = 3$	$\gamma1 = 0.02$, $\gamma2 = 0.01$, $\gamma3 = 0.02$, $\gamma4 = 0.01$

The convergence trajectory of the bisection search procedure embedded within PF-MCROB is traced in **Figure 11**, showing that the algorithm homes in on the target minimal aggregate demand within seven to eight iterations. To shed further light on the general behaviour of PF-MCROB, **Figures 12 and 13** respectively illustrate the minimal aggregate resource volumes of NonOB, TradOB, and PF-MCROB for progressively larger groups of homogeneous VMs, and the

© 2022 The Author(s). | Published by Agro Intelligence Research Press

associated RSR gains of PF-MCROB over TradOB. As shown in **Figure 13**, the proposed PF-MCROB framework progressively unlocks larger fractional resource savings as the degree of VM consolidation on a shared server increases.

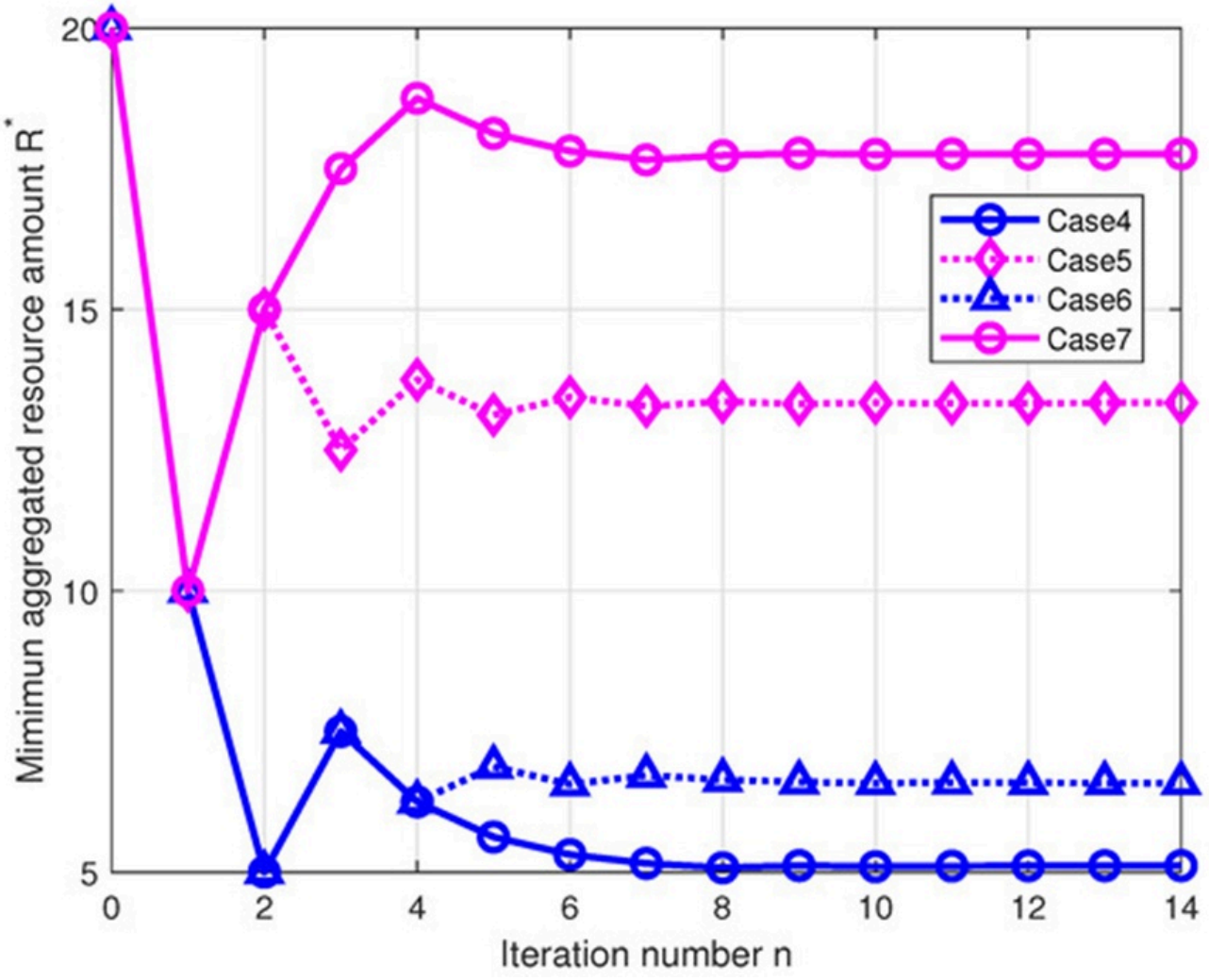

Figure 11. Iteration steps of the bisection search algorithm in PF-MCROB.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

