Resource time-sharing for IoT applications with deadlines

George Karakostas[1⋆] and Stavros G. Kolliopoulos[2]

[1] Department of Computing and Software, McMaster University, Hamilton ON, Canada
karakos@mcmaster.ca
[2] Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Athens, Greece sgk@di.uoa.gr

Abstract. Motivated by time-sharing systems with deadlines, such as 2-way synchronization of Digital Twins, we introduce the study of a very natural problem which can be abstracted as follows. We are given m machines and n jobs, as well as a set of *tolerance capacities* $u_{ij} \geq 0$ for every job j and machine i. Can we assign the jobs so that, if job j ends up on machine i, at most u_{ij} jobs in total are processed on i? We define two natural optimization versions: (i) Maximize the total weight of jobs that can be assigned without violating the tolerance capacities u_{ij}, and (ii) minimize the amount $\rho \geq 1$, by which capacities have to be scaled so that all jobs can be assigned. For the first problem and its generalizations we provide an $(1 - 1/e)$-approximation algorithm. For the second problem we show that it is $n^{1/2-\varepsilon}$-inapproximable and provide linear integrality gap lower bounds for two key relaxations.

Keywords: time-sharing · deadlines · tolerance capacities · scheduling

1 Introduction

An ever increasing number of applications within the framework of the Internet-of-Things (IoT) depend on *real-time* response times, i.e., the ability of delivering data to and from the application, as well as processing data, with acceptable delays [24]. These delay requirements can apply to almost all different components of an IoT architecture. For example, the need for fast delivery of the 'freshest' available sensor data to cyber-physical systems has lead to the recent concept of the Age-of-Information [25], i.e., the scheduling of data transmission so that the largest latency between data generation at a source and its delivery to the application is minimized. A different approach to the requirement for timely data delivery and processing is the imposition of delay constraints (as opposed to delay as objective in AoI), that can guarantee the real-time nature of the system. An example of this latter approach is the concept of *Digital Twins (DT)* [17, 21]. These are virtual replicas of physical systems (PS), which capture a subset of the PS's features, maintain a 2-way synchronization of DT and PS states, and can store data relevant to the PS in order to perform computationally-heavy tasks, such as prediction and data analytics.

The 2-way synchronization requires that a DT performs periodically its data transmission and task processing, and is expected to finish both within a given period, thus assuring the data 'freshness'. With the proliferation of DTs, the time-sharing of critical resources, such as wireless channels and CPUs, by many DTs simultaneously puts a strain on the satisfaction of these timing requirements, and motivates the scheduling problem we introduce in this work as follows: A DT j with a synchronization period T_j, needs c_j CPU cycles in order to complete its data processing task (for

⋆ Research supported by an NSERC Discovery grant.

simplicity we assume the data transmission time is negligible). DT j's task is executed on a server i of CPU frequency f_i together with the tasks of K other DTs, which share the CPU equally and continuously with j. In order for j's task to finish on time, the inequality $\frac{Kc_j}{f_i} \leq T_j$ must hold (j gets every K-th cycle of the CPU), which implies that $K \leq \frac{f_i T_j}{c_j}$, i.e., DT j's task can co-execute with at most $u_{ij} := \frac{f_i T_j}{c_j}$ (including itself) DTs on server i. A similar situation arises when several DTs share a wireless channel with their PSs, using TDMA. The natural question then is how to schedule a set of tasks with deadlines on a set of servers, when Round-Robin time-sharing is applied, and so that all deadlines are respected.

Although time-sharing scheduling with deadlines has been the motivation, we can move one level of abstraction higher, ignore the provenance of u_{ij} above, and ask the following general question, which is a very natural one and can apply to settings well beyond IoT. Given a set of n distinct balls, a set of m distinct bins, and a set of nm nonnegative integers u_{ij}, can the balls be placed in the bins so that if the j-th ball is placed in bin i, at most u_{ij} balls in total are placed in bin i? Note that some tolerance capacities u_{ij} can be equal to 0, i.e., job j cannot be assigned to machine i. The formal definition of the problem we examine is the following.

Machine-Sharing with Tolerance Capacities (MSTC)
Input: Set of n jobs $\mathcal{J}$, set of m machines $\mathcal{M}$, tolerance capacities $u_{ij} \in \mathbb{Z}_{\geq 0}$ for all $(i,j) \in \mathcal{M} \times \mathcal{J}$.
Output: An assignment $\sigma : \mathcal{J} \to \mathcal{M}$ such that $|\sigma^{-1}(\sigma(j))| \leq u_{\sigma(j)j}$ $\forall j \in \mathcal{J}$, or NO if no such assignment exists.

For job $j \in \mathcal{J}$, $\mathcal{M}(j)$ denotes the set of machines on which j can be assigned, i.e., $\mathcal{M}(j) = \{i \in \mathcal{M} \mid u_{ij} > 0\}$. Similarly, for $i \in \mathcal{M}$, $\mathcal{J}(i) = \{j \in \mathcal{J} \mid u_{ij} > 0\}$. With this notation in place, MSTC can be equivalently formulated as finding a feasible solution to the following quadratic program:

$$\sum_{i \in \mathcal{M}(j)} x_{ij} \geq 1 \qquad \forall j \in \mathcal{J} \tag{QP}$$

$$x_{ij} \cdot \sum_{k \in \mathcal{J}(i)} x_{ik} \leq u_{ij} \quad \forall j \in \mathcal{J}, \forall i \in \mathcal{M}(j)$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in \mathcal{M}, \forall j \in \mathcal{J}(i)$$

MSTC can be easily seen to be NP-complete (e.g., via a reduction from SAT). To the best of our knowledge, this is the first time that the problem has been studied. The coverage problem with group budget constraints defined and studied in [4] comes perhaps closest to the spirit of MSTC. Assignment problems with forbidden *pairs* of assignments have been studied in the literature (e.g., [8]), but are incomparable to MSTC.

Problem MSTC gives rise to two natural optimization versions. Let every job j have a weight $w_j \geq 0$. One can ask for a *maximum-weight* set of jobs that can be assigned to machines without violating any tolerance capacities, together, possibly, with additional constraints. An immediate additional constraint is to require that no more than k machines can be used, but more natural constraints include job costs and a budget that should not be exceeded, or resource augmentation (e.g., more UAVs used as relays at a location to increase the number of available channels [18]), or bundles of jobs that *have* to be executed on a machine, or combinations of the above. We call this general family of problems the Maximum Machine-Sharing with Tolerance Capacities (MMSTC). The second problem derived from MSTC is a *congestion* version of the original that asks for the smallest scaling factor that one can multiply all tolerance capacities with, so that there is a feasible assignment for all jobs.

