Digital Twin Model Selection for Feature Accuracy

Hong Chen, *Student Member, IEEE*, Terence D. Todd, *Life Member, IEEE*, Dongmei Zhao, *Senior Member, IEEE*, and George Karakostas

***Abstract*—Digital twins (DTs) can be used to represent the behavior of real physical systems (PSs) in their interaction with other objects. Each DT periodically communicates with its PS and uses these updates to implement *features* that reflect the real behavior of the PS. A given feature can be implemented using different *models* that create the feature with differing levels of system accuracy. In this article, we study the DT model selection problem, where the DTs of multiple PSs are hosted at an execution server (ES). The objective is to maximize the minimum feature accuracy for the requested features by making appropriate model selections subject to the synchronization and ES execution constraints. The model selection problem is first formulated as an NP-complete integer program. It is then decomposed into multiple subproblems, each consisting of a modified Knapsack problem. A polynomial-time approximation algorithm is proposed using dynamic programming to solve it efficiently, by violating its constraints by at most a given factor. A generalization of the model selection problem is then given and an approximation algorithm using relaxation and dependent rounding is proposed to solve the problem efficiently with guaranteed constraint violations. A variety of simulation results are presented that demonstrate the excellent performance of the proposed solutions.**

***Index Terms*—Digital twins (DTs), DT feature accuracy, DT model selection, Internet of Things.**

I. Introduction

A DIGITAL twin (DT) is a virtual representation of a real physical system (PS), which can represent and extend the system's behavior when interacting with external applications [1]. This enables applications to interact with the DT, rather than having them each communicate with the PS directly. By maintaining synchronization with the PS over time, a DT can also provide information and enable features that use historical PS data, which would typically be unavailable otherwise [2].

There are many practical use cases for DTs, including those in industrial manufacturing such as in system reconfiguration, predictive maintenance, optimization, and consistency checking. These illustrate the benefits of the DT concept throughout the entire product life cycle [3]. DTs have also been successfully applied in many other areas, such as aviation [4], [5], healthcare and telemedicine [6], [7], [8], smart homes [9], [10], and smart cities [11], [12], [13].

DTs can provide *features* that are derived from and reflect the behavior of its PS [5]. When one of these features is made available to an external application, there is an associated level of accuracy compared to what would be possible in the ideal case [1], [14]. This deviation is referred to as *similarity* or *accuracy*, which is an important performance indicator [1], [5], [14], [15]. When feature accuracy is the highest, a feature is indistinguishable from the best that would be ideally possible. In practical situations, however, it is often difficult or expensive to realize this level of similarity. Examples of this are where a "noisy" version of PS state information may be used as DT input, or when the feature generation includes approximations to reduce DT computation, so that real-time performance is improved [1].

Different DT feature accuracies are defined by different DT *models*, which describe how PS inputs are processed so that the feature can be generated. As one would expect, features with higher levels of accuracy will typically use models that require more input data from the PS and higher levels of computation at the DT [16], [17], [18], [19], [20]. For example, the work in [16] proposes a DT approach that enables Accuracy-as-a-Service for resistance-based electrical sensors. In order to reduce costs, DT models with differing feature accuracy are used in the control of a production system where the sensors are used.

Applications that require features from multiple DTs may request different feature accuracy levels. In this article, we consider the case where the DTs are hosted at the same execution server (ES). Each DT communicates with its associated PS, so that updates in the PS state can be incorporated into the features provided by the DT. We assume that this communication happens periodically and is referred to as *synchronization*. When a synchronization update occurs, the updating process involves processing at the ES so that any features can be updated and made available to the requesting applications. The amount of processing needed, and the amount of data transferred when synchronization occurs is a function of the level of accuracy associated with the model and the feature that is being provided. Since the DTs are co-located at the same ES, the synchronization update processing must share the ES execution capacity, and this can limit the achievable accuracy of the provided features.

The objective of this article is to provide the best level of accuracy for a given set of feature requests by selecting an appropriate set of DT models. This is referred to

Manuscript received 25 July 2023; revised 25 September 2023; accepted 30 October 2023. Date of publication 6 November 2023; date of current version 26 March 2024. The work of Terence D. Todd, Dongmei Zhao, and George Karakostas was supported by the Natural Sciences and Engineering Research Council of Canada. *(Corresponding author: Hong Chen.)*

Hong Chen, Terence D. Todd, and Dongmei Zhao are with the Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON L8S 4L8, Canada (e-mail: chenh151@mcmaster.ca; todd@mcmaster.ca; dzhao@mcmaster.ca).

George Karakostas is with the Department of Computing and Software, McMaster University, Hamilton, ON L8S 4L8, Canada (e-mail: karakos@mcmaster.ca).

Digital Object Identifier 10.1109/JIOT.2023.3330411

as the *DT model selection problem* and is done using a max–min criterion, i.e., the objective is to maximize the minimum provided feature accuracy among the requested features, subject to the synchronization and ES execution constraints. We first formulate the problem as an integer program, which is reduced to an NP-complete feasibility problem. We then decompose it into multiple subproblems and show that each subproblem is a modified Knapsack problem. A polynomial-time approximation algorithm is proposed using a dynamic programming (DP) fully polynomial-time approximation scheme (FPTAS) to solve it efficiently, by violating its constraints by a known factor. A generalization of the model selection problem is then given and an approximation algorithm using relaxation and dependent rounding is proposed to solve the problem efficiently with guaranteed small constraint violations. A variety of simulation results are presented that demonstrate the excellent performance of the proposed algorithms.

The main contributions of this article are summarized as follows.

1) A DT model selection problem is introduced. The objective is to make model selections that maximize the minimum provided accuracy among the requested features, subject to the PS/DT synchronization uploading and execution requirements. The problem is shown to be NP-complete.
2) The model selection problem is decomposed into multiple subproblems, each consisting of a modified knapsack problem. A polynomial-time approximation algorithm is proposed using DP (FPTAS) to solve it efficiently by violating its constraints by at most a given factor.
3) A generalization of the model selection problem is given and an approximation algorithm is proposed using relaxation and dependent rounding to solve it efficiently with guaranteed small constraint violations.
4) A variety of simulation results are presented that demonstrate the excellent performance of the proposed algorithms. It is verified that our proposed FPTAS algorithm is highly efficient when the system size is large. The approximate solution can be obtained efficiently by violating its constraints by no more than a given factor. We found that the proposed solution to the generalization achieves close-to-optimum feature accuracy and its constraint violation can be reduced by running additional rounds of the dependent rounding.

