A survey on time-sensitive resource allocation in the cloud continuum

1 Introduction

Artificial Intelligence (AI) and the Internet of Things (IoT) paradigm are transforming the field of computing. AI-based applications (such as computer vision, augmented reality, natural language processing, etc.) are inherently compute-intensive and IoT (such as Industrial IoT, smart cities, connected vehicles and etc.) introduces unprecedented decentralization making them communication-intensive as well. Cloud computing seems like a natural choice for these applications. The conventional cloud computing has evolved into today’s edge (also known as cloudlets or fog) where computing occurs closer to the end devices that are typically mobile. Such a generic multi-tier cloud architecture, what we call the cloud continuum, is shown in Figure 1.

One of the foremost challenges in cloud resource allocation is the ability to satisfy the latency or deadline guarantees of an application. With the advent of 5G ultra-reliable low latency communication (uRLLC), time-sensitive applications such as telehealth, digital twins, and connected and autonomous cars, are expected to rely on the cloud continuum [1]. For this reason, we expect to see an evolution of resource allocation techniques in the literature where the cloud continuum is modeled to handle time-sensitive applications, and hence these studies are the focus of this survey.

Most works rely on a specific cloud model and define their own terminology. Therefore, we first define a generic cloud model and terminologies that encompass the surveyed literature. Existing works have majorly focused on three classes of problems: 1) The offloading decision problem of whether to offload application computation from an end device to the edge and cloud or not. 2) The resource provisioning problem of allocating the computation and/or communication resources to the applications. 3) The resource scheduling problem of when to use the allocated computation and communication resources. The aim is to classify these works based on the type of problem they address, as well as the nature of the solution they propose (analytical or heuristic, centralized or decentralized, etc.). For time-sensitivity, we group the literature based on
two objectives: response time minimization and satisfaction of hard deadlines.

There are quite some works in the literature that consider time-sensitive applications. However, due to space limitations, it is not possible to cover all of them in this survey. We have therefore chosen papers based on the publication date (2013–April 2020) and the reputation of the venue (IEEE INFOCOMM, GLOBECOM, TPDS, TC, TCC, ICDCS). We suppose that studies earlier than 2013 are superseded by the later ones. Additionally, we filtered papers based on the quality of the proposed solution; those based on primitive heuristics or a simple application of optimization solvers are ignored.

To the best of our knowledge, we are the first to survey resource allocation studies in the cloud continuum for time-sensitive applications.

2 Multi-tier cloud architecture

Existing literature models the cloud resources either as a collection of servers or as a set of servers interconnected by a backhaul network in a tiered architecture. Some studies consider the application workload as virtual machines (VM) with specific requirements (in terms of computation, storage, etc.) or abstract it using fractional requirements such as cycles/second (computation) or bits/second (communication). Hence, in order to classify this diverse literature there is a need to define a baseline cloud architecture model and terminologies (Figure 1 and Table 1).

The cloud servers, denoted by $N^C$, are the top-tier of the architecture with large amounts of resources. Each cloud server $n \in N^C$ has $C^C_{n,r}$ amount of type-$r$ resources. The cloud servers are connected to the edge servers with lower resource capacity by a high-speed core network. The amount of type-$r$ resource at the edge server location $n \in N^E$ is given by $C^E_{n,r}$. It is assumed that each edge server may have an access point through which the devices are connected to it. Each edge server has a bandwidth capacity for offloading workload tasks (generated by the devices). The servers are internally connected by a backhaul network. Generally, it is assumed that the core and the backhaul network have infinite bandwidth for data transmission. However, with growing data-intensive applications, geo-distributed servers and distributed service providers this assumption would soon become invalid.

The set of resources (processors, memory, storage) available at the cloud/edge is given by $R$. The set of virtual machines (VM’s) of specific configuration or services at the server location $x$ is given by $M_x$ and the corresponding amount of type-$r$ resource required to host them is given by $H_{r,m}$, where $r \in R$, $m \in M_x$. Let $\mu_x$ denote the serving rate of the tasks at $x \in \{N^C \cup N^E\}$. The resource provisioning problem in general encompasses allocation of these resources and/or bandwidth to the device tasks. Different studies consider different provisioning problems such as joint resource (both computation and communication resource) allocation, only VM/bandwidth allocation, etc.

The computation time ($\delta_P$) of a task depends on the computation capacity of the server/device and any queuing delay. The computation capacity is computed either based on the computation speed (cycles per unit time) or the serving rate (tasks per unit time) of the server/de-
Table 1: Model parameters. Note, $\mu_x$ and $T_n$ are mutually exclusive and either one of them can be used.

| Symbol | Description |
|--------|-------------|
| $N^c$  | Set of cloud servers |
| $C_{x,r}$ | Amount of type-$r$ resource available (capacity) at cloud server $n$ |
| $N^e$  | Set of edge servers |
| $C_{x,r}$ | Amount of type-$r$ resource available (capacity) at edge server $n$ |
| $B_n$  | Bandwidth capacity of an edge server $n$ for task offloading |
| $M_x$  | Set of VM’s of specific configurations or services in $x$, where $x \in \{N^c \cup N^e\}$ |
| $R$    | Set of resources (Storage, Memory, CPUs) |
| $H_{r,m}$ | Amount of type-$r$ resource required to host type-$m$ VM or service, where $r \in R$, $m \in M_x$ |
| $\lambda_x$ | Serving rate of tasks in $x$, where $x \in \{N^c \cup N^e\}$ |
| $\lambda_n$ | Arrival rate of task $n$ |
| $N_{n,m}$ | Number of type-$m$ VM’s or services requested by task $n$ |
| $T_n$  | Duration of task $n$ |
| $D_n$  | Deadline constraint of task $n$ |
| $S_n$  | Data size of a task $n$ |
| $E_n$  | Constraint on edge server serving task $n$ |
| $B_n$  | Offloading bandwidth of task $n$ |
| $d_{x,y}$ | Communication delay between entities $x$ and $y$ where $(x,y) \in \{N^c \cup N^e \cup N\}$ |

vice. Certain works assume servers have queues for storing tasks as their arrival rate may be higher than the computation capacity. This waiting time experienced by a task due to other tasks pending ahead of it is denoted as the *queuing delay*. Some works assume the computation time of tasks are known a priori or constant and do not vary based on server location, while other works assume the computation time is dynamic or not known a priori.

The task (or device) $n \in N$ requests for a particular VM or service of type-$m$ for a specified duration (execution time) $T_{n,m}$. Each task is expected to be generated at a rate of $\lambda_n$. Studies that consider offloading decision problem decide whether the task (in part or complete) is chosen to execute on the device and/or the server. The order in which these tasks are executed once they are allocated to a server is determined by the scheduling problem. Each task may need to transfer data of size $S_n$ to the server and can have an offloading bandwidth of $B_n$. Depending on the application the studies consider, the device may be static and offload/receive data from the same server or be mobile and offload/receive data from different servers. This application requirement altogether changes the resource provisioning problem. The task may specify whether it needs to be served within a deadline constraint $D_n$. Applications that are delay-sensitive, such as autonomous driving or industrial IoT may require resource allocation techniques to adhere to their deadlines. There is a delay involved in sending the task data from the device to the servers or between servers. It is given by $d_{x,y}$, where $(x,y) \in \{N^c \cup N^e \cup N\}$. Studies do not necessarily consider all these delay parameters. However, in a realistic platform these delays are bound to occur.