Definition 1. *For $\rho \geq 1$, an assignment σ is* ρ-feasible *if for all $j \in \mathcal{J}$, $|\sigma^{-1}(\sigma(j))| \leq \rho \cdot u_{\sigma(j)j}$.*

The Scaled Machine-Sharing with Tolerance Capacities (SMSTC) asks for the minimum scaling factor ρ, such that there is a ρ-feasible assignment. The case of SMSTC where the tolerance capacities u_{ij} are equal to a common value T_i, for all $j \in \mathcal{J}(i)$, is the famous scheduling problem with machine deadlines problem for which Lenstra et al. gave a 2-approximation [16]. Our problem is more general, as every job has its own upper bound on the completion time of machine i, namely u_{ij}. The algorithm of [16] assumes a size p_{ij} for every job j and machine i. In the negative results we provide for SMSTC every job j has unit size on the set of machines $\mathcal{M}(j)$ it can be assigned to.

MMSTC in its simplest form (only tolerance capacities constraints) via an appropriate reformulation (see Section 2) can be efficiently reduced to the Separable Assignment Problem (SAP) of [9], and also to Maximum Coverage with Group Budgets, defined in [4], with an implicit set system that describes feasible packings of jobs. The simple randomized rounding of [9] yields an $(1-\frac{1}{e})$-approximation, but cannot handle the additional constraints mentioned above. Based on an equivalent formulation of MSTC presented in Section 2, we also design an $(1-\frac{1}{e})$-approximation algorithm for the simplest version of MMSTC (cf. Theorem 2), using the more sophisticated dependent rounding technique of [10, 22] on a configuration LP relaxation. Unlike [9] though, the dependent rounding and the results presented in Section 3.2 can be extended to include the additional constraints. The requirement that no more than k machines are used can still be $(1-\frac{1}{e})$-approximated by the algorithms of [2, 5] that generalize [9] to matroidal constraints. In Section 3.3 we show how dependent rounding can be extended to also give a $(1-\frac{1}{e})$-approximation for this case (cf. Theorem 4), as a template to deal with job costs, resource augmentation, required job bundles, or combinations of the above. Our algorithm can be extended to handle arbitrary integer job sizes p_{ij}, for $j \in \mathcal{J}$ and $i \in \mathcal{M}(j)$, at the cost of a $(1+\varepsilon)$ scaling of the capacities.

Unfortunately, SMSTC turns out to be much harder to approximate. Using a reduction from 3D-Matching, we show that there is no polynomial-time $(n^{1/2-\varepsilon})$-approximation algorithm for SMSTC, unless $\mathsf{P} = \mathsf{NP}$ (cf. Theorem 7). Here $n = |\mathcal{J}|$. The bound holds even when every job j has the same tolerance capacity $u_{ij} = u_j$ on every machine in $\mathcal{M}(j)$. In order to tackle the problem algorithmically, we explore two key relaxations. First, we study the configuration LP, a powerful linear relaxation that was introduced in the context of the cutting stock problem [6, 12] and has been used among other for bin packing [15,19] and scheduling problems with assignment restrictions (e.g., [1,14,23]). Applied to the SMSTC problem, it is strictly stronger than the natural LP that has assignment variables x_{ij} for job-machine pairs. We prove that the configuration LP has an integrality gap of $\Omega(n)$ for congestion even when there are only two distinct tolerance capacity values, every job j has the same tolerance capacity u_j on every machine, and each job can be assigned to at most two machines (cf. Theorem 5). The second relaxation we consider is the formulation resulting from the quadratic program (QP) by relaxing the integrality constraints. Notably, this is a non-convex program. Still we show that it has an integrality gap of at least m, the number of machines. The lower bound holds again when every job has a machine-independent tolerance capacity (cf. Theorem 6). Hence, rounding the fractional solution of these two key formulations cannot give a non-trivial approximation factor. We leave the closing of the gap between Theorem 7 and Theorem 5 (or Theorem 6) as an open problem.

2 An equivalent formulation of MSTC

In this section we give an equivalent formulation of the MSTC problem. Recall that we are given as input a set $\mathcal{M}$ of machines, a set $\mathcal{J}$ of jobs and a set $\{u_{ij} \in \mathbb{Z}_{\geq 0} \mid (i,j) \in \mathcal{M} \times \mathcal{J}\}$. For $i \in \mathcal{M}$, let $d(i)$ denote $|\mathcal{J}(i)|$. Sort the capacities u_{ij}, $j \in \mathcal{J}(i)$, in non-decreasing order $u_{ij_1} \leq u_{ij_2} \leq \ldots \leq u_{ij_{d(i)}}$. Let m_i denote the number of distinct values in the sequence $u_{ij_1}, u_{ij_2}, \ldots, u_{ij_{d(i)}}$. Denote these distinct values in increasing order as $\bar{u}_{i1} < \bar{u}_{i2} < \ldots < \bar{u}_{im_i}$. Each machine $i \in \mathcal{M}$ consists of a set S_i of m_i submachines where submachine $k \in S_i$ has capacity $\bar{u}_{ik}$. The set $\mathcal{J}(i,k)$ of jobs that can be assigned to submachine $k \in S_i$ consists of

$$\{j \in \mathcal{J}(i) \mid u_{ij} \geq \bar{u}_{ik}\}.$$

Similarly, the set of submachines of machine i to which job j can be assigned is denoted $\mathcal{M}(i,j)$.

A *submachine assignment* of the jobs in $S \subseteq \mathcal{J}$ is a mapping $\psi : S \to \cup_{i \in \mathcal{M}} S_i$ such that (i) $\forall j \in S$, $\psi(j) \in \mathcal{M}(i,j)$ and (ii) for all $i \in M$, at most one of the sets $\psi^{-1}(k)$, $k \in S_i$, is nonempty. In words, every job j in S is assigned to a submachine of a machine in $\mathcal{M}(j)$ and for every machine $i \in \mathcal{M}$, at most one of the submachines in S_i can be "open". In analogy to Definition 1, the submachine assignment ψ is ρ-feasible, for $\rho \geq 1$, if $\forall i \in \mathcal{M}, \forall k \in S_i$, $|\psi^{-1}(k)| \leq \rho \bar{u}_{ik}$.

Clearly the two problem formulations are equivalent, i.e., there is a ρ-feasible assignment σ if and only if there is ρ-feasible submachine assignment ψ. In the following sections we will choose each time the problem formulation (with or without submachines) that is more convenient.