The remainder of this article is organized as follows. In Section II, the prior work most related to this article is reviewed. The system model and problem formulation is then described in Section III. Following this in Section IV, the polynomial-time approximation algorithm using DP (FPTAS) is proposed. In Section V, a generalization of the model selection problem is given and an approximation algorithm using relaxation and dependent rounding is proposed. In Section VI, simulation results demonstrate that the proposed solutions are given. Finally, we present the conclusions of the work in Section VII.

II. Related Work

An increasing amount of work has considered the feature similarity/accuracy provided by a DT as an indicator of DT performance [5]. For example, the work in [17] proposes a method for DT-driven defect class recognition using a two-level lifelong deep learning strategy to detect and recognize novel classes. The proposed method has been shown to have a higher recognition accuracy for new defect classes compared to other pretrained DT models. It can be seen from this work that different DT models created by different twinning methods can provide different accuracy levels for defect recognition. Reference [18] gives an exploratory analysis of the geometric accuracy of DTs generated for existing infrastructure using point clouds in the system operation and maintenance stage, especially for structural health monitoring purposes. A level of geometric accuracy (LOGA) is introduced as a measure of the DT quality. The work discussed above motivates the feature model selection problem considered in this article.

The work in [19] investigates a DT approach for improving estimation accuracy in the dynamic thermal rating of transmission lines. Compared to existing physics-based standards, the estimation accuracy can be improved by adopting a data-driven DT using machine learning for the physical sensor data and the conductor temperature. The work in [19] also suggests another advantage of the DT, i.e., dimensionality reduction. It considers which sensor measurements have a significant role through feature selection [21], which provides the operator with meaningful information in terms of sensor importance and reduces the amount of data collected without negatively impacting the DT performance. This work also provides motivation for DT model selection based on feature accuracy that is considered in this article.

The work in [22] integrates DTs with vehicular edge computing (VEC) networks to implement adaptive network management and scheduling. It further proposes a DT-empowered VEC offloading problem to minimize the total offloading latency, subject to deadline requirements and ES computation constraints. Reference [2] proposes a wireless DT edge network framework and formulates the adaptive edge association problem considering the placement and migration of DTs. The work in [23] proposes a secure and latency-aware DT-assisted resource scheduling algorithm for a 5G edge computing-empowered distribution grid to jointly optimize access scheduling, power control, and computational resource allocation. In the above work, the DTs of network elements are used for network management and resource allocations. Although the quality of the DTs directly affects the service quality to network users, the amount of network resources required to maintain the required quality of DTs has not been well studied. Unlike the above work, in this article, we study the best DT feature accuracy obtained using model selection, subject to resource constraints.

III. System Model and Problem Formulation

We consider a network where an ES hosts the DTs of multiple PSs, as shown in Fig. 1. As is typical for DTs, each

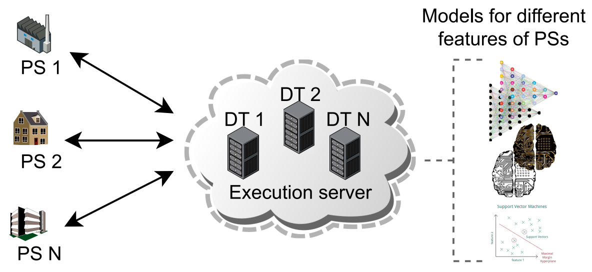


Fig. 1. System model.

PS collects its own status data and periodically uploads the data to the DT, which then processes the data so that the different features implemented at the DT are synchronized with the PS state.

Let N be the total number of PSs in the system and K be the total number of features. Define $\beta_{i,k} \in \{0, 1\}$ as a binary variable representing the demand of PS i for feature k: if $\beta_{i,k} = 1$, PS i requires feature k; otherwise, $\beta_{i,k} = 0$. Let $T_{i,k}$ be the data refreshing period for feature k of PS i. There are M_k models that can be used at the DT to obtain the state of feature k, each requiring different amounts of input data from the PS and computation load at the ES, and resulting in different accuracy to reflect the feature status of the PS. Let $\mathcal{I}, \mathcal{K}$, and $\mathcal{M}_k$ be the sets of PSs, features, and models for feature k, respectively.

Let $\Phi_{i,k,m}$ be the achieved accuracy for feature k of DT i when it is realized using model m and $s_{i,k,m}$ be the corresponding amount of input data for the model. Let $x_{i,k,m} \in \{0, 1\}$ be the decision variable indicating whether the DT of PS i chooses to use model m to realize feature k. Note that $x_{i,k,m} = 0$ if $\beta_{i,k} = 0$. We assume that the requested feature k of PS i can be realized by one and only one model at its DT, i.e.,

$$\sum_{m=1}^{M_k} x_{i,k,m} = 1 \quad \forall i, k : \beta_{i,k} = 1. \tag{1}$$

When feature k is implemented by the ith DT (i.e., $\beta_{i,k} = 1$), the achieved accuracy for this feature is

$$\Psi_{i,k} = \sum_{m=1}^{M_k} x_{i,k,m}\Phi_{i,k,m}. \tag{2}$$

Our objective is the maximization of the minimum accuracy for all features of all DTs, i.e., $\max_x \min_{i,k:\beta_{i,k}=1} \Psi_{i,k}$, while respecting computational and data transmission constraints, as well as the periodicity of DT updates.

More specifically, when the ith PS uploads data to the ES, the data transmission rate R_i should be at least the amount required for the timely transmission of the input data needed by all the feature models, i.e.,

$$R_i \geq \sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{s_{i,k,m}}{T_{i,k}} x_{i,k,m}. \tag{3}$$

Let F be the computation capacity of the ES in the number of CPU cycles per second and $f_{i,k,m}$ denote the number of CPU cycles needed by the ith DT in order to process feature k using model m. The following capacity constraint must hold:

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{f_{i,k,m}}{T_{i,k}} x_{i,k,m} \leq F. \tag{4}$$

TABLE I
SUMMARY OF NOTATION

Notation	Definitions	Units
$\mathcal{I}$	Set of PSs, $\lvert\mathcal{I}\rvert = N$	
$\mathcal{K}$	Set of features, $\lvert\mathcal{K}\rvert = K$	
$\mathcal{M}_k$	Set of models for feature k, $\lvert\mathcal{M}_k\rvert = M_k$	
$\beta_{i,k}$	Demand of PS i for feature k	
$T_{i,k}$	Data refreshing period for feature k of PS i	sec
$\Phi_{i,k,m}$	Achieved accuracy for feature k of DT i using model m	
$s_{i,k,m}$	Amount of input data required by model m to realize feature k of DT i	bits
$f_{i,k,m}$	Computation load needed by DT i to process feature k using model m	CPU cycles
$x_{i,k,m}$	Decision variable indicating whether DT i uses model m for feature k	
$\Psi_{i,k}$	Achieved accuracy for feature k of DT i	
R_i	Data transmission rate of PS i	bits/s
F	Computation capacity of the ES	CPU cycles/sec

Table I gives a summary of the notation used.