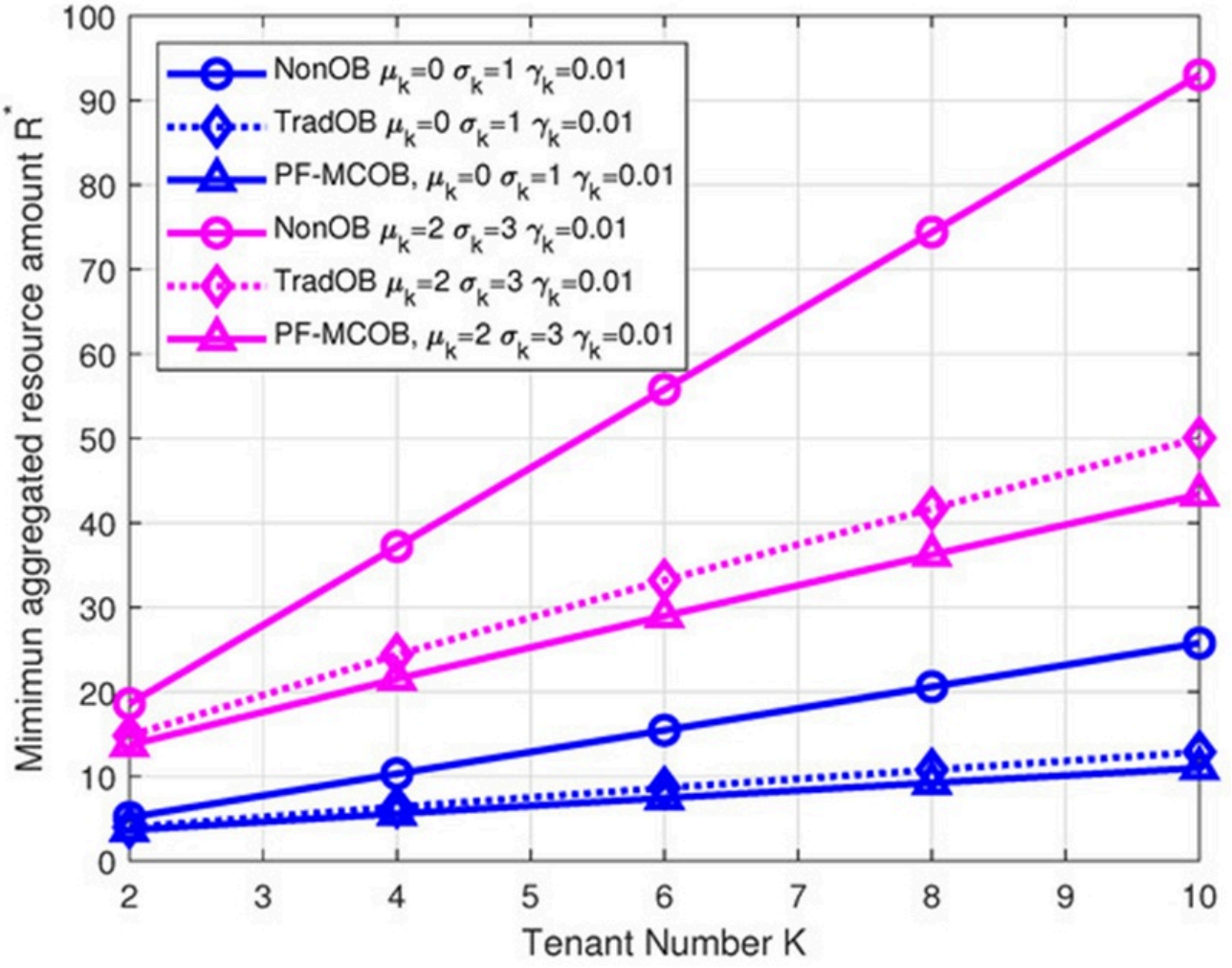

Figure 12. Minimum aggregated resource amounts of different overbooking strategies in K-homogeneous VM placement cases.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