3 Approximation algorithms for MMSTC

In this section we consider the family of MMSTC problems. The simplest version is the following: Given input $\{u_{ij} \in \mathbb{Z}_{\geq 0} \mid (i,j) \in \mathcal{M} \times \mathcal{J}\}$, and a function $w : \mathcal{J} \to \mathbb{Q}_{\geq 0}$ that assigns weights to jobs, find a maximum-weight $S \subseteq \mathcal{J}$ for which there is a 1-feasible assignment.

3.1 Linear relaxation with configurations

For machine i and submachine $k \in S_i$, a subset $C \subseteq \mathcal{J}(i,k)$ is a *configuration* if $|C| \leq \bar{u}_{ik}$. The set of these configurations is denoted $\mathcal{C}(i,k)$. The configuration LP, denoted (CLP), has a variable x_{i,C_k} for each machine i, submachine $k \in S_i$, and configuration $C_k \in \mathcal{C}(i,k)$:

$$\max \sum_{j \in \mathcal{J}} w_j \left(\sum_{i \in \mathcal{M}} \sum_{k \in S_i} \sum_{C_k :\, j \in C_k} x_{i,C_k} \right) \tag{CLP}$$

$$\sum_{k \in S_i} \sum_{C_k} x_{i,C_k} \leq 1 \qquad \forall i \in \mathcal{M} \tag{1}$$

$$\sum_{i \in \mathcal{M}} \sum_{k \in S_i} \sum_{C_k : j \in C_k} x_{i,C_k} \leq 1 \qquad \forall j \in \mathcal{J} \tag{2}$$

$$x_{i,C_k} \geq 0 \qquad \forall i \in \mathcal{M}, \forall k \in S_i, \forall C_k \in \mathcal{C}(i,k) \tag{3}$$

The set of constraints (1) ensures that each machine is assigned at most one configuration and that at most one submachine is open. Constraints (2) ensure that each job is assigned at most once.

Clearly, an integer solution to (CLP) corresponds to a 1-feasible assignment for a maximum-weight subset of $\mathcal{J}$. For a configuration C_k, let $w(C_k) := \sum_{j \in C_k} w_j$. The dual of (CLP) is the following:

$$\min \sum_{i \in M} y_i + \sum_{j \in \mathcal{J}} z_j \tag{D-CLP}$$

$$y_i + \sum_{j \in C_k} z_j \geq w(C_k) \qquad \forall i \in \mathcal{M}, \forall k \in S_i, \forall C_k \in \mathcal{C}(i,k) \tag{4}$$

$$y, z \geq 0 \tag{5}$$

A solution for (D-CLP) (and, therefore, (CLP)) can be computed using the ellipsoid algorithm as follows: Given a candidate solution (y^*, z^*) to (D-CLP), its separation oracle has to solve $\sum_{i \in \mathcal{M}} |S_i|$ instances of a knapsack-like problems, which we denote Kpn. Fix $i \in \mathcal{M}$, $k \in S_i$. Consider a Kpn instance with $\mathcal{J}(i,k)$ being the set of items and knapsack capacity $\bar{u}_{ik}$. Every item j has a size $s_j = 1$ and a (possibly negative) value $v_j = w_j - z_j$. The oracle returns a violated inequality iff there is a feasible packing in the knapsack with total value that exceeds y_i. The Kpn instance can be solved by the obvious greedy algorithm in $O(n \log n)$ time.

3.2 Dependent-rounding algorithm

In this section we present our dependent rounding for the simplest version of MMSTC, i.e., finding and integral solution of (CLP). Although the approximation result we obtain in Theorem 2 can also be obtained by the simple randomized rounding for SAP in [9], this section provides the algorithmic foundations for the extensions of Section 3.3.

Let x^* be an optimal solution of (CLP). Without loss of generality we can assume that

$$\sum_{k \in S_i} \sum_{C_k} x^*_{i,C_k} = 1, \quad \forall i \in \mathcal{M}.$$

The vector x^* induces on each machine a probability distribution on the submachines in S_i. We will use dependent rounding to choose one configuration per machine and ensure that a near-optimal fraction of jobs is scheduled.

Srinivasan [22] (see also [10]) has provided a technique to sample algorithmically from a distribution with the following properties. Consider any sequence of t reals $P = (p_1, \ldots, p_t)$ such that $p_i \in [0,1]$ and $\sum_i p_i$ is an integer l. F_t is defined as a family of distributions over vectors in $\{0,1\}^t$. A member $D(t;\, P)$ of the family F_t guarantees the following properties on any vector $(X_1, \ldots, X_t)$ sampled from $D(t; P)$.

(A1) *(probability preservation)* $\forall i, \Pr[X_i = 1] = p_i$.
(A2) *(degree preservation)* $\Pr[|\{i\colon X_i = 1\}| = l] = 1$.
(A3) *(negative correlation)* For all $S \subseteq [t]$ we have $\Pr[(\bigwedge_{i \in S}(X_i = 0)] \leq \prod_{i \in S} \Pr[X_i = 0]$ and $\Pr[(\bigwedge_{i \in S}(X_i = 1)] \leq \prod_{i \in S} \Pr[X_i = 1]$.

Theorem 1 ([22]). *Given $P = (p_1, \ldots, p_t)$ there is a linear-time algorithm that generates a sample from distribution $D(t; P)$.*

Let $\mathcal{C}_i$ denote the disjoint union of the sets of configurations in $\bigsqcup_{k\in S_i} \mathcal{C}(i,k)$ whose corresponding variable has a nonzero value in the solution x^* of (CLP). Every configuration in $\mathcal{C}_i$ belongs to a unique $\mathcal{C}(i,k)$. Denote by $x^*|i$ the restriction of vector x^* to the entries corresponding to the configurations in $\mathcal{C}_i$. To simplify notation, set $t_i = |\mathcal{C}_i|$. We define a distribution $D(t_i;\, x^*|i)$ that satisfies properties (A1), (A2), (A3) for each machine $i \in \mathcal{M}$. Observe that in our setting $l = 1$. The rounding algorithm is the following.

ALGORITHM DEPROUND
For all $i \in \mathcal{M}$, do independently:

1. Using the algorithm of Theorem 1, sample from $D(t_i; x^*|i)$ to obtain vector $X^{(i)} \in \{0,1\}^{t_i}$. By Property (A2), $X^{(i)}$ has a unique entry equal to 1.
2. Assign the configuration C that corresponds to the nonzero entry of $X^{(i)}$ to machine i.