**Communication time** ($\delta_T$) is the time required to transmit the data from one entity to another including the communication delay. Several factors such as allocated bandwidth, interference, noise, device transmission power and distance play a role in determining this parameter. Depending on the communication technology such as wireless communication, 5G technology, etc. between devices and/or servers this parameter may have a varied impact on resource allocation policies.

The elapsed time between a task’s release and its completion is denoted as the *response time*. This includes the computation time ($\delta_p$) and the communication time ($\delta_T$) for all entities on which the task executes. Some works also consider *makespan*, which is the maximum response time among all the tasks. Applications such as natural language processing, face recognition which are delay-
tolerant would favour resource allocation techniques that minimize response time/makespan.

3 Literature review

In this section, we survey important resource allocation techniques that have been developed for the cloud continuum for time-sensitive applications. To classify this literature, we use the following taxonomy.

1. **Problem type.** We consider two problem types; one based on the timing model and another based on the contention model. The timing model helps us to understand the objective of the study and the nature of the application being considered. Meanwhile, the resource contention model helps us to understand the assumption in the model parameters and architecture. Although resource contention has an impact on the timing model, it helps us to segregate the class of literature that has similar assumptions in the model and focus on the same objective.
   - **Timing model.** Studies that consider workload tasks with hard deadline requirements are classified under **deadline constrained** and presented in Section 3.2. The remaining works are categorized under **response time minimization**, including few studies that consider the makespan minimization problem, and presented in Section 3.1. Studies that consider delay-tolerant applications, and performance metrics such energy/cost minimization and revenue/utility maximization in general consider response time/makespan minimization. On the other hand, studies that consider time-critical applications focus on the deadline constrained model.
   - **Contention model.** Depending on the contention model for the communication and/or the computation resources, the works are further classified as **no contention** (i.e., computation and communication resources are not shared between the tasks), **only communication contention** (i.e., tasks contend ONLY for offloading bandwidth $B_n$ and $\sum_i B_n$ is bounded by $B_n^E$), **only computation contention** (i.e., tasks contend ONLY for computation resources and in general, it is bounded by $C_n^E$, $C_n^R$ or $\mu_n$) and both **communication and computation contention**. Resource contention decides whether allocating resources to one task has an impact on other tasks. Studies with no resource contention are relatively easier to solve as resources are unconstrained. These studies assume the resources are abundant and may consider over-provisioning of resources to meet timing requirements of tasks. In the case of resource constrained studies, the order in which resources are allocated to tasks is important as it may affect the overall performance of the algorithm. As resources are constrained, over-provisioning is not feasible in these studies.

2. **Solution type.** We categorize the works based on the proposed solution type: **centralized** or **decentralized** algorithms. We further classify this based on the nature of solution.
   - **Nature of solution.** Techniques that solve the problem or a relaxed variant of the problem either optimally or with an approximation bound are grouped under **analytical** solutions. The approximation bound could either be a **constant** or depend on the task and server parameters (denoted as **parameterized approximation bound**). The remaining works that propose heuristic techniques including meta-heuristic approaches are grouped under **heuristic** solutions. Note that, we cannot directly compare the accuracy or complexity of these solutions as each study has different model parameters and assumptions.

Table 2 shows the classification of literature based on the above taxonomy. We also identify the problem class (offloading, provisioning and scheduling) for each study in the same table. The literature review discussed in the subsequent sub-sections is based on the classification presented in this table.

3.1 Response time minimization

Many studies aim to minimize the latency experienced by tasks under various constraints. The most common timing-related objective found in these studies is that of task response time minimization. These include minimizing the average task response times (i.e., $\min \sum_N (\delta_p + \delta_f)$) or minimizing the overall makespan (i.e., $\min \max_{N} (\delta_p + \delta_f)$). In this section, we review the literature that consider these two problems and categorize them based on their respective contention model.

**No contention**

Works in this category mainly focus on the task offloading problem on single-tier architectures with optimization
objectives such as minimizing task response times \([39, 77]\) and device energy \([20, 77]\) and multi-tier architectures with response time minimization \([22]\).

Kao and Krishnamachari \([39]\) model the workload as a Directed Acyclic Graph (DAG) where vertices represent tasks and edges represent data dependencies among them. Using dynamic programming the DAG is split into multiple trees and the response-time of each tree is optimized using time quantization, as in \([38]\). They present a Fully Polynomial Time Approximation Scheme (FPTAS) with an approximation factor of \((1 + \epsilon)\), where \(\epsilon \in [0, 1]\) is chosen by users to reach a trade-off between optimality and algorithm runtime. Ding et al. \([20]\) formulate the problem as a Mixed Integer Non-Linear Problem (MINLP) with a fixed offloading bandwidth for tasks. They reduced it to a Quadratically Constrained Quadratic Programming (QCQP) problem and apply semi-definite relaxation (SDR) to obtain optimal offloading decisions using optimization solvers. Zhan et al. \([77]\) formulate the decentralized offloading decision problem as a partially observable Markov decision process, and solve using gradient deep reinforcement learning based approach. Considering multi-tier architectures, Du et al. \([22]\) model the task offloading problem between different tiers as a min-cut problem, and solve the optimization problem using MAX-2SAT.

### Communication and computation contention

Studies in this category mainly focus on the task offloading and server provisioning problems with optimization objectives such as minimizing task response times \([9, 24, 27, 33, 37]\), makespan \([23, 28, 51, 83]\), device energy \([24, 54]\), server usage costs and communication overhead \([13, 23]\).

Heydari et al. \([33]\) consider the task offloading problem on a single-tier architecture. They formulate the problem as a Markov Decision Process and propose an actor-critic based reinforcement learning heuristic to learn the offloading decisions.

Some studies consider the server provisioning problem on single-tier \([23, 83]\) and multi-tier architectures \([27]\). Gao et al. \([27]\) formulate it as a Pure Integer Non-Linear Programming (PINLP) problem as well as a sub-divided Integer Non-Linear Programming (INLP) problem. They propose a lazy switch algorithm to control the task migration frequency between servers and use a solver for the INLP iteratively, providing a parameterized performance approximation bound. Duan et al. \([23]\) model tasks as a DAG and propose a decentralized online algorithm based on cooperative sequential games for the problem of allocating processors across servers to each DAG node, where the allocated bandwidth capacity is also proportional to the number of allocated processors. Zhao et al. \([83]\) formulate the makespan minimization problem as mixed-integer programming, and solve the optimization problem using hypergraph partitioning method for server provisioning and bandwidth allocation.

Some studies consider the combined task offloading and server provisioning problem on single-tier \([9, 37, 51, 54]\) and multi-tier architectures \([24, 13]\). Modeling task response times generically using server-specific utility functions, \([9]\) presents a decentralized max-consensus based greedy algorithm for the problem with a constant approxi-
mation bound of \((1 - 1/e)\) and shows polynomial-time convergence under some conditions on the utility function. On the other hand, Jošilo et al. [37] model the problem in a decentralized game-theoretic framework, and derive a policy with guaranteed convergence to a Nash equilibrium using Stackelberg games with a constant approximation bound of \((3 + \sqrt{5})/2\). Pang et al. [51] propose a heuristic using dynamic programming where the servers provision resources in proportion to the amount of resources requested in a decentralized manner by exchanging information on the tasks. Saleem et al. [54] formulate an MINLP optimization problem with energy constraints and propose a greedy heuristic to allocate communication resources based on tasks’ offloading bandwidth. Eshraghi and Liang [24] formulate a non-convex mixed-integer problem which is further reduced to a convex form with binary relaxation. They provide an optimal solution using a geometric programming that is iteratively applied on each processor of a multi-processor server. Chen et al. [13] formulate it as a QCQP problem and propose a heuristic combining SDR, alternating optimization and sequential tuning, and provide a lower bound on server usage cost.