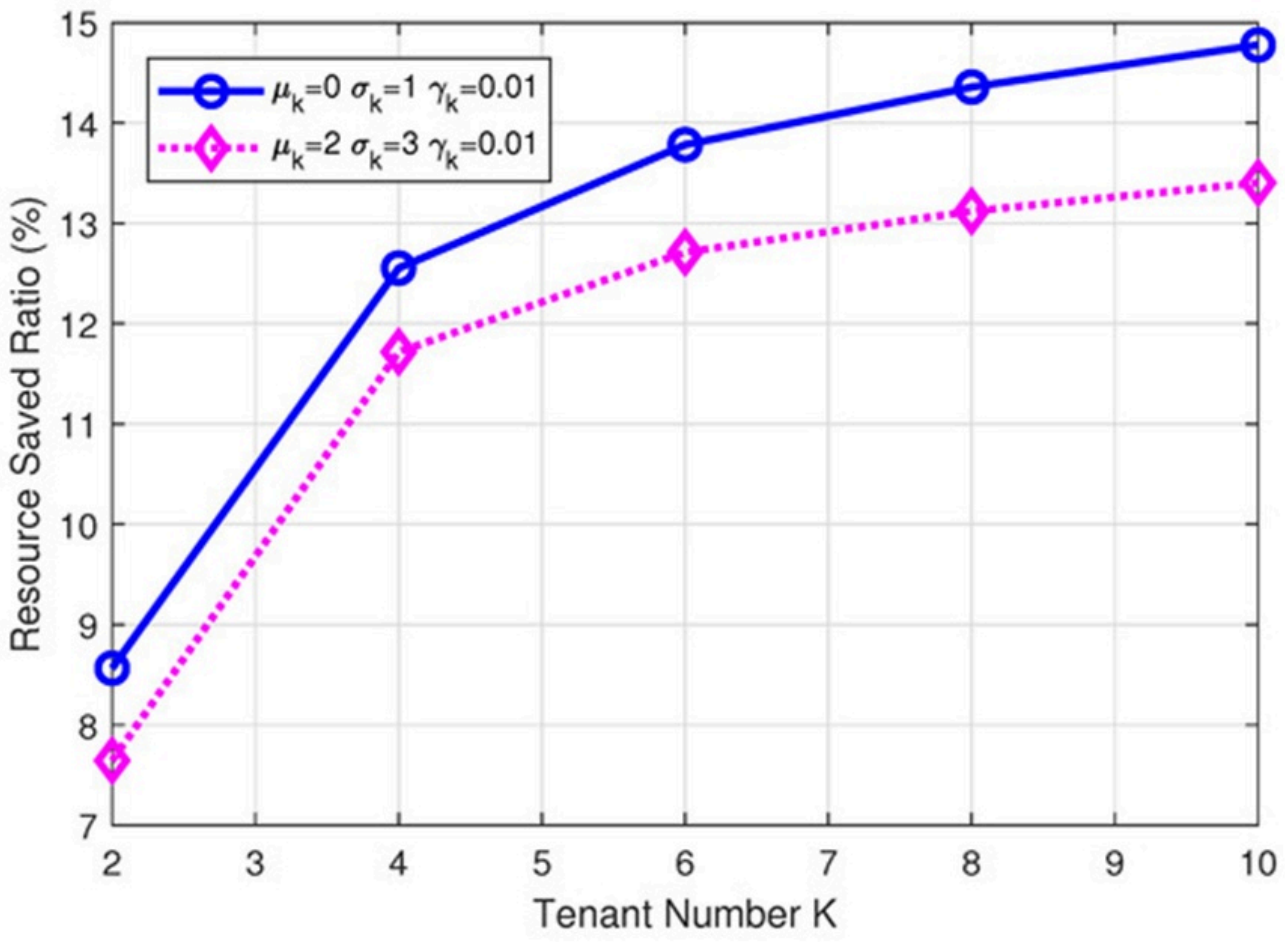

Figure 13. RSR of PF-MCROB in K-homogeneous VM placement cases.

FFD-Based cooperative application placement

To assess the superiority of the FFD-ACP method against competing placement strategies, we monitor the packing sequence of fifty applications and determine the resulting minimal UAV fleet size. Every VM exhibits resource demand that conforms to an independent Truncated Normal Distribution, where the location parameter μk is sampled uniformly across the interval [0,1] and the scale parameter σk is sampled uniformly across [1, 2]. The permissible RDVR ceilings are randomly drawn from the pool {0.002, 0.01, 0.02, 0.03, 0.04}. **Figure 14** illustrates the operational progression of three distinct placement policies taken from a single simulation trial: placement without any overbooking (NonOB AP), placement driven by conventional overbooking (TradOB AP), and the FFD-ACP algorithm advanced in this work. The NonOB AP baseline mirrors the FFD-VMP technique described by Kumaraswamy and Nair [51]. The TradOB AP alternative represents a reduced-complexity derivative of FFD-ACP, wherein the admission test for packing a VM onto an already-active server checks exclusively whether the severest RDVR limit among co-located VMs remains unbreached, expressed as $\sum_{n=1}^{N} \mathscr{I}\left(Y\left(n\right) > Rc\right)/N \leq min\left\{\gamma k, \forall n\right\}$.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