Theorem 2. *Algorithm* DEPROUND *runs in polynomial-time and outputs a* 1*-feasible assignment for a set of jobs* S *whose expected total weight is at least* $(1-1/e)$ *times the optimum of the* (CLP) *relaxation.*

Proof. Let $\mathcal{C}$ denote the disjoint union $\bigsqcup_{i\in\mathcal{M}} \mathcal{C}_i$. That is, every configuration in $\mathcal{C}$ corresponds to a unique (i,k) pair. For $j \in \mathcal{J}$, let z_j be the random variable that takes value 1 if job j is assigned by Algorithm DEPROUND and zero otherwise. Our analysis is quite similar to the analysis in [22] for Maximum Coverage versions of Set Cover. We slightly abuse notation and index the entries of the vectors $X^{(i)}$ by the corresponding configurations. Since every configuration C belongs to a unique $\mathcal{C}_i$ we omit the superscript i as well.

$$\begin{aligned}
\Pr[z_j = 1] =& 1 - \Pr[\bigwedge_{C\in\mathcal{C}:C\ni j} (X_C = 0)] \\
\geq& 1 - \prod_{C\in\mathcal{C}:C\ni j} \Pr[X_C = 0] \qquad (6)\\
=& 1 - \prod_{C\in\mathcal{C}:C\ni j} (1 - x^*_C) \qquad (7)
\end{aligned}$$

Inequality (6) follows from the "negative correlation" property (A3) and Equality (7) from property (A1). Define $z^*_j := \sum_{C\in\mathcal{C}:C\ni j} x^*_C$. Thus the fractional amount by which job j is scheduled and the objective value of the solution x^* is equal to $\sum_{j\in\mathcal{J}} w_j z^*_j$. Using the AM-GM inequality and the fact that $z^*_j \leq 1$, it is easy to see that

$$\prod_{C\in\mathcal{C}:C\ni j} (1 - x^*_C) \leq (1 - z^*_j/s)^s$$

where s is the maximum number of configurations in the support of x^* that a job belongs to. By calculus, $1 - (1 - z^*_j/s)^s \geq (1 - (1-1/s)^s) \cdot z^*_j > (1 - 1/e) \cdot z^*_j$. □

The results in this section can be easily extended to the case where every job j has an integer size $p_{ij} \geq 1$ for $i \in \mathcal{M}(j)$ and an assignment σ of set $S \subseteq \mathcal{J}$ is *ρ-feasible* if

$$\forall j \in S, \quad \sum_{k\in\sigma^{-1}(\sigma(j))} p_{\sigma(j)k} \leq \rho \cdot u_{\sigma(j)j}.$$

By rounding the job sizes as explained in [1] we can solve in polynomial time the Configuration LP while using configurations whose size is at most $(1+\varepsilon)$ the capacity of the corresponding submachine. As an alternative, standard techniques [3,13] can be used to bring the capacity violation factor into the approximation factor instead. Applying the algorithm DepRound yields the following (also by simple randomized rounding [9]):

Theorem 3. *If every job j in $\mathcal{J}$ has an integer size p_{ij}, for $i \in \mathcal{M}(j)$, one can in polynomial-time compute a set of jobs S whose expected total weight is at least $(1-1/e)$ times the optimum of the* (CLP) *relaxation and an assignment for S that is $(1+\varepsilon)$-feasible for any arbitrary constant $\varepsilon > 0$.*

3.3 Constraint extensions of MMSTC

The techniques of the previous section can be extended to derive a good approximation for the extensions of MMSTC, as mentioned in Section 1. More specifically, an $(1-\frac{1}{e})$-approximation of the objective can be obtained for the following additional constraints:

Coverage: No more than k machines can be used.
Bundles of required jobs: Given k bundles of jobs $C_1, \ldots, C_k$, maximize the total scheduled job weight, while scheduling all bundles on machines.
Required machines: Given a set of machines $A \subseteq \mathcal{M}$, the machines in A *must* be used.
Resource augmentation: Given integers $l_1, \ldots, l_{|\mathcal{M}|}$, the schedule can use at most l_i copies of the i-th machine.
Budget: Given assignment costs c_{ij} for all jobs j on machines i, and a budget B, the total cost of scheduled jobs cannot exceed the budget B.

In the full version of this abstract, we show that the dependent rounding of Section 3.2 can be extended to $(1-\frac{1}{e})$-approximate these versions, or their combinations, while not violating the the coverage, resource augmentation, bundles of required jobs constraints, required machines, and not violating the budget and capacity constraints by much (or, alternatively, not at all with an ε cost to the approximation factor, for any $\varepsilon > 0$). As a template for dealing with the extra constraints, in this abstract we show how this can be done for the Coverage-MMSTC, which requires that no more than k machines can be used. Since this constraint by itself happens to be matroidal, the algorithms of [2,5] also achieve an approximation factor of $(1-\frac{1}{e})$.

To simplify our exposition, we will assume that we have guessed the exact number of machines $k_0 \leq k$ used by the optimal solution (by 'guessing' we mean the exhaustive enumeration of k_0 values, and the output of the maximum obtained solution). Formulation (CLP) can be extended by adding a special empty configuration $C_\emptyset$:

$$\max \sum_{j\in\mathcal{J}} w_j \left(\sum_{i\in\mathcal{M}} \sum_{k\in S_i} \sum_{C_k:\ j\in C_k} x_{i,C_k} \right) \tag{CovCLP}$$

$$\sum_{i\in\mathcal{M}} x_{i,\mathcal{C}_\emptyset} = m - k_0 \tag{8}$$

$$\sum_{k\in S_i} \sum_{C_k} x_{i,C_k} + x_{i,\mathcal{C}_\emptyset} = 1 \qquad \forall i \in \mathcal{M} \tag{9}$$

$$\sum_{i\in\mathcal{M}} \sum_{k\in S_i} \sum_{C_k: j\in C_k} x_{i,C_k} \leq 1 \qquad \forall j \in \mathcal{J} \tag{10}$$

$$x_{i,C_k} \geq 0 \qquad \forall i \in \mathcal{M}, k \in S_i, C_k \in \mathcal{C}(i,k) \cup \{\mathcal{C}_\emptyset\} \tag{11}$$

Note that it can still be the case of assigning an empty configuration in $\mathcal{C}_i$ to i, but that doesn't matter (it means that eventually fewer than k_0 machines are used by the solution).