As a result of the discussion above, we obtain the following integer programming (IP) formulation of the problem where $\mathbf{x}_\beta = [x_{i,k,m} \quad \forall i, k, m \text{ and } \beta_{i,k} = 1]$:

$$\max_{\mathbf{x}_\beta} \min_{i,k:\beta_{i,k}=1} \Psi_{i,k} \text{ s.t.} \tag{IP}$$

$$\sum_{m=1}^{M_k} x_{i,k,m} = 1 \quad \forall i, k : \beta_{i,k} = 1 \tag{5}$$

$$\sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{s_{i,k,m}}{T_{i,k}} x_{i,k,m} \leq R_i \quad \forall i \tag{6}$$

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{f_{i,k,m}}{T_{i,k}} x_{i,k,m} \leq F \tag{7}$$

$$x_{i,k,m} \in \{0, 1\} \quad \forall i, k, m. \tag{8}$$

Equivalently, the problem can be formulated as follows:

$$\begin{aligned}
\max_{\mathbf{x}_\beta,\tau} \ & \tau \text{ s.t.} \\
& \Psi_{i,k} \geq \tau \quad \forall i, k : \beta_{i,k} = 1 \\
& \sum_{m=1}^{M_k} x_{i,k,m} = 1 \quad \forall i, k : \beta_{i,k} = 1 \\
& \sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{s_{i,k,m}}{T_{i,k}} x_{i,k,m} \leq R_i \quad \forall i \\
& \sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{f_{i,k,m}}{T_{i,k}} x_{i,k,m} \leq F \\
& x_{i,k,m} \in \{0, 1\} \quad \forall i, k, m \\
& \tau \geq 0.
\end{aligned} \tag{IP'}$$

We observe that (2) implies that given i and k, the optimal $\Psi_{i,k}$ can take one of the $\Phi_{i,k,m}$ values, where $m = 1, 2, \ldots, M_k$. Therefore, the optimal τ can take one of at most $N \sum_k M_k$ values. Hence, in what follows, we will assume that we have already "guessed" (i.e., we try all possible values of) $\hat{\tau}$, which is the optimal τ. Given guessed $\hat{\tau}$, we set $x_{i,k,m} := 0$ for all i, k, m that correspond to $\Phi_{i,k,m} < \hat{\tau}$, and reduce problem (IP') to the following simpler feasibility problem:

$$\max_{\mathbf{x}_{\beta,\hat{\tau}}} \ 0 \quad \text{s.t.} \tag{FIP}$$

$$\sum_{m=1}^{M_k} x_{i,k,m} = 1 \quad \forall i, k : \beta_{i,k} = 1 \tag{9}$$

$$\sum_{k=1}^{K} \sum_{m=1}^{M_k} \frac{s_{i,k,m}}{R_i T_{i,k}} x_{i,k,m} \le 1 \quad \forall i \tag{10}$$

$$\sum_{i=1}^{N} \sum_{k=1}^{K} \sum_{m=1}^{M_k} \frac{f_{i,k,m}}{F T_{i,k}} x_{i,k,m} \le 1 \tag{11}$$

$$x_{i,k,m} \in \{0, 1\} \quad \forall i, k, m \tag{12}$$

where $\mathbf{x}_{\beta,\hat{\tau}} = [x_{i,k,m} \ \ \forall i, k, m, \ \ \beta_{i,k} = 1 \text{ and } \Phi_{i,k,m} \ge \hat{\tau}]$.

Theorem 1: Deciding whether problem (FIP) is feasible is NP-complete.

Proof: It is straightforward to see that (FIP) is in NP. To prove it is NP-complete, we reduce the KNAPSACK problem to it. Recall that the input to KNAPSACK is a set of n items, each item k with a value v_k and a weight w_k, a value V and a capacity W. KNAPSACK outputs "yes" iff there is a subset of items of total value at least V and total weight at most W. Given an instance of KNAPSACK, we construct an instance of (FIP) as follows. We set $i = 1$ (i.e., we consider a single DT) and each feature corresponds to an item (i.e., $K = n$). Each feature has two models, say 1, 2, where $x_{1,k,1} = 1$ corresponds to picking item k, while $x_{1,k,2} = 1$ corresponds to not picking it. We set $R_1 := W, F := \sum_{k=1}^{n} v_k - V$, and $T_{1,k} := 1, s_{1,k,1} := w_k, s_{1,k,2} := 0, f_{1,k,1} := 0, f_{1,k,2} := v_k \ \forall k$. Then, it is clear that (9) will either pick or not pick item k, (10) requires that the picked item has total weight no more than W, and (11) requires that the total value of the items *not* picked is at most $\sum_{k=1}^{n} v_k - V$. Hence, (FIP) is feasible iff the KNAPSACK instance is a "yes" one. ■

Since there are no prospects of solving (FIP) in polynomial time, we will propose an approximation algorithm. We observe that (FIP) can be transformed into the following optimization problem by putting the left-hand side of (11) as the objective function and keeping the remaining constraints (9), (10), and (12), i.e.,

$$\min_{\mathbf{x}_{\beta,\Theta}} \ \sum_{i=1}^{N} \sum_{k=1}^{K} \sum_{m=1}^{M_k} \frac{f_{i,k,m}}{F T_{i,k}} x_{i,k,m} \quad \text{s.t.} \tag{OIP}$$

$$\sum_{m=1}^{M_k} x_{i,k,m} = 1 \quad \forall i, k : \beta_{i,k} = 1 \tag{13}$$

$$\sum_{k=1}^{K} \sum_{m=1}^{M_k} \frac{s_{i,k,m}}{R_i T_{i,k}} x_{i,k,m} \le 1 \quad \forall i \tag{14}$$

$$x_{i,k,m} \in \{0, 1\} \quad \forall i, k, m \tag{15}$$

Algorithm 1 General Solution Method

```
1:  Sort $\{\Phi_{i,k,m}, \ \forall i,k,m\}$ in decreasing order as $\mathcal{O} = \{\Phi_1, \Phi_2, \Phi_3, \ldots\}$
2:  for guessed $\hat{\tau} = \Phi_1, \Phi_2, \Phi_3, \ldots$ do
3:      for all $i, k, m$ do
4:          if $\Phi_{i,k,m} < \hat{\tau}$ or $\beta_{i,k} = 0$ then
5:              $x_{i,k,m} = 0$
6:          end if
7:      end for
8:      $\hat{x}$ = solution of (OIP$_i$) for all $i$
9:      if (OIP$_i$) feasible $\forall i$ and $\sum_{i=1}^{N} opt$(OIP$_i$) $\le 1$ then
10:         return solution $\hat{x}, \hat{\tau}$
11:     end if
12: end for
13: return Infeasible
```