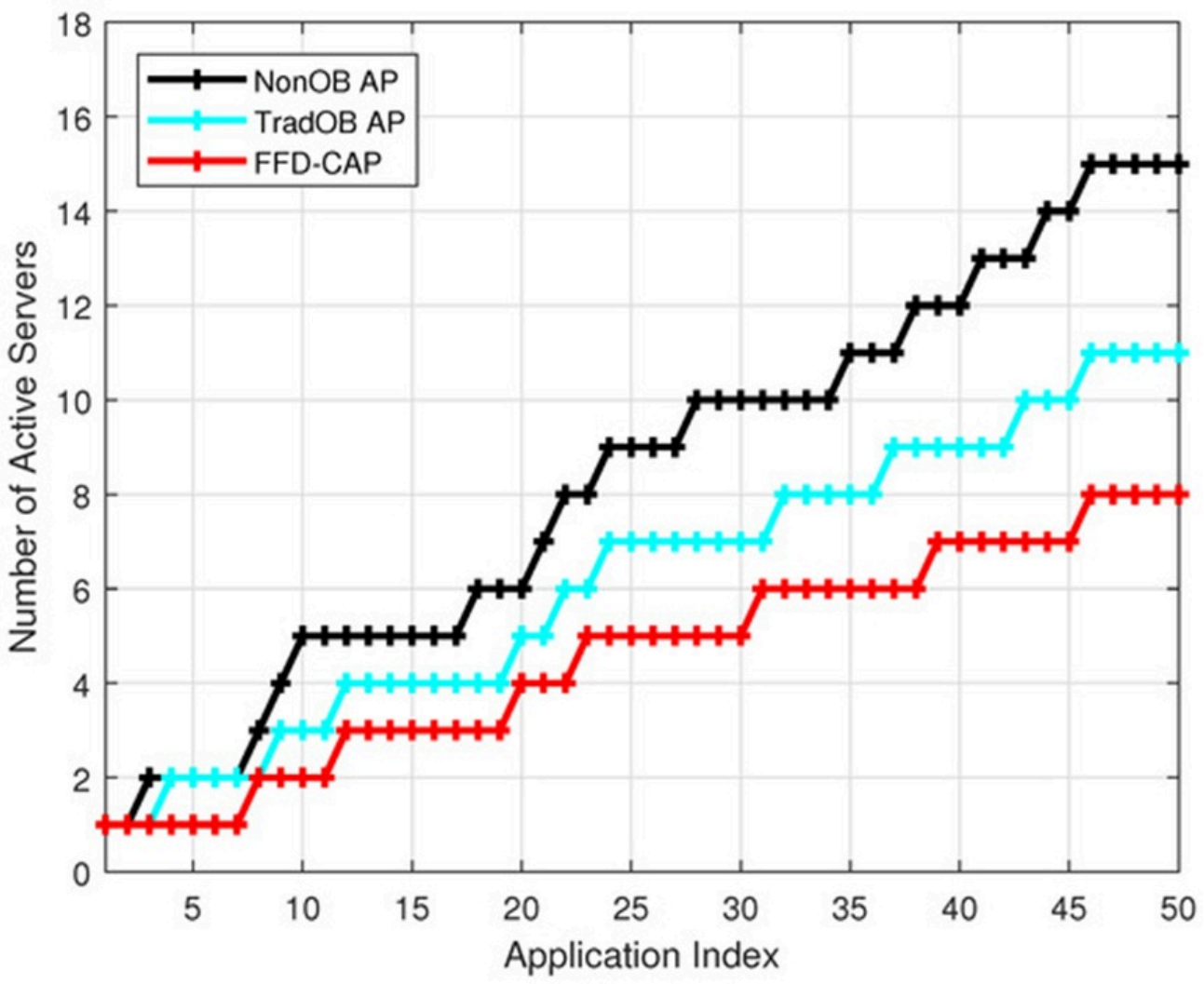

Figure 14. Executing processes of different application placement schemes from one simulation instance.