If w is the (unrestricted) dual variable corresponding to (8), then the objective of (D-CLP) becomes $\sum_{i\in M} y_i + \sum_{j\in\mathcal{J}} z_j + (m-k_0)w$, and the constraints $w + y_i \geq 0,\ \forall i \in \mathcal{M}$ are added. The separation oracle of (D-CLP) can be easily extended to solve the new dual LP, and, therefore compute the optimal solution x^* to (CovCLP). The dependent rounding of Section 3.2 becomes the rounding procedure of [10] on a bipartite graph $G = (A, B, E)$, constructed as follows: Each distribution $D(t_i;\ x^*|i)$ gives rise to a star with machine i at its center, and the configurations of $\mathcal{C}_i$ in the support of x^* at the leaves. Side A contains the machines/centers of these stars, while the leaves of the stars are vertices in side B. We add another vertex $C_\emptyset$ to B, and connect all vertices $i \in A$ with $x^*_{i,\mathcal{C}_\emptyset} > 0$ to $C_\emptyset$.

Note that variable x^*_{i,C_k} corresponds to edge (i, C_k). We require a rounding of these variables that achieves degree 1 for all vertices in A, and degree k_0 for vertex $C_\emptyset$ in B (we don't care about the degrees of the rest of the nodes in B). The dependent randomized rounding of [10] satisfied these requirements with probability 1 (property (A2)). Moreover, as is proven in Theorem 4.4 of [20], the configurations in B are negatively correlated (property (A3)), i.e., the indicator random variables X_C of configurations C being picked or not are negatively correlated, and this for arbitrary fractional degree bounds on the vertices of B ([20, p. 684]). Hence, (6) carries through, and, together with the known lower bound for the Maximum Coverage problem [7], we have the following (also by the algorithms of [2, 5]):

Theorem 4. *A solution for problem* Coverage-MMSTC *that is* 1*-feasible and assigns in expectation at least* $(1 - 1/e)$ *times the optimal weight of jobs can be computed in polynomial time. This factor is best possible, unless* $\mathsf{P} = \mathsf{NP}$.

Similarly to Theorem 3, Theorem 4 can be extended to the case where job j has an integer size p_{ij}, for $i \in \mathcal{M}(j)$.

4 Minimizing Congestion

In this section we consider the SMSTC problem. Given input $\{u_{ij} \in \mathbb{Z}_{\geq 0} \mid (i,j) \in \mathcal{M} \times \mathcal{J}\}$, find the minimum $\rho \geq 1$ for which there is a ρ-feasible assignment for the set $\mathcal{J}$. We show integrality gap lower bounds in Section 4.1 and a hardness of approximation result in Section 4.2.

4.1 Integrality gaps for SMSTC

Let P be a valid mathematical programming relaxation for computing an assignment for the jobs in $\mathcal{J}$. For $f \geq 1$, we say that P has an *integrality gap of at least* f *for congestion* if there is an instance $I = \{u_{ij} \mid (i,j) \in \mathcal{M} \times \mathcal{J}\}$ for which P is feasible, but P has no integer feasible solution for any instance $I_\rho = \{\rho \cdot u_{ij} \mid (i,j) \in \mathcal{M} \times \mathcal{J}\}$ with $\rho < f$.

We start by showing an integrality gap for the Configuration LP relaxation. We define an instance Ξ where each job will have the same capacity on each machine it can be assigned to. The set of machines consists of three blocks of machines, blocks A, B and C. Each machine has two submachines, one with large and one with small capacity. Accordingly we refer to the *big* and the *small* submachine of a given machine. All machines within the same block X have the same large and small capacities at their two submachines. These capacity values are denoted U_X and u_X respectively.

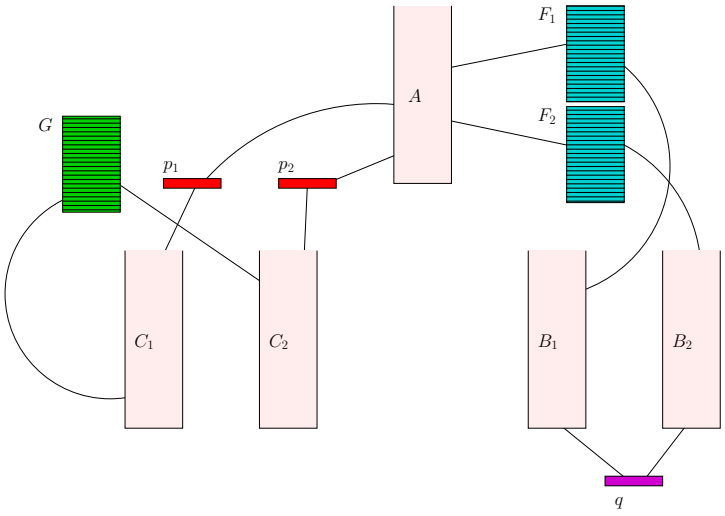

Fig. 1: Instance Ξ used in the integrality gap construction.

Block A consists of a single machine with $U_A = 2k$ and $u_A = 2$, where $k \geq 2$ is a positive integer of our choice. We refer to this single machine as machine A.

Block B consists of 2 machines, B_1 and B_2. The submachine capacities are $U_B = 2k$ and $u_B = 2$.

Block C consists of 2 machines, C_1 and C_2. The submachine capacities are $U_C = 2k$ and $u_C = 2$.

All jobs have processing time (height) 1. They are partitioned into two sets, those that can only be assigned to small submachines and those that can be assigned to big and small submachines. By slightly abusing terminology we refer to the corresponding sets as *small* and *large* jobs respectively. In what follows when we say that a job may be assigned to the big submachine of machine x it is implied that it can also be assigned to the small submachine of x.

The set of large jobs consists of the disjoint union of two sets F and G. Set F contains $2k$ jobs divided into 2 groups F_1, F_2 each containing k jobs. G consists of k jobs.

- The jobs of F can be assigned to the big submachine of machine A. The jobs of F_i, can be assigned to the big submachine of machine B_i, $i = 1, 2$.
- The jobs of G can be assigned to the big submachines of machines C_1, C_2.

The set of small jobs consists of the disjoint union of two sets P and Q.

- P contains 2 jobs p_1, p_2 that can be scheduled on the small submachine of machine A. Moreover p_i can be assigned on the small submachine of machine C_i, $i \in \{1, 2\}$.
- Set Q contains 1 job, call it q. Job q can be scheduled on the small submachine of machines B_1, B_2.