and the feasibility of (FIP) is equivalent to achieving an objective value smaller than 1 in (OIP). In turn, (OIP) can be decomposed into N subproblems, with the ith one given as follows:

$$\min_{\mathbf{x}_{\beta,\hat{\tau},i}} \ \sum_{k=1}^{K} \sum_{m=1}^{M_k} \frac{f_{i,k,m}}{F T_{i,k}} x_{i,k,m} \quad \text{s.t.}$$

$$\sum_{m=1}^{M_k} x_{i,k,m} = 1 \quad \forall k : \beta_{i,k} = 1$$

$$\sum_{k=1}^{K} \sum_{m=1}^{M_k} \frac{s_{i,k,m}}{R_i T_{i,k}} x_{i,k,m} \le 1$$

$$x_{i,k,m} \in \{0, 1\} \quad \forall k, m \tag{OIP$_i$}$$

where $\mathbf{x}_{\beta,\hat{\tau},i} = [x_{i,k,m} \ \ \forall k, m, \ \ \beta_{i,k} = 1 \text{ and } \Phi_{i,k,m} \ge \hat{\tau}]$, and the feasibility of (FIP) is equivalent to checking that $\sum_{i=1}^{N} \text{opt(OIP}_i) \le 1$, where opt$(\cdot)$ represents the optimal objective value of the corresponding problem.

To summarize, solving problem (IP) is reduced to the following steps.

1) Sort $\{\Phi_{i,k,m} \ \ \forall i, k, m\}$ in decreasing order as $\mathcal{O} = \{\Phi_1, \Phi_2, \Phi_3, \ldots\}$.
2) Starting with Φ_1, set guessed $\hat{\tau}$ equal the current element of $\mathcal{O}$.
3) Set $x_{i,k,m} = 0$ if $\Phi_{i,k,m} < \hat{\tau}$ or $\beta_{i,k} = 0$.
4) Solve (OIP$_i$) for all i.
5) If $\sum_{i=1}^{N}$ opt(OIP$_i$) ≤ 1 then return guessed $\hat{\tau}$, else (i.e., if one of the (OIP$_i$)'s is infeasible, or $\sum_{i=1}^{N}$ opt(OIP$_i$) > 1) let guessed $\hat{\tau}$ equal the next element in $\mathcal{O}$ and go back to step 3), or return *Infeasible* if the latter does not exist.

The general structure of our method is given in Algorithm 1.

IV. FPTAS FOR THE PROBLEM

Since solving problem (IP) or (IP') is NP-complete (Theorem 1), we propose a polynomial-time approximation algorithm, that will guarantee a solution τ_s of (IP') with $\tau_s \ge \tau_{\text{opt}}$, where τ_{opt} is the optimal solution, but by violating constraint (7) by a factor of at most $(1 + \varepsilon)$.

We show that subproblem (OIP$_i$) is a modification of the classic Knapsack problem, defined as follows: given a set of

n items, a value v_i and size w_i for each item i, and a knapsack of capacity W, find the subset of items of total size at most W and maximum total value. The Knapsack problem can be approximately solved by a polynomial-time algorithm using DP. We transform the subproblem (OIP$_i$) into the following modified Knapsack problem.

TYPED KNAPSACK
Input: K types of items, with M_k items of type $k = 1, 2, \ldots, K$, weight $w_{k,m}$ and value $v_{k,m}$ for each item m of type k, number W.
Output: Pick *exactly* one item of each type, so that their total weight is at most W and their total value is minimized.

Note that TYPED KNAPSACK differs from the typical Knapsack problem in its types requirement and the minimization of its total value (as opposed to maximization). Let $M_{\max} = \max_k M_k$.

A. *FPTAS for* TYPED KNAPSACK

We give a solution for TYPED KNAPSACK, based on a recursive definition of the minimum weight of items that can achieve a total value *exactly* equal to a given value V. We abuse notation a little and denote by $\mathcal{M}_k$ the set of items of type k.

Let OPT(k, V) be the minimum weight of items from types $1, \ldots, k$ that yields value exactly V, when we pick exactly one item of each type. Then, OPT(k, V) is defined by recursion (16), shown at the botttom of the page. In (16), cases 1 and 2 are the base cases when there is no item to pick. In case 3, if all items of type k have values greater than V, then OPT$(k, V) = \infty$; otherwise, in case 4, OPT(k, V) picks the item of type k achieving the minimum total weight. Note that if OPT$(k, V) = \infty$, then it is infeasible to achieve value exactly V using exactly one item from types $1, 2, \ldots, k$.

A DP algorithm that solves TYPED KNAPSACK is given in Algorithm 2, where $V_{\max} = \sum_{k=1}^{K} \max_{m\in\mathcal{M}_k} v_{k,m}$.

Unfortunately, the DP algorithm that solves TYPED KNAPSACK exactly is pseudo-polynomial, since it runs in time $O(KM_{\max}V_{\max})$ and $V_{\max}$ is pseudo-polynomial on the size of the input. However, we can apply the well-known value scaling of the classical Knapsack problem to derive an FPTAS [24] that produces a solution which exceeds the optimal total value by a factor of at most $(1+\varepsilon)$, for any constant $\varepsilon > 0$.

The FPTAS algorithm for solving the TYPED KNAPSACK problem is given in Algorithm 3, where $\theta = \varepsilon v_{\max}/K$ is defined as the scaling factor, $\varepsilon \in (0, 1]$ is the precision parameter, and $v_{\max}$ is the largest item value used by the optimal solution. In line 1, the approximation algorithm rounds all item values up into integers lying in a finite range $[0, \lceil K/\varepsilon \rceil]$, then it runs the DP algorithm on the rounded instance, and finally returns the optimal solution of the rounded instance, which is the near-optimal solution to the original instance. We assume that we have "guessed" the largest value $v_{\max}$ used by the optimal solution. The "guessing" is performed by exhaustive enumeration of all items m for all types k as the "maximum-valued" item (given in the double for-loop in the algorithm), and running Algorithm 2 for each choice (line 7), excluding all items of value greater than the guessed maximum item value (line 6); then we keep the solution of minimum total item value. This exhaustive enumeration increases the running time of our algorithm by a factor of $O(\sum_k M_k) = O(KM_{\max})$. The obtained optimal solution to the rounded instance is the near-optimal solution to the original instance.