In the depicted realisation, FFD-ACP successfully accommodates all 50 IoFT applications using only 8 UAVs while upholding their individual RDVR commitments—a near-halving of the fleet compared to what NonOB AP demands. The minimal active servers tallied by the three competing approaches across 50 independent simulation repetitions are shown in **Figure 15**, which reveals that FFD-ACP reduces the UAV count required by traditional overbooking by roughly one-third. The implications are a considerable uplift in onboard resource exploitation and revenue generation, paired with a meaningful contraction in the upfront investment required for UMEC-powered IoFT deployments.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

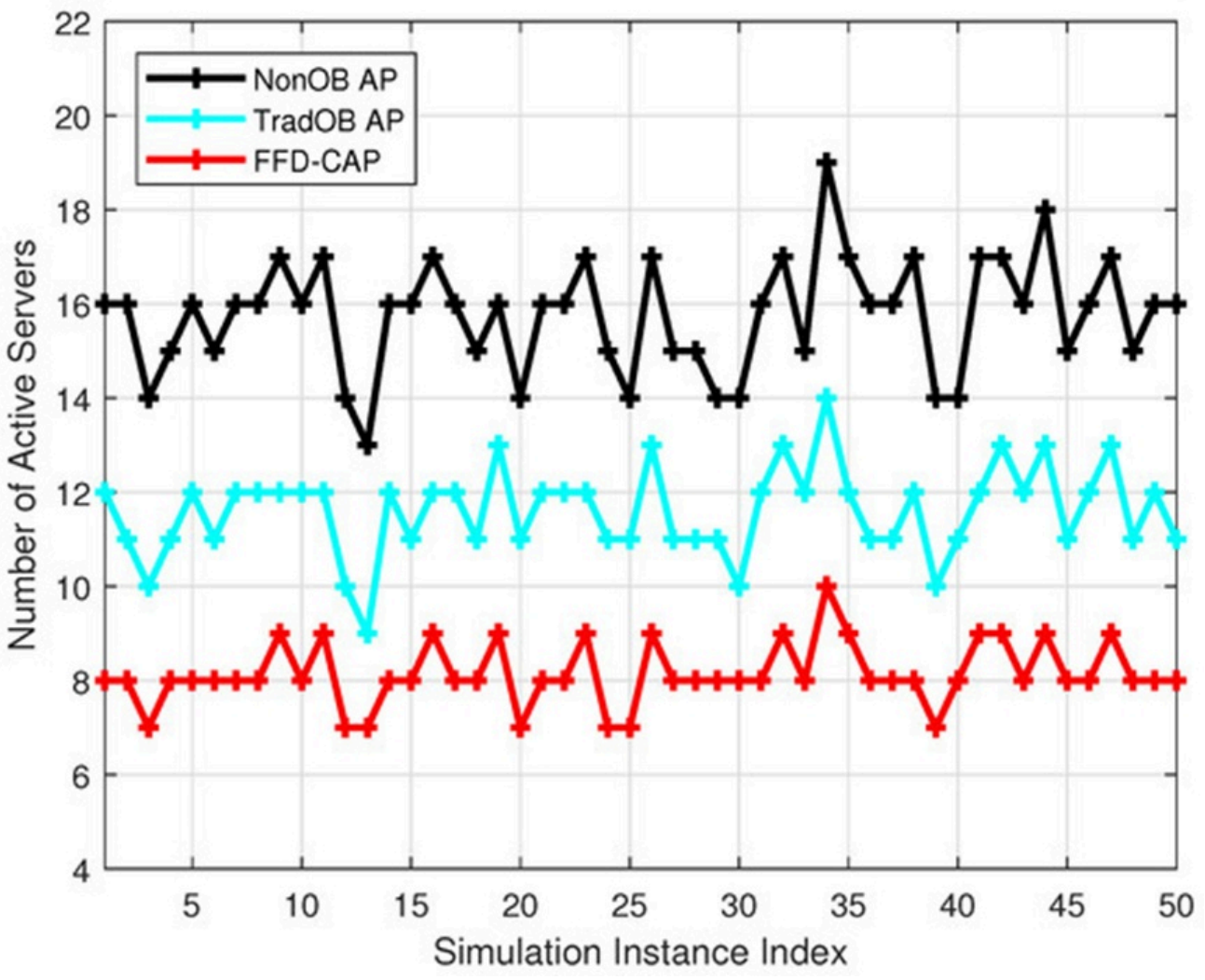

Figure 15. Minimum number of active servers of different application schemes from fifty simulation instances.