Giroire et al. [28] consider the joint server provisioning and task scheduling problem on single-tier architectures. They model tasks as a DAG and propose a greedy list scheduling algorithm based on communication overhead that is optimal for tasks with constant response times and bounded bandwidth capacity. Further, they extend the solution with parameterized approximation algorithms using k-balanced (k-servers) partitioning for tasks with unbounded bandwidth capacity.

Only computation contention

In this category, studies mainly focus on the server provisioning and task scheduling problems with optimization objectives such as minimizing task response times [2, 8, 12, 19, 52, 56, 59, 61, 69, 70, 71, 80], makespan [76], device energy [72], server energy [36, 60, 78], server usage costs [32, 50] and communication overhead [32].

Some studies focus on VM and server provisioning problems on single-tier [2, 3, 8, 19, 35, 36, 55, 70, 78] and multi-tier [69] architectures. Abouaomar et al. [2] propose a matching game-based heuristic solution to identify servers for offloading using a decentralized deferred acceptance algorithm. Bao et al. [3] model the virtual network functions as DAGs and use graph pruning techniques to convert them to series-parallel graphs. Further, the optimization problem is solved using dynamic programming. Cao et al. [8] model the response time using an M/M/m queuing model where \(m\) is the number of servers, and solve the optimization problem using Lagrange multipliers and bisection methods for optimal server speed and workload arrival. Di and Wang [19] model the response time as a ratio of the task workload over its allocated resources, both abstracted with input parameters. The optimal resource allocation for each task is then determined using the Karush-Kuhn-Tucker (KKT) conditions in polynomial time. Modeling the response time as a function of the queuing delay on servers, [78] proposes a centralized online algorithm with a parameterized approximation bound, using integer relaxation to a linear programming (LP) problem and first-fit strategy to subsequently satisfy the integrality constraints. Jia et al. [35] model the makespan minimization problem using M/M/m queues, and propose a centralized heuristic solution based on min-cost max-flow problem and the Hungarian algorithm. They also propose a decentralized heuristic solution based on genetic algorithm. Considering DAG tasks, Shi et al. [55] model the response time using M/M/1 queuing model and propose a heuristic solution based on genetic algorithm for VM allocation. On the other hand, [36] models the response time as a function of the number of co-allocated VMs, and proposes a centralized online greedy algorithm with a parameterized approximation bound by sorting the VMs based on their arrival order. It also proposes a decentralized heuristic extension to this algorithm where each server performs a cost-benefit analysis comparing the cost of provisioning a VM alone to the incremental cost of provisioning that VM given the current provisions. Considering a single-tier architecture made up of interconnected access points, [70] proposes a graph representation method to solve the problem using capacitated k-median problem and derives parameterized approximation bounds. While considering multi-tier architectures, Xiao and Krunz [69] propose a decentralized strategy using Lagrange decomposition to transform the global provisioning problem into server-specific convex optimization problems. They also show that the proposed strategy converges to the global optimum at a rate inversely proportional to the number of iterations.

A few studies focus on the task scheduling problem on single-tier [60] and multi-tier [76] architectures. Tarplee et al. [60] formulate the problem as an ILP and solve using a relaxation method, where they assume tasks can be decomposed in chunks of arbitrary size to be run in parallel. They propose a heuristic solution based on the Convex Fill algorithm. Whereas, [76] uses an M/M/1 queuing model and formulates the problem as an MINLP. It decomposes the problem into sub-problems and proposes a heuristic solution to solve each sub-problem sequentially using LP relaxation.
Some studies consider the joint task offloading and server provisioning problem on single-tier [52, 72] and multi-tier [50] architectures. Ren et al. [52] formulate response-time minimization as a piece-wise convex function to determine the optimal proportion of each task to be executed on the device and the server with a fixed offloading bandwidth per task. For the special case of limited device computation capacity, the optimal length of Time Division Multiple Access (TDMA) slots is also computed for each device. Yaqub and Sorour [72] present a priority-based heuristic and bisection method for offloading decisions on neighboring devices and servers, respectively. They propose a heuristic solution for the provisioning problem using the Lagrangian method. Ouyang et al. [50] propose an offline solution using the shortest path algorithm for DAG tasks. It also presents an online learning algorithm for provisioning using multi-arm bandit with a parameterized regret bound.

Some studies focus on both server provisioning and task scheduling problems on single-tier [12, 56, 65, 80] and multi-tier [59, 61] architectures. Considering max-min fairness, which maximizes the minimum resource allocation across tasks sharing servers, Chen et al. [12] reduce the optimization problem to an LP for a single task case and find the optimal solution. For multiple tasks, they iterate the procedure to ensure max-min fairness. Considering DAG tasks, Shu et al. [56] propose an FPTAS for the makespan minimization problem through a reduction to the constrained shortest path problem for single-resource VMs. For the more general case of multi-resource VMs, they propose a greedy heuristic based on critical paths and binary search. Wan et al. [65] model the response time using M/M/m queueing model and solve the optimization problem using optimal stopping theory and particle swarm optimization. Zhang et al. [80] present a priority-based weighted algorithm for provisioning with a constant approximation bound of 2 in terms of the number of servers and a heuristic scheduling algorithm based on the Karmarkar-Karp differencing algorithm. Tong et al. [61] consider fractional resource allocations with a fixed offloading bandwidth per task. For the special case of one server per tier of the architecture, they present optimal centralized solutions using convex optimization and branch-and-bound methods, whereas, for the more general problem, they present a solution based on simulated annealing. Tan et al. [59] propose a decentralized solution by selecting the server with the least increase in response time and schedule using the shortest remaining computation time first policy. They prove this to be $O(1/\epsilon)$-competitive (in terms of response time) with a corresponding constant approximation bound of $1 + \epsilon$ on the speed of servers.

A few studies consider the joint problem of task offloading, server provisioning and task scheduling on single-tier [32, 71] architectures. Considering DAG tasks, Han et al. [32] present a priority-based heuristic solution where tasks are sorted based on total average computation and communication time. Considering sequential tasks with a constraint on the number of tasks allocated per server, Yang et al. [71] propose a greedy heuristic in which tasks are first offloaded to the server without any resource constraint and later to meet the constraint some tasks are moved back based on a reward function.

### Only communication contention

In this category, works focus on task offloading and bandwidth provisioning problem with optimization objectives such as minimizing device energy [11, 41, 43] and server energy [11, 41].

Chen et al. [11] model the bandwidth as a function of the interference among tasks in the wireless network. They model the problem in a decentralized game-theoretic framework to minimize both task response time and makespan on single-tier architectures. They derive a policy using potential games with finite improvement property with guaranteed convergence to a Nash equilibrium and a parameterized approximation bound. Mao et al. [43] formulate it as a stochastic optimization problem for multi-tier architectures. They propose a Lyapunov optimization-based algorithm and use the Lagrangian method and KKT conditions to determine the optimal device power and offloading bandwidth. On the other hand, Liu et al. [41] consider only the task offloading problem on a single-tier architecture. They derive a heuristic policy using population games, where player strategies are modeled using a Markov evolutionary process.