Algorithm 2 DP Algorithm for TYPED KNAPSACK Problem

```
1:  Find $OPT(k, V)$ for $0 \le k \le K$ and $0 \le V \le V_{\max}$ using
    DP by following (16)
2:  $V^* = \infty$
3:  for $V = 0 : V_{\max}$ do
4:      if $OPT(K, V) \le W$ then
5:          $V^* = \min\{V^*, V\}$
6:      end if
7:  end for
8:  if $V^* = \infty$ then
9:      return Infeasible
10: else
11:     return $V^*$
12: end if
```

Algorithm 3 FPTAS for TYPED KNAPSACK Problem

```
1:  $\hat{v}_{k,m} = \lceil v_{k,m}/\theta \rceil$ for all $k, m$
2:  $\hat{V}^* = \infty$
3:  for $k' = 1 : K$ do
4:      for $m' = 1 : M_{k'}$ do
5:          $\hat{v}_{\max} = \hat{v}_{k',m'}$
6:          $\hat{v}_{k,m} = \infty$ for all $k, m$ with $\hat{v}_{k,m} > \hat{v}_{\max}$
7:          Run Algorithm 2 using $\hat{v}_{k,m}$ for all $k, m$, which
    returns $V^*$
8:          $\hat{V}^* = \min\{\hat{V}^*, V^*\}$
9:      end for
10: end for
11: return $\hat{V}^*$
```

Theorem 2: For any given constant $\varepsilon > 0$, the algorithm returns a feasible solution (when one exists) of total value at most $(1+\varepsilon)$OPT, where OPT is the optimal solution value and runs in polynomial time $O(K^4 {M_{\max}}^2/\varepsilon)$.

Proof: Let O be an optimal solution to TYPED KNAPSACK when we use the original values, and $v_{\max}$ the maximum item value used in O (which we have guessed). Let

$$\mathrm{OPT}(k, V) = \begin{cases} 0, & \text{if } k = 0, V = 0 \\ \infty, & \text{if } k = 0, V > 0 \\ \infty, & \text{if } k > 0, \min_{m\in\mathcal{M}_k} v_{k,m} > V \\ \min_{m\in\mathcal{M}_k : v_{k,m}\le V}\left\{w_{k,m} + \mathrm{OPT}\left(k-1, V - v_{k,m}\right)\right\}, & \text{if } k > 0, \min_{m\in\mathcal{M}_k} v_{k,m} \le V \end{cases} \tag{16}$$

$\hat{O}$ be the optimal solution obtained by the DP algorithm when run on the rounded-up values instance. Let m_k and $\hat{m}_k$ be the items of type k picked by O and $\hat{O}$, respectively. Let $|O|$ and $|\hat{O}|$ be the values achieved by these solutions when the original item values are used, i.e., $|O| = \sum_k v_{m_k}$ and $|\hat{O}| = \sum_k v_{\hat{m}_k}$. Note that $|O| \geq v_{\max}$. Then, we have

$$|\hat{O}| \leq \theta \sum_k \hat{v}_{\hat{m}_k} \tag{17}$$

$$\leq \theta \sum_k \hat{v}_{m_k} \tag{18}$$

$$\leq \sum_k v_{m_k} + K\theta \tag{19}$$

$$\leq |O| + \varepsilon v_{\max} \leq (1+\varepsilon)|O| \tag{20}$$

where (18) is due to the fact that $\hat{O}$ is an optimal solution for the rounded values. Although $v_{\max}$ is unknown, the algorithm exhaustively tries all possibilities for $v_{\max}$ and returns the solution that results in the smallest total value. Therefore, the returned solution also satisfies (20).

For the running time of the algorithm, first note that all rounded values are integers in the range $[0, \lceil K/\varepsilon \rceil]$. Therefore, $0 \leq V_{\max} \leq K^2/\varepsilon$, and the running time of the DP algorithm for a single "guess" of $v_{\max}$ is $O(K^3 M_{\max}/\varepsilon)$. Adding over all possible choices for the $v_{\max}$ item, we get a total running time of $O(K^4 {M_{\max}}^2/\varepsilon)$, which is polynomial on the size of the TYPED KNAPSACK input. ■

B. *Mapping of Subproblem* (OIP$_i$) *to* TYPED KNAPSACK

The mapping of subproblem (OIP$_i$) for a given PS-DT pair i to TYPED KNAPSACK is straightforward: each feature k defines a type, and the models in $\mathcal{M}_k$ are the items of type k. Item m of type k has weight $w_{k,m} = (s_{i,k,m}/R_i T_{i,k})$ and value $v_{k,m} = (f_{i,k,m}/F T_{i,k})$. Finally, we set $W = 1$. After the mapping, we run the FPTAS algorithm on the resulting TYPED KNAPSACK instance, and that provides us with an approximate solution of subproblem (OIP$_i$), according to Theorem 2. Algorithm 1 is modified, so that line 8 uses the approximation algorithm for (OIP$_i$), and the condition $\sum_{i=1}^N \text{opt}(\text{OIP}_i) \leq 1$ in line 9 is modified to $\sum_{i=1}^N \text{opt}(\text{OIP}_i) \leq 1+\varepsilon$.

V. GENERALIZATION WITH MODEL UPPER AND LOWER BOUNDS

In this section, we will give a generalization of the model selection problem for DTs formulated in Section III and propose an approximation algorithm to solve it efficiently with guaranteed small violation of constraints.

We introduce limitation constraints on the utilization of the models for the demanded features of DTs, i.e., integer lower and upper bounds $L_m^{\min}, L_m^{\max}$ on the times a model m for feature k can be used by different DTs, i.e.,

$$L_m^{\min} \leq \sum_{i=1}^N x_{i,k,m} \leq L_m^{\max}. \tag{21}$$

As special cases, when $L_m^{\min} = 0$, model m may not be selected by any DTs; and when $L_m^{\max} = \infty$, model m may be selected by all DTs. These extra constraints capture model accuracy requirements, such as "the high-accuracy model $m \in \mathcal{M}_k$ must be used by at least two sensors (i.e., $L_m^{\min} = 2$)," or "computationally-heavy model $m \in \mathcal{M}_k$ can be used by at most one camera (i.e., $L_m^{\max} = 1$)."