Conclusion

This study has examined the coupled resource allocation and application placement problem arising in UMEC settings for IoFT services. A pair of cooperative resource overbooking constructions was engineered—one addressing the two-tenant overbooking case and another extending to arbitrary numbers of tenants—whereby hosted applications or their corresponding VMs can pool their RDV occurrences to compress the total resource reservation. Two separate bisection search routines were devised to locate the minimum aggregate resource volumes for the dual-tenant and multi-tenant scenarios, respectively. Leveraging the multi-tenant cooperative overbooking framework, an FFD-ACP procedure was then formulated to consolidate applications into a minimised set of UAV-borne servers. Simulation-based evaluation substantiated the merit of the proposed contributions, demonstrating that the fractional resource savings unlocked by cooperative overbooking become increasingly substantial as the tenant count grows and as RDVR allowances are set higher or made more diverse; additionally, FFD-ACP proved capable of eliminating one-third of the otherwise necessary UAVs. Collectively, the cooperative resource overbooking and application placement solutions outlined here can substantially pare back deployment costs and amplify the operational profitability of UMEC platforms. That said, several boundaries of the current work warrant acknowledgement.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

To begin with, both 2CROB and PF-MCROB presuppose that the resource demands of distinct VMs are statistically independent and that their PDFs have been reliably inferred from historical measurement logs. In settings where VM demands display interdependence or where the underlying distributions remain unidentified, the proposed mechanisms cease to function. Beyond that, the scope of this paper was confined to computing resource overbooking. In contrast, the same philosophy could be transposed to additional resource types—link bandwidth, energy budgets, and storage volumes—paving the way toward multidimensional overbooking. Moreover, the VM's demand distribution is itself contingent on the UAV's sensing radius, the population of ground devices, and the influx rate of MEC task submissions. Accordingly, future enhancements to UMEC application placement could pursue joint optimisation across a richer set of decision variables, including UAV flight trajectories, communication bitrates, and energy-efficiency targets.

Acknowledgements

The authors would like to thank the Editor-in-Chief, Editor, and anonymous reviewers for their valuable reviews.

Conflict of interest

None

Financial support

This work was supported by the National Natural Science Foundation of China (62101415).

Ethics statement

None

Received: 03 Aug 2021 Revised: 08 Oct 2021 Accepted: 25 Oct 2021
Published online: 25 May 2022

Rights and permissions

Open Access The author(s) retain copyright. This article is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. It may be shared and adapted for non-commercial purposes with appropriate attribution, an indication of changes, and distribution of adaptations under the same license. Third-party material may be subject to separate terms identified in its credit line. View the license at https://creativecommons.org/licenses/by-nc-sa/4.0/.

References

1. Li C, Li G, Song Y, He Q, Tian Z, Xu H, et al. Fast forest fire detection and segmentation application for uav-assisted mobile edge computing system. IEEE Internet Things J. 2024;11:26690–9.

2. Lv L, Zhao Y, Li X, Yu J, Song M, Huang L, et al. UAV-based intelligent detection of individual trees in Moso bamboo forests with complex canopy structure. IEEE J Sel Top Appl Earth Obs Remote Sens. 2024;17:11915-30.

3. Wu D, Qian Z, Wu D, Wang J. FSNet: Enhancing forest-fire and smoke detection with an advanced UAV-based network. Forests. 2024;15:787.

4. Singh R, Gehlot A, Vaseem Akram S, Kumar Thakur A, Buddhi D, Kumar Das P. Forest 4.0: Digitalization of forest using the Internet of Things (IoT). J King Saud Univ Comput Inf Sci. 2022;34:5587-601.

5. Spencer G, Torres PMB. New CAN bus communication modules for digitizing forest machines functionalities in the context of Forestry 4.0. IEEE Access. 2023;11:9058-66.

6. Sahal R, Alsamhi SH, Breslin JG, Ali MI. Industry 4.0 towards Forestry 4.0: Fire detection use case. Sensors. 2021;21:694.

7. Pham HX, La HM, Feil-Seifer D, Deans MC. A distributed control framework of multiple unmanned aerial vehicles for dynamic wildfire tracking. IEEE Trans Syst Man Cybern Syst. 2020;50:1537-48.

8. Xu XR, Xu YH, Zhou W, Nallanathan A. Energy efficient resource allocation for UAV-served energy harvesting-supported cognitive industrial M2M networks. IEEE Wirel Commun Lett. 2023;12:1454-8.

9. Arafat MY, Pan S. Urban air mobility communications and networking: Recent advances, techniques, and challenges. Drones. 2024;8:702.