### 3.2 Deadline constrained

Most time-critical tasks request for resources with a notion of a deadline. In this section, we assume that the deadline defines a requirement on the task’s response time which includes both computation and communication times, unless specified otherwise. We present all studies that consider workload tasks with such deadlines, irrespective of the optimization objective they address.
No contention

In this category, works mainly focus on the server provisioning and/or task scheduling problems with optimization objectives such as minimizing task response times [15, 48], device energy [31, 38] and task deadline misses [79]. A few studies only consider the server provisioning [15, 79] problem on single-tier [15] and multi-tier architectures [79]. Chen et al. [15] additionally consider a greedy task replication strategy for fault tolerance and propose a multi-arm bandit learning algorithm with a parameterized approximation bound for sub-modular marginal reward functions (reward is based on a probabilistic prediction of task completion times). Zhang et al. [79] use a singleton weighted congestion game based heuristic to arrive at a consensus on task allocation at the lower tier. They also use a stochastic Lyapunov optimization-based greedy heuristic to estimate task response times and decide whether to admit the task or to provision it on another server at the higher tier.

Some studies consider the server provisioning and task scheduling problems on single-tier [38, 31] and multi-tier [48] architectures. Considering a set of task flows allocated on a resource graph where each flow is a sequence of sub-tasks with an end-to-end deadline, Millnert et al. [48] present a centralized analytical technique for dynamic adjustments to the response times experienced by tasks. They propose protocols that use an upper bound on the rate of change of response times which would ensure the satisfaction of all end-to-end deadlines. They present protocols for dynamically changing task flows as well as resource graphs. On the other hand, considering tasks modeled as a collection of trees with end-to-end deadlines and fixed offloading bandwidth, [38] presents a centralized dynamic programming based polynomial-time solution using time quantization, and an exponential-time extension for tasks with probabilistic computation times. Guo et al. [31] formulate the convex optimization problem as a three-stage flow-shop scheduling problem by separately considering the offloading, constant execution and downloading duration of each task. They solve the problem optimally when the minimum offloading duration is larger than the maximum execution duration of all tasks using KKT conditions and bisection search method.

Communication and computation contention

Studies in this category mainly focus on server provisioning and task scheduling problems with optimization objectives such as minimizing VM delays [17, 47], task deadline misses [45, 46], device energy [63, 64] and maximizing task utility [82]. Some studies consider the problem of server provisioning for single-tier [17] and multi-tier [47] architectures with re-provisioning for changes in the device coverage area. Cziva et al. [17] model the resources of the servers with bounded bandwidth capacity and communication delay, and propose a technique using Optimal Stopping theory. Millnert et al. [47] consider task flows pre-allocated on a resource graph with end-to-end deadlines as in [48], and present a decentralized heuristic solution through deadline decomposition based on control theoretic and optimization frameworks to reduce control theoretic and optimization frameworks to reduce VM creation delays.

A few studies consider the task offloading and server provisioning problem on multi-tier [63, 64] architectures. Vu et al. [63] formulate the problem as an MINLP, and use integer relaxation and branch and bound algorithm to find an optimal solution and prune the search space. They extend this in [64] with additional parameters such as offloading and downloading bandwidth. They propose a decentralized heuristic algorithm through decomposition using the bender’s cuts.

A few works focus on the task provisioning and scheduling problems on single-tier [82] and multi-tier architectures [45, 46]. Zheng and Shroff [82] propose an online algorithm for stochastic tasks in the continuous and discrete-time domain with a competitive ratio of 2 and 1.8, respectively. On the other hand, [45, 46] proposes an online heuristic based on the largest computation time to reduce the number of deadline misses and derives a parameterized competitive ratio on the makespan.

Only computation contention

In this category, studies mainly focus on the server provisioning and task scheduling problems with optimization objectives such as minimizing the task response times [16, 18, 21, 40, 66, 81], device energy [10, 21], server energy [14, 29, 75], server usage costs [6, 7, 25, 42, 44, 53, 58, 68, 73], peak resource utilization on servers [34, 67], task deadline misses [4, 5, 44, 84] and communication overhead [58].

Some studies focus on both server provisioning and task scheduling problems on single-tier [4, 5, 66, 73, 84] or multi-tier [25, 58] architectures. Considering a variety of different objectives, they propose heuristic solutions using techniques such as prioritization based on task parameters with best-fit provisioning [4, 5], agent-based decentralized bidding between tasks and server VMs based on task parameters [84], as early as possible scheduling with load balancing [66], ant colony optimization with a response time dependent utility function [25], and a
discretization strategy that combines the provisioning results of a convex optimization solver with greedy deadline-driven scheduling [58]. Wang et al. [66] also consider fault tolerance using backup tasks that are executed as late as possible with their allocations being reclaimed when not required. Yin et al. [73] formulate it as an LP relaxation and solve using dual decomposition with an online algorithm that has a parameterized competitive ratio in terms of resource capacity.

Some studies only focus on the server provisioning problem on single-tier [14, 29, 40, 44, 67, 81] and multi-tier architectures [42]. Again considering a variety of different objectives, they either present analytical [14, 29, 40, 44] or heuristic [67, 42, 81] solutions. Chen et al. [14] consider a demand-response setting that enforces a maximum peak power for each server. They present an online solution with a parameterized approximation bound using Vickrey-Clark-Groves (VCG) auctions and also consider the trade-off between switching costs and energy loss for server activations and deactivations. Gu et al. [29] and Liu et al. [40] formulate MINLP problems and optimally solve relaxed duals using either block coordinate descent method [40] or Lagrangian with a dynamic voltage and frequency scaling strategy [29]. Considering a $M/M/m$ queuing model, Mei et al. [44] optimally solve the problem using bisection method assuming the number of servers $m$ and the speed of each server are continuous variables. Then, they recover integer values for these variables with the least server usage costs. Considering a bound on VM allocation delay, Wei et al. [67] propose an online greedy heuristic strategy based on balancing the remaining resource capacities across servers with future workload predictions modeled as a Markov chain that uses moving averages [67]. Ma et al. [42] model the costs separately for on-demand and reserved resource provisioning on servers, and present heuristics based on gradient descent, bisection method and piece-wise convex optimization to provision reserved, on-demand and both the resources, respectively. Zhang et al. [81] model the queuing delay, network delay and processing time as costs and propose a decentralized game-theoretic solution based on the Stackelberg game to minimize the response time of tasks.

A few studies consider either the task offloading problem [10], the joint task offloading and server provisioning problem [18, 21] or the task scheduling problem [75] on single-tier [10, 18, 75] and multi-tier [21] architectures. Chang et al. [10] use queuing theory and show that the presented centralized solution is guaranteed to converge to the optimal value because the objective function is quasi-convex. They also propose a decentralized heuristic that uses Lagrange decomposition and transforms the global problem into device-specific relaxed convex optimization problems. Dai et al. [18] consider fixed offloading bandwidth for tasks and iteratively solve the joint problem as an MINLP, where the offloading problem is relaxed to a real-valued NLP and solved using bipartite graph-based rounding method with a parameterized approximation bound, and the provisioning problem is solved optimally using Lagrangian multipliers with a gradient descent method. Du et al. [21] minimize the weighted sum of task response time and device energy with a fixed offloading bandwidth for tasks. They formulate it as a QCQP and reduce it to a convex problem using SDR and use the bisection method to determine the offloading decisions. They present a sub-optimal power and offloading bandwidth allocation algorithm using Lagrange multipliers. Finally, Yu et al. [75] model energy costs as battery losses. They transform the problem into a queue stability problem using the framework of Lyapunov optimization and present an algorithm for task and battery scheduling with a parameterized approximation bound, where admission control is performed based on the available server capacity.