Introducing the new constraints to the original problem (IP) results in the following more general one:

$$\max_{\mathbf{x}_\beta} \min_{i,k:\beta_{i,k}=1} \Psi_{i,k} \text{ s.t.} \tag{GIP}$$

$$\sum_{m=1}^{M_k} x_{i,k,m} = 1 \quad \forall i,k : \beta_{i,k} = 1 \tag{22}$$

$$\sum_{i=1}^{N} x_{i,k,m} \geq L_m^{\min} \quad \forall k \quad \forall m \in \mathcal{M}_k \tag{23}$$

$$\sum_{i=1}^{N} x_{i,k,m} \leq L_m^{\max} \quad \forall k \quad \forall m \in \mathcal{M}_k \tag{24}$$

$$\sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{s_{i,k,m}}{T_{i,k}} x_{i,k,m} \leq R_i \quad \forall i \tag{25}$$

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{f_{i,k,m}}{T_{i,k}} x_{i,k,m} \leq F \tag{26}$$

$$x_{i,k,m} \in \{0,1\} \quad \forall i,k,m. \tag{27}$$

Using the same procedure of problem transformation for (5) in Section III, with given $\hat{\tau}$, the problem can be transformed to the following feasibility problem:

$$\max_{\mathbf{x}_{\beta,\hat{\tau}}} 0 \quad \text{s.t.} \tag{GFIP}$$

$$\sum_{m=1}^{M_k} x_{i,k,m} = 1 \quad \forall i,k:\beta_{i,k} = 1 \tag{28}$$

$$\sum_{i=1}^{N} x_{i,k,m} \geq L_m^{\min} \quad \forall k \quad \forall m \in \mathcal{M}_k \tag{29}$$

$$\sum_{i=1}^{N} x_{i,k,m} \leq L_m^{\max} \quad \forall k \quad \forall m \in \mathcal{M}_k \tag{30}$$

$$\sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{s_{i,k,m}}{R_i T_{i,k}} x_{i,k,m} \leq 1 \quad \forall i \tag{31}$$

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{f_{i,k,m}}{F T_{i,k}} x_{i,k,m} \leq 1 \tag{32}$$

$$x_{i,k,m} \in \{0,1\} \quad \forall i,k,m. \tag{33}$$

Note that the added constraints are *hard*, i.e., we cannot violate them. In addition, the equivalence to TYPED KNAPSACK does not hold anymore, and, therefore, the approximation algorithm of Section IV does not work in this case. Hence, we will propose a different approximation algorithm in order to solve (GFIP) approximately in polynomial time, but with a worse approximation guarantee, which is not surprising since we are solving a more constrained problem.

A. $O(\frac{\ln N}{\ln\ln N})$-Approximation Algorithm

In this section, we present a polynomial-time approximation algorithm, that will guarantee a solution τ_s with $\tau_s \geq \tau_{\text{opt}}$, where τ_{opt} is the optimal solution, by respecting *exactly* constraints (28)–(30), but by violating constraints (31), (32) by a factor of at most $O(\ln N/\ln\ln N)$.

We note that by relaxing constraints (33) in (GFIP) to $x_{i,k,m} \geq 0 \ \forall i,k,m$, problem (GFIP) becomes a linear programming (LP) problem. This LP-relaxed problem is referred to as Relaxed-(GFIP). When lines 8–11 in Algorithm 1 are applied to Relaxed-(GFIP) instead of (OIP$_i$), the algorithm returns solution $\tau_f \geq \tau_{\text{opt}}$, unless the relaxed problem is infeasible (and therefore, the original problem is also infeasible).

Let x_f be the optimal fractional solution of Relaxed-(GFIP) achieving τ_f, computed in polynomial time by an LP solver. Our approximation algorithm will use the *dependent rounding* of [25] to round the fractional components of x_f to values 0 or 1 with the required guarantees. The rounded solution cannot have a τ value smaller than τ_f, since Algorithm 1 guarantees that $x_{i,k,m} = 0$ when $\Phi_{i,k,m} < \tau_f$ (lines 3–7), and the rounding does not change this fact.

First, we give a high-level description of the dependent rounding procedure of [25]. Assume we are given a bipartite graph (V_1, V_2, E) with bipartition (V_1, V_2) and a value $x_{i,j} \in [0,1]$ for each edge $(i,j) \in E$. Initialize $y_{i,j} = x_{i,j}$ for each $(i,j) \in E$. Values $y_{i,j}$ will be probabilistically modified in several (at most $|E|$) iterations such that $y_{i,j} \in \{0,1\}$ at the end, at which point we will set $X_{i,j} := y_{i,j}$ for all $(i,j) \in E$, where $X_{i,j}$ are the (randomly) rounded final values for edges $(i,j) \in E$.

The iterations that modify y proceed as follows. We call an edge (i,j) *floating* if its value $y_{i,j}$ is not integral (i.e., $y_{i,j} \in (0,1)$). Let $\tilde{E} \subseteq E$ be the current set of floating edges. If $\tilde{E} = \emptyset$, the process terminates by setting $X_{i,j} := y_{i,j}$ for all $(i,j) \in E$. Otherwise, find a simple cycle or maximal path S in the subgraph $(V_1, V_2, \tilde{E})$ in $O(|V_1| + |V_2|)$ time running depth-first search (DFS). Partition the edge set of S into two alternating matchings A and B. We define

$$\alpha := \min\Big\{\gamma > 0 : \big(\exists (i,j) \in A : y_{i,j} + \gamma = 1\big) \ \vee \ \big(\exists (i,j) \in B : y_{i,j} - \gamma = 0\big)\Big\} \tag{34}$$

$$\beta := \min\Big\{\gamma > 0 : \big(\exists (i,j) \in A : y_{i,j} - \gamma = 0\big) \ \vee \ \big(\exists (i,j) \in B : y_{i,j} + \gamma = 1\big)\Big\}. \tag{35}$$

Then, we execute the following randomized step.

1) With probability $\beta/(\alpha+\beta)$ set $y_{i,j} := y_{i,j} + \alpha \ \forall (i,j) \in A$, $y_{i,j} := y_{i,j} - \alpha \ \forall (i,j) \in B$.
2) With probability $\alpha/(\alpha+\beta)$ set $y_{i,j} := y_{i,j} - \beta \ \forall (i,j) \in A$, $y_{i,j} := y_{i,j} + \beta \ \forall (i,j) \in B$.