10. Bushnaq OM, Chaaban A, Al-Naffouri TY. The role of UAV-IoT networks in future wildfire detection. IEEE Internet Things J. 2021;8:16984-99.

11. He Y, Wang D, Huang F, Zhang R, Min L. Aerial-ground integrated vehicular networks: A UAV-vehicle collaboration perspective. IEEE Trans Intell Transp Syst. 2024;25:5154-69.

12. Zhao W, Liu J, Guo H, Hara T. ETC-IoT: Edge-node-assisted transmitting for the cloud-centric Internet of Things. IEEE Netw. 2018;32:101-7.

13. Yang S, Li F, Shen M, Chen X, Fu X, Wang Y. Cloudlet placement and task allocation in mobile edge computing. IEEE Internet Things J. 2019;6:5853-63.

14. Pan J, McElhannon J. Future edge cloud and edge computing for Internet of Things applications. IEEE Internet Things J. 2018;5:439-49.

15. Liu B, Liu C, Peng M. Resource allocation for energy-efficient MEC in NOMA-enabled massive IoT networks. IEEE J Sel Areas Commun. 2021;39:1015-27.

16. Zhang J, Li T, Ying Z, Ma J. Trust-based secure multi-cloud collaboration framework in cloud-fog-assisted IoT. IEEE Trans Cloud Comput. 2023;11:1546-61.

17. Yu Z, Gong Y, Gong S, Guo Y. Joint task offloading and resource allocation in UAV-enabled mobile edge computing. IEEE Internet Things J. 2020;7:3147-59.

18. Ji J, Zhu K, Niyato D. Joint communication and computation design for UAV-enabled aerial computing. IEEE Commun Mag. 2021;59:73-9.

19. Xiong J, Guo H, Liu J. Task offloading in UAV-aided edge computing: Bit allocation and trajectory optimization. IEEE Commun Lett. 2019;23:538-41.

20. Zhan C, Hu H, Sui X, Liu Z, Niyato D. Completion time and energy optimization in the UAV-enabled mobile-edge computing system. IEEE Internet Things J. 2020;7:7808-22.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

21. Quan W, Liu Y, Zhang H, Yu S. Enhancing crowd collaborations for software defined vehicular networks. IEEE Commun Mag. 2017;55:80-6.

22. Cheng N, Lyu F, Quan W, Zhou C, He H, Shi W, et al. Space/aerial-assisted computing offloading for IoT applications: A learning-based approach. IEEE J Sel Areas Commun. 2019;37:1117-29.

23. Chakraborty T, Toosi AN, Kopp C. Elastic power utilization in sustainable micro cloud data centers. IEEE Trans Sustain Comput. 2023;8:465-78.

24. Marquez C, Gramaglia M, Fiore M, Banchs A, Costa-Perez X. Resource sharing efficiency in network slicing. IEEE Trans Netw Serv Manag. 2019;16:909-23.

25. Tang Z, Zhang F, Zhou X, Jia W, Zhao W. Pricing model for dynamic resource overbooking in edge computing. IEEE Trans Cloud Comput. 2023;11:1970-84.

26. Yang R, Hu C, Sun X, Garraghan P, Wo T, Wen Z, et al. Performance-aware speculative resource oversubscription for large-scale clusters. IEEE Trans Parallel Distrib Syst. 2020;31:1499-517.

27. Malla S, Christensen K. A survey on power management techniques for oversubscription of multi-tenant data centers. ACM Comput Surv. 2019;52:1-31.

28. Al–khatib AA, Hassan MU, Moessner K. Heuristic optimization of bandwidth reservation cost for vehicular applications. In: Proceedings of the GLOBECOM 2022—2022 IEEE Global Communications Conference; Rio de Janeiro, Brazil; 2022; p. 4909-15.

29. Shy O. How to price: A guide to pricing techniques and yield management. In: How to Price: A Guide to Pricing Techniques and Yield Management. Cambridge, UK: Cambridge University Press; 2008. p. 297-324.

30. Farhadi V, Mehmeti F, He T, Porta TFL, Khamfroush H, Wang S, et al. Service placement and request scheduling for data-intensive applications in edge clouds. IEEE/ACM Trans Netw. 2021;29:779-92.

31. Wang W, Zhao Y, Tornatore M, Gupta A, Zhang J, Mukherjee B. Virtual machine placement and workload assignment for mobile edge computing. In Proceedings of the 2017 IEEE 6th International Conference on Cloud Networking (CloudNet), Prague, Czech Republic; 2017. p. 1–6.