See Fig. 1 for a depiction of the instance Ξ. The Configuration LP for congestion minimization is the following.

$$\begin{aligned} &\sum_{k\in S_i}\sum_{C_k} x_{i,C_k} \le 1 && \forall i \in \mathcal{M} \\ &\sum_{i\in\mathcal{M}}\sum_{k\in S_i}\sum_{C_k : j\in C_k} x_{i,C_k} \ge 1 && \forall j \in \mathcal{J} \\ &x_{i,C_k} \ge 0 && \forall i \in \mathcal{M}, \forall k \in S_i, \forall C_k \in \mathcal{C}(i,k) \end{aligned} \tag{CCLP}$$

Lemma 1. *Linear program* (CCLP) *has a feasible half-integral solution x for the instance Ξ.*

Proof. There is a feasible half-integral solution where to every submachine a configuration of x-value $1/2$ is assigned. The jobs in $F \cup G \cup P \cup Q$ can be assigned exactly to two machines each and in x they are split equally among these two machines. It is easy to see that these split jobs (of width $1/2$ and height 1) can be packed into configurations of width $1/2$ and height that does not exceed the capacity of the corresponding submachine. □

Lemma 2. *Any integer solution for the instance Ξ that leaves no job unassigned has a congestion of at least $k/4$.*

Proof. There are two possible cases for a feasible solution.

Case 1. There is an $i \in \{1,2\}$ such that at least half of the jobs in F_i are not scheduled on B_i. I.e., there is an $i \in \{1,2\}$ such that the big submachine of B_i contains less than half of the jobs in F_i. Therefore at least $k/2$ jobs from F_i are scheduled on machine A. If some job p_j from P is present on A, then p_j experiences a congestion of at least $(k/2)/(u_A) = k/4$. If no job from P is present on A, each machine of block C hosts exactly one job from P. For every $i \in \{1,2\}$ at least half of the jobs in G must end up on a machine C_{i^*}, for some $i^* \in \{1,2\}$. The corresponding small job p_{i^*} experiences a congestion of at least $(k/2)/u_C = k/4$.

Case 2. For every $i \in \{1,2\}$, less than half of the jobs in F_i are not scheduled on B_i. I.e., for every $i \in \{1,2\}$ the big submachine of B_i contains at least half of the jobs of F_i. There is an $i^* \in \{1,2\}$ such that the job q is assigned to B_{i^*}. Then this small job experiences a congestion of at least $(k/2)/(u_B) = k/4$. □

The total number n of jobs in the instance Ξ is equal to $3(k+1)$. We have proved the following theorem.

Theorem 5. *The integrality gap of the Configuration LP* (CCLP) *for minimizing congestion on an instance of n jobs is at least $(n-3)/12$ even when (i) there are only two distinct capacity values (ii) every job j has the same tolerance capacity on every machine in $\mathcal{M}(j)$ (iii) each job can be assigned to at most two machines.*

Define (QP-F) to be the relaxation of (QP) where the integrality constraints are replaced by

$$x_{ij} \ge 0 \quad \forall i \in \mathcal{M}, \forall j \in \mathcal{J}(i).$$

The instance Ξ we used in Theorem 5 is infeasible for (QP-F). We define a new instance Υ. Let $m = U$ be the number of machines, for some integer $U \ge 2$. The number n of jobs is $m(U-1)+1$.

Recalll that for a positive integer t, $[t]$ denotes the set $\{1, 2, \ldots, t\}$. Each machine $i \in [m]$, has a cluster $J_i = \{j_l^i \mid l \in [U-1]\}$ of $U-1$ "private" clients that can only be assigned to i and have each tolerance capacity U. The remaining single job out of the n, call it job 1, can be assigned to all machines, with tolerance capacity $u_{i1} = 1$, for all i. The set of jobs is $\mathcal{J} = \{1\} \cup \bigcup_{i=1}^{m} J_i$. In any feasible integer solution, there is a machine $i^* \in [m]$ that processes job 1. In order to service the jobs in J_{i^*} a congestion of U has to be incurred.

There is a feasible fractional solution to (QP-F). For every $i \in [m]$ and for every job $d = j_l^i \in \mathcal{J}_i$, set $x_{i,d} = 1$. For every $i \in [m]$, set $x_{i1} = 1/U$. Because $U = m$, job 1 is completely serviced. It is easy to see that all capacity constraints are met.

Theorem 6. *The integrality gap of (QP-F) with respect to congestion is at least m, where m is the number of machines in the instance. This holds even when (i) there are only two distinct capacity values (ii) every job j has the same tolerance capacity on every machine in $\mathcal{M}(j)$.*

4.2 Hardness of approximation for SMSTC

It is well-known that the following problem is NP-complete [11].

Bounded 3D-Matching
Input: Set of triples $M \subseteq A \times B \times C$, where A, B, and C are pairwise disjoint sets having the same number q of elements. Every element of $A \cup B \cup C$ occurs in at most 3 triples.
Question: Does M contain matching, i.e., a subset $M' \subseteq M$ such that $|M'| = q$ and no two elements of M' agree in any coordinate?

Recall Definition 1. An instance of SMSTC which has an f-feasible assignment is called an *f-feasible instance.*

Given an instance I of Bounded 3D-Matching we construct an instance $\phi(I)$ of SMSTC. For every triple in M we have in $\phi(I)$ a dedicated *triple-machine.* For every $i \in [q]$, group together all machines that correspond to triples whose first coordinate is $a_i \in A$ in a group M_i. For every such group, add $|M_i|$ *overflow machines* OF_i^p, $p \in [|M_i|]$. Without loss of generality we may assume that $|M_i| > 1$. The reader is invited to bear in mind from now on that $|M_i| \in \{2, 3\}$. The total number of machines in the instance $\phi(I)$ is $|M| + \sum_{i=1}^{q} |M_i| \leq 12q$.

For $i \in [q]$, we create $|M_i|$ blocks of "dummy" jobs F_i^p, $p \in [|M_i|]$. Every block contains f jobs where $f > 1$ is an integer we will define later. The jobs of block F_i^p can be scheduled only on machine $m_i^p \in M_i$ and on the overflow machine OF_i^p. The tolerance capacity of every dummy job is f on both machines it can be assigned to. Observe that if a block of dummies ends up on a machine, nothing else can be assigned there without incurring congestion larger than 1.

For every $b_j \in B$ we create a set B_j of f' jobs b_j^r, $r \in [f']$, each of which can only be scheduled on the triple-machines that correspond to triples on which b_j is the second component. The quantity f' is an integer larger than f and will be defined later. For every $c_k \in C$ we create a set C_k of f' jobs c_k^r, $r \in [f']$, each of which can only be scheduled on the triple-machines that correspond to triples on which c_k is the third component. The tolerance capacity of these jobs is $2f'$. See Figure 2a.