In either case, at least one edge $(i,j) \in \tilde{E}$ will stop being floating, i.e., $y_{i,j} \in \{0,1\}$, and, therefore, after at most $|E|$ iterations or $O(|E|(|V_1| + |V_2|))$ time, all values y will become integral and the rounding process terminates. Algorithm 4 codifies the dependent rounding procedure. Note that dependent rounding

Algorithm 4 Dependent Rounding

```
1: Bipartite graph $(V_1, V_2, E)$, $x_{i,j} \in [0,1]$ for each edge $(i,j) \in E$
2: $y_{i,j} = x_{i,j}$, $\forall (i,j) \in E$; $\tilde{E} = E$
3: for all $(i,j) \in \tilde{E}$ do
4:     if $y_{i,j} \in \{0,1\}$ then
5:         $\tilde{E} := \tilde{E} \setminus \{(i,j)\}$
6:     end if
7: end for
8: while $\tilde{E} \neq \emptyset$ do
9:     Simple cycle or maximal path $S = DFS(V_1, V_2, \tilde{E})$; $S = A \cup B$ for alternating matchings $A, B$
10:    Obtain $\alpha$ and $\beta$ from (34) and (35)
11:    With probability $\beta/(\alpha+\beta)$, $y_{i,j} := y_{i,j} + \alpha$, $\forall (i,j) \in A$, and $y_{i,j} := y_{i,j} - \alpha$, $\forall (i,j) \in B$
12:    With probability $\alpha/(\alpha+\beta)$, $y_{i,j} := y_{i,j} - \beta$, $\forall (i,j) \in A$, and $y_{i,j} := y_{i,j} + \beta$, $\forall (i,j) \in B$
13:    for all $(i,j) \in \tilde{E}$ do
14:        if $y_{i,j} \in \{0,1\}$ then
15:            $\tilde{E} := \tilde{E} \setminus \{(i,j)\}$
16:        end if
17:    end for
18: end while
19: $X_{i,j} = y_{i,j}$, $\forall (i,j) \in E$
20: return $X_{i,j}$
```

is a randomized algorithm, and the final rounded solution X is random variables.

We apply this framework to the fractional solution of Relaxed-(GFIP), which can be obtained in polynomial time. The fractional solution corresponds to a bipartite graph $G = (V_1, V_2, E)$, with $V_1 = \{(i,k) : \beta_{i,k} = 1\}$, $V_2 = \{m \in \mathcal{M}_k \ \forall k\}$, and $E = \{((i,k), m) : 0 < x_{i,k,m} < 1\}$.

Property (P2) of dependent rounding in [25] states the following.

Theorem 3 (Degree-Preservation): For a vertex $u \in V_1 \cup V_2$, let $d_u = \sum_{v:(u,v)\in E} x_{u,v}$ be the fractional degree of u. Then, if $X_{u,v}$ is the rounded value of $x_{u,v}$, $X_{u,v} \in \{\lfloor d_v \rfloor, \lceil d_v \rceil\}$.

Theorem 3 ensures that the rounded solution will satisfy (28), since these are constraints on the fractional degree $d_{i,k}$ of the vertices $(i,k) \in V_1$ and $d_{i,k} = 1$. Also, it ensures the satisfaction of (29) and (30), since these constraints imply $L_m^{\min} \leq d_m \leq L_m^{\max}$ and $L_m^{\min}, L_m^{\max}$ are integers.

It remains to study by how much constraints (31), (32) are violated. If $[x_{i,k,m} \ \ \forall i,k,m]$ is the fractional solution of Relaxed-(GFIP), let

$$D^{\text{LP}} := \sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{f_{i,k,m}}{FT_{i,k}} x_{i,k,m}$$

$$R_i^{\text{LP}} := \sum_{k=1}^{K}\sum_{m=1}^{M_k} \frac{s_{i,k,m}}{R_i T_{i,k}} x_{i,k,m} \quad \forall i.$$

The application of part (i) of [25, Th. 3.1] (proved by [26]) on the (randomly) rounded values $X_{i,k,m}$ implies the following Chernoff-type bounds.

Theorem 4: The following inequalities hold:

$$\Pr\left[\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M_k}\frac{f_{i,k,m}}{FT_{i,k}}X_{i,k,m} \geq (1+\varepsilon)\right] \leq \frac{e^{\varepsilon}}{(1+\varepsilon)^{1+\varepsilon}} \quad (36)$$

$$\Pr\left[\exists i\colon \sum_{k=1}^{K}\sum_{m=1}^{M_k}\frac{s_{i,k,m}}{R_i T_{i,k}}X_{i,k,m} \geq (1+\varepsilon)\right] \leq \frac{Ne^{\varepsilon}}{(1+\varepsilon)^{1+\varepsilon}}. \quad (37)$$

Proof: Note that $0 \leq (f_{i,k,m}/FT_{i,k}), (s_{i,k,m}/R_iT_{i,k}) \leq 1 \ \ \forall i,k,m$, since, otherwise, the optimal LP solution sets $x_{i,k,m} = 0$. Also, assuming the solution of (GFIP) does not result in infeasibility, i.e., $D^{\text{LP}} \leq 1$ and $R_i^{\text{LP}} \leq 1 \ \forall i$ for some $\hat{\tau}$, the properties of dependent rounding imply that $E[\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M_k}(f_{i,k,m}/FT_{i,k})X_{i,k,m}] = D^{\text{LP}} \leq 1$ and $E[\sum_{k=1}^{K}\sum_{m=1}^{M_k}(s_{i,k,m}/R_iT_{i,k})X_{i,k,m}] = R_i^{\text{LP}} \leq 1 \ \ \forall i$. Hence, given that properties (P1) and (P3) of [25] (extended by [27, Th. 4.4]) of dependent rounding hold, the conditions of [25, Th. 3.1] are satisfied, and we have the inequalities of the theorem statement. ■

As in [25, Th. 3.2], setting $\varepsilon = O(\ln N/\ln\ln N)$ guarantees a violation of (31) and (32) by at most a factor $O(\ln N/\ln\ln N)$ with probability larger than a constant. By repeating the experiment, i.e., dependent rounding, a constant number of times, the probability of success can be made bigger than any constant, e.g., 0.75.

As a result, with a high probability, the available resources need to be augmented only by small amounts in order to render our solutions feasible for the system. Also, note that although the upper bound of $O(\ln N/\ln\ln N)$ for resource augmentation is guaranteed with high probability, the randomized nature of our algorithm suggests that we can have better results if we run it a few times and keep the best solution.

VI. Simulation Results

In this section, we present simulation results to demonstrate the performance of the proposed solutions for both the original problem and the extended problem. For comparison, we also obtain the optimum solutions using an exhaustive search. We consider that the PSs access the ES through wireless transmissions [28]. All the PSs are uniformly distributed in the circular coverage area of a base station (BS), which has the maximum coverage of 150 m. The ES is located at the BS and hosts a DT for each of the PSs. We assume that each PS is individually preallocated a channel of bandwidth w to communicate with the ES co-located at the BS. Thus, the transmission rate between PS i and the BS is $R_i = w\log_2(1 + [P_i^T g_i]/\sigma^2)$, where P_i^T and g_i are the wireless transmission power and the link gain from PS i to the BS, respectively, and σ^2 denotes the noise power at the BS receiver input. Distance-based path loss is used for the link gains and the path-loss exponent is 3. Default parameters used in the simulation are summarized in Table II, where $U[a,b]$ denotes the uniform distribution between a and b. These parameter values are similar to those used in [2], [22], and [23], and varied during the simulation. We intentionally use a wide range of parameter values based on the referenced ranges so that we can make conclusions that apply in general settings.