Some studies model the workload with a DAG and end-to-end deadline constraint on the DAG [6, 7, 16, 34, 53, 68]. Focusing on both server provisioning and DAG scheduling problems on single-tier architectures, studies present heuristic solutions using particle swarm optimization [53] and greedy deadline decomposition and scheduling strategies based on slowest-cheapest VMs and earliest ready tasks with fixed offloading bandwidth [6]. Extensions to handle variations in the computation and communication times using task replication and critical path detection have also been proposed [7]. Note, in these studies, although the scheduling problem uses a contention model for computation time, the provisioning problem is modeled without contention by allowing for an arbitrary number of VM instantiations. Other studies only consider the DAG scheduling problem on single-tier architectures, and propose deadline decomposition-based heuristic solutions [34, 68]. Hu et al. [34] use LP by converting the DAG to a set of independent task groups with deadlines decomposed in proportion to the number and computation time of tasks in each group. Whereas, Wu et al. [68] use probabilistic list scheduling with tasks ordered using ant colony optimization and deadlines decomposed based on critical paths. Chen et al. [16] propose a statistical model checking based framework to evaluate resource allocation strategies and present a supervised learning-based heuristic to solve the problem.
Only communication contention

Studies in this category mainly focus on task offloading and server/bandwidth provisioning problems with optimization objectives such as minimizing server energy [57], device energy [49], server usage costs and communication overhead [26, 30, 74].

Some studies consider the task offloading problem [30], bandwidth provisioning problem [57] and joint task offloading and server provisioning problem [74] on single-tier architectures. Guo et al. [30] model the tasks as DAGs and formulate the problem as a non-convex problem. They relax it and optimally solve its dual problem using Lagrangian multipliers and sub-gradient method. Considering a bound on queuing delay, Sun et al. [57] derive a probability function for deadline misses and use interior point method to find the optimal solution. Yu et al. [74] formulate the problem as a multi-commodity max-flow problem and propose an FPTAS assuming tasks can be arbitrarily parallelized. They also propose a randomized algorithm with a parameterized approximation bound for tasks that are not parallelizable. Considering multi-tier architectures, Nguyen et al. [49] formulate the joint problem as a min-max INLP and use the bisection search method to compute the optimal device frequency and wireless channel assignment. They also present a low-complexity heuristic solution using decoupled ILP based optimization.

Tong and Gao [62] only consider the wireless network scheduling problem on single-tier architectures. They propose a dynamic programming solution, for a burst of transmissions, by computing the optimal delay in task communications. Gao et al. [26] focus on the joint task offloading, server provisioning and task scheduling problem on a multi-tier architecture with a bound on communication delay. They propose a greedy offline algorithm based on a task-specific utility function and an opportunistic online algorithm in which tasks offload in the first convenient slot they find, both with an approximation bound of 2.

4 Summary and future research directions

We consolidated the literature based on our proposed taxonomy in Table 2. As seen, with respect to the timing model, there are sufficient studies for both response time minimization and deadline constrained problems. However, there are limited works that minimize makespan. This is reasonable as makespan minimization is, in general, a harder problem to solve as the complexity is higher due to the inherent min-max optimization. In terms of contention, most contributions are on only computation contention and relatively fewer contributions consider both computation and communication contention. Note that, the literature on no contention forms an interesting body of work since they mainly consider multi-objective optimization such as energy-delay trade-offs. From the perspective of problem classes, we find that there are very few studies that investigate all three problem classes combined: offloading, provisioning and scheduling. The existing literature primarily focuses on centralized solutions and there is little focus on decentralization. With the rapid proliferation of decentralized IoT applications such as remote/crowd sensing, fleet management, smart cities, etc. there is a need for more decentralized solutions. Further, among the decentralized solutions, very few works considered the deadline constrained timing model. Most of which are heuristic solutions and do not have any guarantees. As a result, time-critical applications cannot be hosted on cloud platforms without guaranteed analytical algorithms. Considering the problem classes, most decentralized studies either focus on offloading or provisioning problem and the remaining combination of classes are relatively unexplored.

An overview of how time-related model parameters (see Section 2) are used in the literature is presented in Table 3. As seen, there are fewer contributions towards multi-tier architectures. Only a few papers model queues on servers by considering serving rate and arrival rate of tasks. Lack of queuing models make it harder to address the multi-tier architecture problems. Most existing works assume the computation time of tasks on servers is known a priori, which may not be realistic. Finally, it can also be seen that compared to computation resource modeling, communication resources are relatively less explored in the literature. Observe that only those papers that model bandwidth capacity have communication contention and those that bound the computation resources either in the form of computation capacity or serving rate have computation contention. Even among those that consider bandwidth capacity, the capacity is restricted to communication bandwidth between devices and servers and not between servers. Another major timing related model parameter is the device-server delay and server-server delay. As can be seen, very few studies (both single-tier and multi-tier architecture) consider them in their model and not all of them consider both together. However, in realistic systems, both these delays would be present. Modeling them both increases the complexity of the problem and hence, there are limited studies in the area.
Table 3: Literature classification based on timing related model parameters.

| Cloud Architecture | Server Parameters | Task Parameters | Delay Parameters |
|--------------------|-------------------|----------------|-----------------|
|                    | Computation       | Arrival rate   | Device-Server   |
|                    | capacity          | Duration/Execution time |             |
|                    | Bandwidth         | Deadline constraint |             |
|                    | capacity          | Offloading bandwidth |             |
|                    | Serving rate      |                | Server-Server   |
| Single-tier        | [2, 4, 5, 6, 9, 12, 14, 16, 17, 18, 19, 23, 28, 29, 32, 33, 34, 36, 37, 40, 41, 51, 56, 60, 65, 66, 67, 68, 70, 71, 72, 73, 75, 78, 77, 80, 82, 83, 84] | [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 18, 19, 20, 23, 28, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 43, 51, 52, 54, 57, 62, 68, 70, 71, 72, 73, 75, 77, 81, 82, 84] | [2, 6, 9, 10, 11, 15, 17, 18, 20, 23, 28, 30, 31, 32, 33, 37, 38, 40, 44, 53, 57, 62, 66, 67, 73, 75, 74, 81, 82, 84] |
|                    | [6, 9, 11, 12, 17, 18, 20, 23, 28, 30, 37, 38, 33, 41, 43, 51, 54, 57, 62, 74, 77, 83, 82] | [8, 10, 29, 35, 44, 55, 57, 65, 67, 75, 78, 80, 81] | [4, 5, 6, 7, 10, 14, 15, 16, 17, 18, 29, 30, 31, 34, 38, 40, 44, 53, 57, 62, 66, 67, 68, 73, 75, 74, 81, 82, 84] |
| Multi-tier         | [13, 21, 24, 25, 27, 45, 46, 47, 50, 58, 59, 61, 63, 64] | [27, 42, 47, 69, 76] | [21, 25, 26, 42, 45, 46, 47, 48, 49, 58, 63, 64, 79] |
|                    | [13, 21, 24, 25, 26, 27, 45, 46, 47, 49, 61, 63, 64] | [27, 42, 47, 69, 76] | [22, 13, 24, 25, 27, 45, 46, 49, 61, 63, 64, 79] |
|                    | [27, 42, 47, 69, 76] | [22, 13, 21, 24, 25, 45, 46, 49, 50, 58, 59, 61, 63, 64, 79] | [13, 21, 24, 25, 27, 45, 46, 49, 61, 63, 64, 79] |
|                    | [21, 25, 26, 42, 45, 46, 47, 48, 49, 58, 63, 64, 79] | [22, 13, 24, 25, 27, 45, 46, 49, 61, 63, 64, 79] | [13, 21, 24, 25, 27, 45, 46, 49, 58, 63, 69, 76] |
|                    | [22, 13, 21, 24, 25, 45, 46, 49, 58, 63, 69, 76] | [13, 21, 24, 25, 27, 45, 46, 49, 58, 63, 69, 76] | [22, 13, 21, 25, 45, 46, 50, 58, 63, 69, 76] |
Table 2 and Table 3 individually classify the studies based on the problem classes and model parameters respectively. These two tables can be used together to further classify the literature and identify open research problems and closely related works. If one decides to start working on makespan minimization problems with multi-tiered architecture, considering the queuing model, then the two tables come in handy to determine the list of closely related works that can potentially be extended. Comparing across these two tables, we see works in both computation and communication contention category are majorly on single-tier architectures. Many multi-tier architecture works ignore the delay between servers. All contributions based on queuing theory consider only computation contention and provide only heuristic solutions. Interestingly, no surveyed work modeled queues and provided a decentralized solution.