32. Zhao L, Liu J. Optimal placement of virtual machines for supporting multiple applications in mobile edge networks. IEEE Trans Veh Technol. 2018;67:6533-45.

33. Zhang X, Li Z, Lai C, Zhang J. Joint edge server placement and service placement in mobile-edge computing. IEEE Internet Things J. 2022;9:11261-74.

34. Ouyang T, Zhou Z, Chen X. Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing. IEEE J Sel Areas Commun. 2018;36:2333-45.

35. Li J, Yi C, Chen J, Zhu K, Cai J. Joint trajectory planning, application placement, and energy renewal for UAV-assisted MEC: A triple-learner-based approach. IEEE Internet Things J. 2023;10:13622-36.

36. Huda SA, Moh S. Survey on computation offloading in UAV-enabled mobile edge computing. J Netw Comput Appl. 2022;201:103341.

37. Qu Y, Dai H, Wang H, Dong C, Wu F, Guo S, et al. Service provisioning for UAV-enabled mobile edge computing. IEEE J Sel Areas Commun. 2021;39:3287-305.

© 2022 The Author(s). | Published by Agro Intelligence Research Press

38. Yang Z, Bi S, Zhang YJA. Dynamic offloading and trajectory control for UAV-enabled mobile edge computing system with energy harvesting devices. IEEE Trans Wirel Commun. 2022;21:10515-28.

39. Urgaonkar B, Shenoy P, Roscoe T. Resource overbooking and application profiling in shared hosting platforms. ACM SIGOPS Oper Syst Rev. 2002;36:239-54.

40. Sulistio A, Kim KH, Buyya R. Managing cancellations and no-shows of reservations with overbooking to increase resource revenue. In Proceedings of the 2008 Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID), Lyon, France; 2008; IEEE: Piscataway Township, NJ, USA, 2008. p. 267–76.

41. Bashir N, Deng N, Rzadca K, Irwin D, Kodak S, Jnagal R. Take it to the limit: Peak prediction-driven resource overcommitment in datacenters. In Proceedings of the Sixteenth European Conference on Computer Systems, Virtual; 2021. p. 556–73.

42. Salvat JX, Zanzi L, Garcia-Saavedra A, Sciancalepore V, Costa-Perez X. Overbooking network slices through yield-driven end-to-end orchestration. In Proceedings of the 14th International Conference on Emerging Networking Experiments and Technologies, Heraklion, Greece; 2018. p. 353–65.

43. Wang X, Wang X, Zheng K, Yao Y, Cao Q. Correlation-aware traffic consolidation for power optimization of data center networks. IEEE Trans Parallel Distrib Syst. 2015;27:992-1006.

44. Meng X, Isci C, Kephart J, Zhang L, Bouillet E, Pendarakis D. Efficient resource provisioning in compute clouds via VM multiplexing. In Proceedings of the 7th International Conference on Autonomic Computing, Washington, DC, USA; 2010. p. 11–20.

45. Guerin R, Ahmadi H, Naghshineh M. Equivalent capacity and its application to bandwidth allocation in high-speed networks. IEEE J Sel Areas Commun. 1991;9:968-81.

46. Wang M, Meng X, Zhang L. Consolidating virtual machines with dynamic bandwidth demand in data centers. In Proceedings of the 2011 Proceedings IEEE INFOCOM, Shanghai, China, 2011; IEEE: Piscataway Township, NJ, USA; 2011. p. 71–5.

47. Zhang Z, Su S, Zhang J, Shuang K, Xu P. Energy aware virtual network embedding with dynamic demands: Online and offline. Comput Netw. 2015;93:448-59.

48. Liu F, Li H, Shi K. Temporal Graph based Overcommitted Routing for Deterministic Networking. In Proceedings of the 2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall), Beijing, China, 2022; IEEE: Piscataway Township, NJ, USA; 2022. p. 1–6.

49. Sexton C, Marchetti N, DaSilva LA. On provisioning slices and overbooking resources in service tailored networks of the future. IEEE/ACM Trans Netw. 2020;28:2106-19.

50. Liwang M, Wang X, Chen R. Computing resource provisioning at the edge: An overbooking-enabled trading paradigm. IEEE Wirel Commun. 2022;29:68-76.

51. Kumaraswamy S, Nair MK. Bin packing algorithms for virtual machine placement in cloud computing: A review. Int J Electr Comput Eng. 2019;9:512.

© 2022 The Author(s). | Published by Agro Intelligence Research Press