Finally, there is a set of "placeholder" jobs whose mission will be to block some of the overflow machines. In particular, for every $i \in [q]$, there are d_i placeholder jobs with $d_i = 1$ if $|M_i| = 2$ and $d_i = 3$ if $|M_i| = 3$. The set of placeholder jobs is denoted as $\{x_i^p\}_{p \in [d_i]}$. The assignment possibilities are defined as follows. *Case 1:* $d_i = 1$. The single placeholder job x_i^1 can be assigned to the overflow machines OF_i^p, $p \in \{1, 2\}$, with a tolerance capacity of 1. *Case 2:* $d_i = 3$. Every placeholder job can

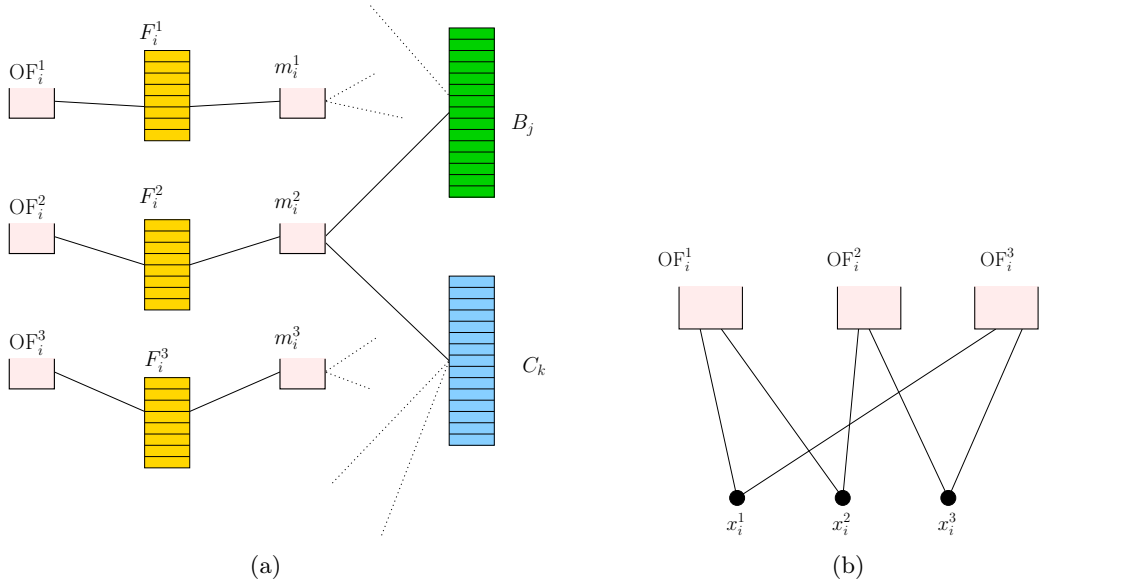

Fig. 2: The construction for the hardness reduction of SMSTC. (a) Example of a group M_i of size 3 and the jobs that can be assigned to the triple-machines in M_i. Machine m_i^2 corresponds to the triple (a_i, b_j, c_k). (b) Allowed assignments for the three placeholder jobs x_i^p, $p \in \{1, 2, 3\}$.

be assigned to exactly two of the overflow machines OF_i^p, $p \in [3]$, in the way shown in Figure 2b. The tolerance capacity of each placeholder job is 2.

Remark 1. In a solution to $\phi(I)$ with congestion 1 the following hold for every $i \in [q]$.

- If Case 1 holds for d_i, at least one overflow machine must be reserved exclusively for the placeholder job x_i^1. Therefore at most one dummy block can be assigned to an overflow machine.
- If Case 2 holds for d_i, the placeholder jobs must be assigned on at least two overflow machines. No more than a single dummy job may be assigned to an overflow machine that carries a placeholder.

Lemma 3. *If I is a "YES"-instance of* Bounded 3D-Matching*, then $\phi(I)$ is a 1-feasible instance of* SMSTC.

Proof. For all $i \in [q]$ perform the following. Let $\tau = (a_i, b_j, c_k)$ be the triple in the matching that contains element a_i. For all $r \in [f']$ assign the jobs b_j^r and c_k^r to the machine $m_i^p \in M_i$ that corresponds to the triple τ. The dummy jobs of block F_i^p are assigned to the overflow machine OF_i^p. The remaining dummy jobs $F_i^{p'}$, $p' \neq p$, are assigned each to their corresponding machine in M_i. There are $|M_i| - 1$ available overflow machines and we can schedule the d_i placeholders on them. □

Define ρ so that the following relations are satisfied

$$2 \cdot \rho < f \text{ and } \rho \cdot f < f'/3. \tag{12}$$

Lemma 4. *If I is a "NO"-instance of* Bounded 3D-Matching*, then $\phi(I)$ is not a ρ-feasible instance of* SMSTC.

Proof. Assume to the contrary that there is a ρ-feasible assignment σ. Fix a $j \in [q]$. Consider the jobs b_j^r, $r \in [f']$. By the structure of the instance I there are at most 3 triple-machines to which these f' jobs can be assigned. Therefore in σ there is a machine that carries at least $f'/3$ jobs from B_j. By (12) a triple machine that carries even one dummy job can tolerate a total of at most $\rho \cdot f < f'/3$ jobs. Therefore among the triple-machines that can accept the jobs of B_j at least one must have no dummy assigned. The above holds for all j, and in each triple-machine m only jobs from the same set $B_{m(j)}$ can be assigned. Hence there must be at least q machines in M that are empty from dummy so that each receives a job from a distinct B_j, $j \in [q]$.

Claim. In a ρ-feasible assignment σ, for every $i \in [q]$, exactly one triple-machine in group M_i is empty from dummy.

Proof (of claim). We have shown that at least q machines in M must be empty from dummy. We will show that for each i at most one machine in M_i can be empty from dummy. This will establish the claim. We distinguish two cases.

Case 1: $d_i = 1$. Assume that in σ both machines m_i^1, m_i^2 in M_i are empty from dummy. Then the two overflow machines $\mathrm{OF}_i^1, \mathrm{OF}_i^2$ take each one block of dummy jobs. The placeholder job x_i^i must live on the same machine with f other jobs. By (12) this incurs a congestion larger than ρ, a contradiction.