From the literature, we observed that certain assumptions on problem classes and solution types leave some open problems. As discussed before, decentralized solutions with deadline constrained model are generally lacking. With the growth of decentralization in IoT applications, this is one potential research problem that needs to be addressed in the near future. Another important aspect to note is that most studies assume zero latency for the downlink data transfer (transmission of results from the servers to the devices). However, this assumption is unrealistic as certain AI applications (such as image/video search) have large data to be sent back to the devices. Although 5G technology offers higher downloading bandwidth, multiple tasks could contend for this bandwidth increasing the task response times. Another aspect that has recently surfaced in this domain is the dynamic availability of resources and mobility of devices. These two new features would introduce more complexity and challenges in coming up with a solution.

**Funding:** This work was financially supported in part by the Singapore National Research Foundation under its Campus for Research Excellence And Technological Enterprise (CREATE) programme.

**References**

1. 5G for business: a 2030 market compass [Ericsson Report]. Retrieved March 14, 2020 from https://www.ericsson.com/ea8e35/assets/local/5g/the-5g-for-business-a-2030-compass-report-2019.pdf.

2. A. Abouaomar, A. Kobbane and S. Cherkaoui. *Matching-Game for User-Fog Assignment*. In: IEEE Global Communications Conference, 2018.

3. W. Bao, D. Yuan, B. B. Zhou and A. Y. Zomaya. *Prune and Plant: Efficient Placement and Parallelism of Virtual Network Functions*. IEEE Transactions on Computers, 2020.

4. R. Begam, H. Moradi, W. Wang and D. Zhu. *Flexible VM Provisioning for Time-Sensitive Applications with Multiple Execution Options*. In: IEEE International Conference on Cloud Computing, 2018.

5. R. Begam, W. Wang and D. Zhu. *TIMER-Cloud: Time-Sensitive VM Provisioning in Resource-Constrained Clouds*. IEEE Transactions on Cloud Computing, 2020.

6. Z. Cai, X. Li and R. Ruiz. *Resource Provisioning for Task-Batch Based Workflows with Deadlines in Public Clouds*. IEEE Transactions on Cloud Computing, 2019.

7. R. N. Calheiros and R. Buyya. *Meeting Deadlines of Scientific Workflows in Public Clouds with Tasks Replication*. IEEE Transactions on Parallel and Distributed Systems, 2014.

8. J. Cao, K. Li and I. Stoimenovic. *Optimal Power Allocation and Load Distribution for Multiple Heterogeneous Multicore Server Processors across Clouds and Data Centers*. IEEE Transactions on Computers, 2014.

9. G. Castellano and F. Esposito and F. Risso. *A Distributed Orchestration Algorithm for Edge Computing Resources with Guarantees*. In: IEEE International Conference on Computer Communications, 2019.

10. Z. Chang, Z. Zhou, T. Ristaniemi and Z. Niu. *Energy Efficient Optimization for Computation Offloading in Fog Computing System*. In: IEEE Global Communications Conference, 2017.

11. X. Chen, L. Jiao, W. Li and X. Fu. *Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing*. IEEE/ACM Transactions on Networking, 2016.

12. L. Chen, S. Liu, B. Li and B. Li. *Scheduling jobs across geo-distributed datacenters with max-min fairness*. In: IEEE Conference on Computer Communications, 2017.

13. M. Chen, B. Liang and M. Dong. *Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point*. In: IEEE Conference on Computer Communications, 2017.

14. S. Chen, L. Jiao, L. Wang and F. Liu. *An Online Market Mechanism for Edge Emergency Demand Response via Cloudlet Control*. In: IEEE Conference on Computer Communications, 2019.

15. L. Chen and J. Xu. *Task Replication for Vehicular Cloud: Contextual Combinatorial Bandit with Delayed Feedback*. In: IEEE Conference on Computer Communications, 2019.

16. M. Chen, S. Huang, X. Fu, X. Liu and J. He. *Statistical Model Checking-Based Evaluation and Optimization for Cloud Workflow Resource Allocation*. IEEE Transactions on Cloud Computing, 2020.

17. R. Cziva, C. Anagnostopoulos and D. P. Pezaros. *Dynamic, Latency-Optimal vNF Placement at the Network Edge*. In: IEEE Conference on Computer Communications, 2018.

18. Y. Dai, D. Xu, S. Maharjan and Y. Zhang. *Joint Offloading and Resource Allocation in Vehicular Edge Computing and Networks*. In: IEEE Global Communications Conference, 2018.

19. S. Di and C. Wang. *Dynamic Optimization of Multiatribute Resource Allocation in Self-Organizing Clouds*. IEEE Transactions on Parallel and Distributed Systems, 2013.
20. C. Ding, J. Wang, M. Cheng, C. Chang, J. Wang and M. Lin. *Joint Beamforming and Computation Offloading for Multi-User Mobile-Edge Computing*. In: IEEE Global Communications Conference, 2019.

21. J. Du, L. Zhao, J. Feng and X. Chu. *Computation Offloading and Resource Allocation in Mixed Fog/Cloud Computing Systems with Min-Max Fairness Guarantee*. IEEE Transactions on Communications, 2018.

22. M. Du, Y. Wang, K. Ye and C. Xu. *Algorithms of Cost-Driven Computation Offloading in the Edge-Cloud Environment*. IEEE Transactions on Computers, 2020.

23. R. Duan, R. Prodan and X. Li. *Multi-Objective Game Theoretic Scheduling of Bag-of-Tasks Workflows on Hybrid Clouds*. IEEE Transactions on Cloud Computing, 2014.

24. N. Eshraghi and B. Liang. *Joint Offloading Decision and Resource Allocation with Uncertain Task Computing Requirement*. In: IEEE Conference on Computer Communications, 2019.

25. J. Fan, X. Wei, T. Wang, T. Lan and S. Subramaniam. *Deadline-Aware Task Scheduling in a Tiered IoT Infrastructure*. In: IEEE Global Communications Conference, 2017.

26. G. Gao, M. Xiao, J. Wu, K. Han, L. Huang and Z. Zhao. *Opportunistic Mobile Data Offloading with Deadline Constraints*. IEEE Transactions on Parallel and Distributed Systems, 2017.

27. B. Gao, Z. Zhou, F. Liu and F. Xu. *Winning at the Starting Line: Joint Network Selection and Service Placement for Mobile Edge Computing*. In: IEEE Conference on Computer Communications, 2019.

28. F. Giroire, N. Huin, A. Tomassilli and S. Pérennes. *When Network Matters: Data Center Scheduling with Network Tasks*. In: IEEE Conference on Computer Communications, 2019.

29. L. Gu, D. Zeng, A. Barnawi, S. Guo and I. Stoimenovic. *Optimal Task Placement with QoS Constraints in Geo-Distributed Data Centers Using DVFS*. IEEE Transactions on Computers, 2015.

30. S. Guo, B. Xiao, Y. Yang and Y. Yang. *Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing*. In: IEEE International Conference on Computer Communications, 2016.

31. J. Guo, Z. Song, Y. Cui, Z. Liu and Y. Ji. *Energy-Efficient Resource Allocation for Multi-User Mobile Edge Computing*. In: IEEE Global Communications Conference, 2017.

32. Y. Han, Z. Zhao, J. Mo, C. Shu and G. Min. *Efficient Task Offloading with Dependency Guarantees in Ultra-Dense Edge Networks*. In: IEEE Global Communications Conference, 2019.

33. J. Heydari, V. Ganapathy and M. Shah. *Dynamic Task Offloading in Multi-Agent Mobile Edge Computing*. In: IEEE Global Communications Conference, 2019.

34. Z. Hu, B. Li, C. Chen and X. Ke. *FlowTime: Dynamic Scheduling of Deadline-Aware Workflows and Ad-Hoc Jobs*. In: IEEE International Conference on Distributed Computing Systems, 2018.

35. M. Jia, W. Liang, Z. Xu, M. Huang and Y. Ma, *QoS-Aware Cloudlet Load Balancing in Wireless Metropolitan Area Networks*. IEEE Transactions on Cloud Computing, 2020.

36. X. Jin, F. Zhang, L. Wang, S. Hu, B. Zhou and Z. Liu. *Joint Optimization of Operational Cost and Performance Interference in Cloud Data Centers*. IEEE Transactions on Cloud Computing, 2017.

37. S. Josiño and G. Dan. *Wireless and Computing Resource Allocation for Selfish Computation Offloading in Edge Computing*. In: IEEE Conference on Computer Communications, 2019.

38. Y. Kao and B. Krishnamachari. *Optimizing mobile computational offloading with delay constraints*. In: IEEE Global Communications Conference, 2014.

39. Y. Kao, B. Krishnamachari, M. Ra and F. Bai. *Hermes: Latency optimal task assignment for resource-constrained mobile computing*. In: IEEE Conference on Computer Communications, 2015.

40. Q. Liu, T. Han and N. Ansari. *Joint Radio and Computation Resource Management for Low Latency Mobile Edge Computing*. In: IEEE Global Communications Conference, 2018.

41. D. Liu, A. Hafid and L. Khoukhi. *Population Game Based Energy and Time Aware Task Offloading for Large Amounts of Competing Users*. In: IEEE Global Communications Conference, 2018.

42. X. Ma, S. Wang, S. Zhang, P. Yang, C. Lin and X. S. Shen. *Cost-Efficient Resource Provisioning for Dynamic Requests in Cloud Assisted Mobile Edge Computing*. IEEE Transactions on Cloud Computing, 2019.

43. Y. Mao, J. Zhang, S. H. Song and K. B. Letaief. *Power-Delay Tradeoff in Multi-User Mobile-Edge Computing Systems*. In: IEEE Global Communications Conference, 2016.

44. J. Mei, K. Li, A. Ouyang and K. Li. *A Profit Maximization Scheme with Guaranteed Quality of Service in Cloud Computing*. IEEE Transactions on Computers, 2015.

45. J. Meng, H. Tan, C. Xu, W. Cao, L. Liu and B. Li. *Deadline-Aware Task Scheduling in a Tiered IoT Infrastructure*. IEEE Transactions on Computer Communications, 2019.

46. J. Meng, H. Tan, X. Li, Z. Han and B. Li. *Online Deadline-Aware Task Dispatching and Scheduling in Edge Computing*. IEEE Transactions on Parallel and Distributed Systems, 2020.

47. V. Millnert, J. Eker and E. Bini. *Achieving Predictable and Low End-to-End Latency for a Network of Smart Services*. In: IEEE Global Communications Conference, 2018.

48. V. Millnert, J. Eker and E. Bini. *End-To-End Deadlines over Dynamic Topologies*. In: Euromicro Conference on Real-Time Systems, 2019.

49. T. T. Nguyen and B. L. Long. *Joint Computation Offloading and Resource Allocation in Cloud Based Wireless HetNets 2017*. In: IEEE Global Communications Conference, 2017.

50. T. Ouyang, R. Li, X. Chen, Z. Zhou and X. Tang. *Adaptive User-managed Service Placement for Mobile Edge Computing: An Online Learning Approach*. In: IEEE Conference on Computer Communications, 2019.

51. A. Pang, W. Chung, T. Chiu and J. Zhang. *Latency-Driven Cooperative Task Computing in Multi-user Fog-Radio Access Networks*. In: International Conference on Distributed Computing Systems, 2017.

52. J. Ren, G. Yu, Y. Cai, Y. He and F. Qu. *Partial Offloading for Latency Minimization in Mobile-Edge Computing*. In: IEEE Global Communications Conference, 2017.

53. M. A. Rodriguez and R. Buyya. *Deadline Based Resource Provisioning and Scheduling Algorithm for Scientific Workflows on Clouds*. IEEE Transactions on Cloud Computing, 2014.

54. U. Saleem, Y. Liu, S. Jangsher and Y. Li. *Performance Guaranteed Partial Offloading for Mobile Edge Computing*. In: IEEE Global Communications Conference, 2018.