Case 2: $d_i = 3$. Assume that in σ at least two among the three machines in M_i are empty from dummy. Then at least two overflow machines $\mathrm{OF}_i^p, \mathrm{OF}_i^{p'}$ take each one block of dummy jobs. At least one of the three placeholder jobs x_i^1, x_i^2, x_i^3 must live in σ on the same machine with f other jobs. By (12) this incurs a congestion larger than ρ, a contradiction. □

By the claim, exactly one machine from each of the q groups M_i must be empty from dummy. Let M' be the set of these machines. Each machine in M' gets at least one member from a distinct B_j. M' induces a 2D perfect matching of $A \times B$. Similarly, a triple machine that carries a dummy job can tolerate at most $\rho \cdot f < f'/3$ jobs from C_k, for any $k \in [q]$. Since σ is ρ-feasible, for every $k \in [q]$ at least one job from C_k is assigned to a machine that is empty of dummy, i.e., to a machine of M'. Clearly, no two jobs from different C_k, $C_{k'}$ sets, with $k \neq k'$, can appear on the same triple-machine. The q machines of M' that are empty from dummy induce a feasible 3D-Matching of $A \times B \times C$. We have reached a contradiction. As long as f' is larger than a suitable constant, setting $\rho = \frac{\sqrt{f'}}{4}$ and $f = \lceil \sqrt{f'}/2 \rceil + 1$ satisfies (12). □

We conclude that unless $\mathsf{P} = \mathsf{NP}$ there is no polynomial-time algorithm that on input $\phi(I)$ can output a solution with congestion at most $\sqrt{f'}/4$ times the optimum. Given that the number n of jobs in $\phi(I)$ is equal to $2f'q + f \cdot |M| + \sum_{i=1}^{q} d_i$ and that $q \leq |M| \leq 9q$, we have that $n = \Theta(f'q)$. To keep the reduction polynomial-time it must be that $f' = O(q^c)$ for some constant $c > 0$. In other words, $f' = n^{1-\varepsilon}$ for an arbitrary constant $\varepsilon > 0$ of our choice.

Theorem 7. *For any constant $\varepsilon > 0$, there is no polynomial-time $(n^{1/2-\varepsilon})$-approximation algorithm for* SMSTC*, unless $\mathsf{P} = \mathsf{NP}$. The result holds even when each job j has the same tolerance capacity on each machine in $\mathcal{M}(j)$.*

References

1. Bansal, N., Sviridenko, M.: The Santa Claus problem. In: Proceedings of the 38th annual ACM symposium on Theory of Computing (STOC). pp. 31–40 (2006)
2. Călinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a monotone submodular function subject to a matroid constraint. SIAM J. Comput. **40**(6), 1740–1766 (2011)
3. Carr, R.D., Vempala, S.S.: Randomized metarounding. Random Struct. Algorithms **20**(3), 343–352 (2002)
4. Chekuri, C., Kumar, A.: Maximum coverage problem with group budget constraints and applications. In: APPROX-RANDOM. LNCS, vol. 3122, pp. 72–83. Springer (2004)
5. Chekuri, C., Vondrák, J.: Randomized pipage rounding for matroid polytopes and applications. CoRR **abs/0909.4348** (2009)
6. Eisemann, K.: The trim problem. Management Science **3**(3), 279 – 284 (1957)
7. Feige, U.: A threshold of ln n for approximating Set Cover. J. ACM **45**(4), 634–652 (1998)
8. Ficker, A.M.C., Spieksma, F.C.R., Woeginger, G.J.: The transportation problem with conflicts. Ann. Oper. Res. **298**(1), 207–227 (2021)
9. Fleischer, L., Goemans, M.X., Mirrokni, V.S., Sviridenko, M.: Tight approximation algorithms for maximum separable assignment problems. Math. Oper. Res. **36**(3), 416–431 (2011)
10. Gandhi, R., Khuller, S., Parthasarathy, S., Srinivasan, A.: Dependent rounding and its applications to approximation algorithms. J. ACM **53**(3), 324–360 (2006)
11. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York (1979)
12. Gilmore, P.C., Gomory, R.E.: A linear programming approach to the cutting-stock problem. Operations Research **9**, 849 – 859 (1961)
13. Jain, K., Mahdian, M., Salavatipour, M.R.: Packing steiner trees. In: Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 12-14, 2003, Baltimore, Maryland, USA. pp. 266–274. ACM/SIAM (2003)
14. Jansen, K., Rohwedder, L.: A quasi-polynomial approximation for the restricted assignment problem. SIAM J. Comput. **49**(6), 1083–1108 (2020)
15. Karmarkar, N., Karp, R.: An efficient approximation scheme for the one-dimensional bin-packing problem. In: Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science (FOCS). pp. 312–320 (1982)
16. Lenstra, J.K., Shmoys, D.B., Tardos, E.: Approximation algorithms for scheduling unrelated parallel machines. Mathematical Programming A **46**, 259–271 (1990)
17. Minerva, R., Lee, G.M., Crespi, N.: Digital twin in the IoT context: A survey on technical features, scenarios, and architectural models. Proceedings of the IEEE **108**(10), 1785–1824 (2020)
18. Oteafy, S.M.A.: Resource augmentation in heterogeneous internet of things via uavs. In: 2021 IEEE Global Communications Conference (GLOBECOM). pp. 1–6 (2021)
19. Rothvoss, T.: Better bin packing approximations via discrepancy theory. SIAM J. Comput. **45**(3), 930–946 (2016)
20. Saha, B., Srinivasan, A.: A new approximation technique for resource-allocation problems. Random Struct. Algorithms **52**(4), 680–715 (2018)
21. Sharma, A., Kosasih, E.E., Zhang, J., Brintrup, A., Calinescu, A.: Digital twins: State of the art theory and practice, challenges, and open research questions. CoRR **abs/2011.02833** (2020)
22. Srinivasan, A.: Distributions on level-sets with applications to approximation algorithms. In: Proceedings of 42nd IEEE Annual Symposium on Foundations of Computer Science (FOCS). pp. 588–597 (2001)
23. Svensson, O.: Santa claus schedules jobs on unrelated machines. SIAM J. Comput. **41**(5), 1318–1341 (2012)
24. Verma, S., Kawamoto, Y., Fadlullah, Z., Nishiyama, H., Kato, N.: A survey on network methodologies for real-time analytics of massive IoT data and open research issues. IEEE Communications Surveys Tutorials **19**(3), 1457–1477 (2017)

25. Yates, R.D., Sun, Y., Brown, D.R., Kaul, S.K., Modiano, E., Ulukus, S.: Age of information: An introduction and survey. IEEE Journal on Selected Areas in Communications **39**(5), 1183–1210 (2021)