55. T. Shi, H. Ma, G. Chen and S. Hartmann. *Location-Aware and Budget-Constrained Service Deployment for Composite
Applications in Multi-Cloud Environment. IEEE Transactions on Parallel and Distributed Systems, 2020.
56. T. Shu and C. Q. Wu. Performance Optimization of Hadoop Workflows in Public Clouds through Adaptive Task Partitioning. In: IEEE Conference on Computer Communications, 2017.
57. C. Sun, C. She and C. Yang. Energy-Efficient Resource Allocation for Ultra-Reliable and Low-Latency Communications. In: IEEE Global Communications Conference, 2017.
58. S. Sundar and B. Liang. Offloading Dependent Tasks with Communication Delay and Deadline Constraint. In: IEEE Conference on Computer Communications, 2018.
59. H. Tan, Z. Han, X. Li and F. C. M. Lau. Online job dispatching and scheduling in edge-clouds. In: IEEE Conference on Computer Communications, 2017.
60. K. M. Tarplee, R. Friese, A. A. Maciejewski, H. J. Siegel and E. K. P. Chong. Energy and Makespan Tradeoffs in Heterogeneous Computing Systems using Efficient Linear Programming Techniques. IEEE Transactions on Parallel and Distributed Systems, 2016.
61. L. Tong and Y. Li and W. Gao. A Hierarchical Edge Cloud Architecture for Mobile Computing. In: IEEE International Conference on Computer Communications, 2016.
62. L. Tong and W. Gao. Application-aware traffic scheduling for workload offloading in mobile clouds. In: IEEE International Conference on Computer Communications, 2016.
63. T. T. Vu, N. V. Huynh, D. T. Hoang, D. N. Nguyen and E. Dutkiewicz. Offloading Energy Efficiency with Delay Constraint for Cooperative Mobile Edge Computing Networks. In: IEEE Global Communications Conference, 2018.
64. T. T. Vu, D. N. Nguyen, D. T. Hoang and E. Dutkiewicz. QoS-Aware Fog Computing Resource Allocation Using Feasibility-Finding Benders Decomposition. In: IEEE Global Communications Conference, 2019.
65. B. Wan, J. Dang, Z. Li, H. Gong, F. Zhang and S. Oh. Modeling Analysis and Cost-Performance Ratio Optimization of Virtual Machine Scheduling in Cloud Computing. IEEE Transactions on Parallel and Distributed Systems, 2020.
66. J. Wang, W. Bao, X. Zhu, L. T. Yang and Y. Xiang. FESTAL: Fault-Tolerant Elastic Scheduling Algorithm for Real-Time Tasks in Virtualized Clouds. IEEE Transactions on Computers, 2015.
67. L. Wei, C. H. Foh, B. He and J. Cai. Towards Efficient Resource Allocation for Heterogeneous Workloads in IAaaS Clouds. IEEE Transactions on Cloud Computing, 2018.
68. Q. Wu, F. Ishikawa, Q. Zhu, Y. Xia and J. Wen. Deadline-Constrained Cost Optimization Approaches for Workflow Scheduling in Clouds. IEEE Transactions on Parallel and Distributed Systems, 2017.
69. Y. Xiao and M. Krunz. QoE and power efficiency tradeoff for fog computing networks with fog node cooperation. In: IEEE Conference on Computer Communications, 2017.
70. Z. Xu, W. Liang, W. Xu, M. Jia and S. Guo. Efficient Algorithms for Capacitated Cloudlet Placements. IEEE Transactions on Parallel and Distributed Systems, 2016.
71. L. Yang, J. Cao, H. Cheng and Y. Ji. Multi-User Computation Partitioning for Latency Sensitive Mobile Cloud Applications. IEEE Transactions on Computers, 2015.
72. U. Yaqub and S. Sameh. Multi-Objective Resource Optimization for Hierarchical Mobile Edge Computing. In: IEEE Global Communications Conference, 2018.
73. B. Yin, Y. Cheng, L. X. Cai and X. Cao. Online SLA-Aware Multi-Resource Allocation for Deadline Sensitive Jobs in Edge-Clouds. In: IEEE Global Communications Conference, 2017.
74. R. Yu, G. Xue and X. Zhang. Application Provisioning in FOG Computing-enabled Internet-of-Things: A Network Perspective. In: IEEE Conference on Computer Communications, 2018.
75. L. Yu, T. Jiang, Y. Cao and Q. Qi. Joint Workload and Battery Scheduling with Heterogeneous Service Delay Guarantees for Data Center Energy Cost Minimization. IEEE Transactions on Parallel and Distributed Systems, 2015.
76. D. Zeng, L. Gu, S. Guo, Z. Cheng and S. Yu. Joint Optimization of Task Scheduling and Image Placement in Fog Computing Supported Software-Defined Embedded System. IEEE Transactions on Computers, 2016.
77. Y. Zhan, S. Guo, P. Li and J. Zhang. A Deep Reinforcement Learning Based Offloading Game in Edge Computing. IEEE Transactions on Computers, 2020.
78. Q. Zhang, M. Zhan, R. Boutaba and J. Hellerstein. Dynamic Heterogeneity-Aware Resource Provisioning in the Cloud. IEEE Transactions on Cloud Computing, 2014.
79. D. Y. Zhang and D. Wang. An Integrated Top-down and Bottom-up Task Allocation Approach in Social Sensing based Edge Computing Systems. In: IEEE Conference on Computer Communications, 2019.
80. Q. Zhang, Y. Xiao, F. Liu, J. C. S. Lui, J. Guo and T. Wang. Joint Optimization of Chain Placement and Request Scheduling for Network Function Virtualization. In: IEEE International Conference on Distributed Computing Systems, 2017.
81. H. Zhang, Y. Xiao, S. Bu, F. R. Yu, D. Niyato and Z. Han. Distributed Resource Allocation for Data Center Networks: A Hierarchical Game Approach. IEEE Transactions on Cloud Computing, 2020.
82. Z. Zheng and N. B. Shroff. Online multi-resource allocation for deadline sensitive jobs with partial values in the cloud. In: IEEE International Conference on Computer Communications, 2016.
83. L. Zhao, Y. Yang, A. Munir, A. X. Liu, Y. Li and W. Qu. Optimizing Geo-Distributed Data Analytics with Coordinated Task Scheduling and Routing. IEEE Transactions on Parallel and Distributed Systems, 2020.
84. X. Zhu, C. Chen, L. T. Yang and Y. Xiang. ANGEL: Agent-Based Scheduling for Real-Time Tasks in Virtualized Clouds. IEEE Transactions on Parallel and Distributed Systems, 2015.

Bionotes

Saravanan Ramanathan
TUMCREATE, 1 Create Way, #10-02 CREATE Tower, Singapore 138602, Singapore saravanan.ramanathan@tum-create.eu.sg

Dr. Saravanan Ramanathan received the Ph. D. degree in Computer Science and Engineering from Nanyang Technological University (NTU), Singapore, in 2019. Currently, he is a research fellow at TUMCREATE, Singapore. His research interests include real-time cyberphysical systems and energy-efficient internet of things.
Nitin Shivaraman is a research associate at TUMCREATE and is currently pursuing a Joint-Ph. D. degree at Technische Universität München (TUM), Germany and NTU. He received the M. Sc. degree in Embedded Systems from NTU in 2013. His research focuses on communication strategies for power-efficiency and fault-tolerance in IoT networks.

Seima Suryasekaran is a project officer at TUMCREATE and is currently pursuing a Master's degree in Computer Science and Engineering at NTU. She received her B. E. degree in Electronics and Communications Engineering from Anna University in 2011. Her research interests include application of deep learning and energy modeling for IoT devices.

Arvind Easwaran is an Associate Professor in the School of Computer Science and Engineering at NTU. He received a PhD degree in Computer and Information Science from the University of Pennsylvania, USA, in 2008. In NTU, his research focuses on the design and analysis of real-time and cyber-physical systems, including their application in domains such as automotive, manufacturing and energy systems.

Etienne Borde is an Associate Professor in the School of Computer Engineering at Telecom Paris (Paris, France) since 2011. He received his Ph. D. degree in Computer Science from the same university in 2009. His research interests are: component-based software engineering, architecture description languages, model transformation, code generation and formal verification.

Sebastian Steinhorst is an Associate Professor in Electrical and Computer Engineering at TUM where he leads the Embedded Systems and Internet of Things group. His research centers around hardware-software architecture co-design of secure decentralized embedded and cyber-physical systems in the application areas of industrial IoT, automotive and smart energy. He is also a Co-Program PI at TUMCREATE.