TABLE II
Default Parameter Settings

Parameters	Values
$T_{i,k}$	$U[1,5]$ s
$s_{i,k,m}$	$U[30,150]$ M bits
$f_{i,k,m}$	$U[50,500]$ M CPU cycles
$\Phi_{i,k,m}$	$U[0.7,1]$
F	2 GHz
P_i^{T}	0.1 W
w	10 MHz
σ^2	-120 dBm

TABLE III
Comparison of Running Time

Number of features (K)	2	4	6	8	10
Proposed solution	0.05s	0.15s	2.18s	5.12s	13.34s
Optimum solution	0.06s	0.09s	32.5s	16236s	24 days

A. FPTAS for the Original Problem

In the first set of simulation, there are six PSs, the number of features (K) for each PS is varied during the simulation, and each feature has six models. The simulations are performed on a server with Ubuntu 18.04.6 LTS, Intel Xeon CPU E5-2640 v2 @ 2.00 GHz, and 196-GB memory. Table III shows the running time of the proposed approximation solution and the optimum solution. As K increases, the size of the problem increases, and the running time of both solutions increases. For the optimum solution, the running time increases exponentially and quickly becomes prohibitively long, e.g., more than 24 days when $K = 10$. Although the running time of our proposed solution is sometimes longer than the optimum solution (but still short) when K is small (e.g., $K = 4$), it increases much slower and is significantly shorter than the running time of the optimum solution when $K = 10$. This demonstrates that our proposed solution is highly efficient when the system size is large.

In the second set of simulations, we compare the minimum achieved accuracies using the proposed FPTAS solution and the optimum, respectively. Due to the long running time of the optimum solution, the comparison can only be performed for small-size systems. In the simulation, there are three PSs, each PS requires five features, and each feature can be implemented by using one of the four models (with differing accuracies). The simulation results are averaged over 100 independent experiments, each of which is for one set of randomly generated PS locations and feature/model parameters.

Fig. 2 shows the minimum achieved accuracy of features versus the ES CPU capacity F. It can be seen that in general, the minimum achieved feature accuracy increases with the ES CPU capacity. The increase is more significant when F is relatively small and gradually becomes saturated as the wireless transmission rates eventually become the performance bottleneck. It is also seen that the minimum achieved accuracy using the proposed approximate solution can be higher than the optimum one, and the approximate solution achieves this at the price of violating constraint (7), i.e., ES CPU capacity

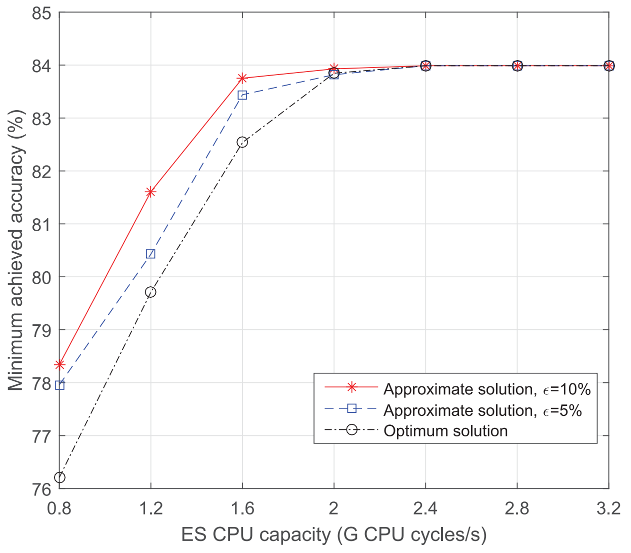


Fig. 2. Minimum achieved accuracy versus ES computation capacity.

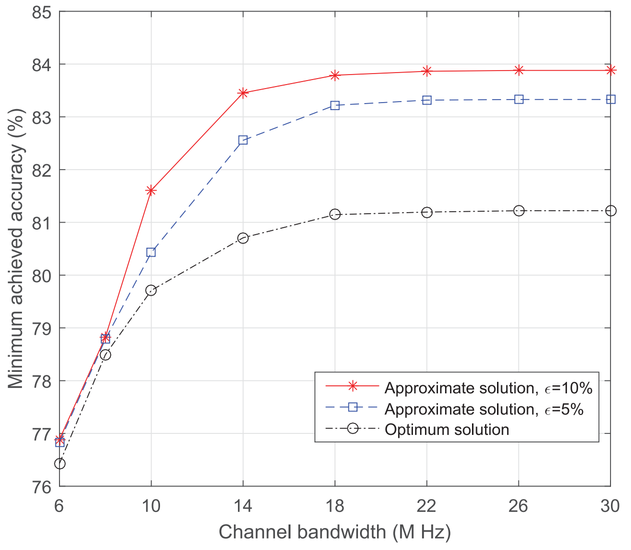


Fig. 3. Minimum achieved accuracy versus wireless channel bandwidth.

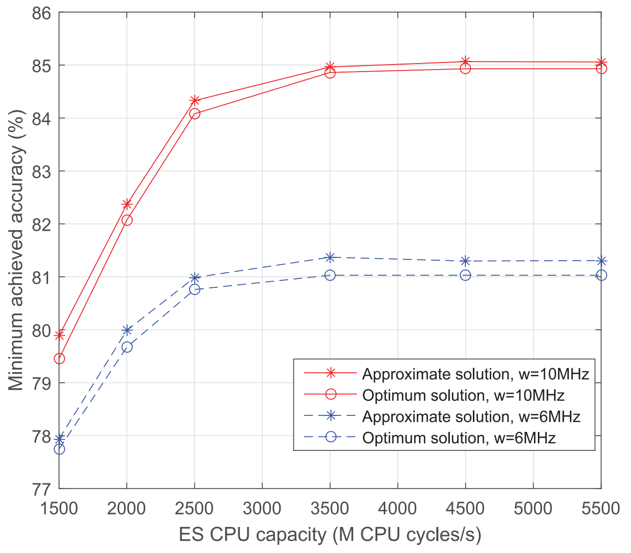


Fig. 4. Minimum achieved accuracy versus ES computation capacity (Generalization case).

constraint, by a factor of at most $(1 + \varepsilon)$. We can see that when the ES CPU capacity is relatively small, the minimum achieved feature accuracy with $\varepsilon = 10\%$ is higher than that with $\varepsilon = 5\%$, since the approximate solution can achieve a better minimum accuracy by violating the ES capacity constraint more. However, as the ES CPU capacity becomes large enough, the minimum achieved accuracies using the proposed solution are the same as the optimum solution when the proposed solution does not violate the constraints.

Fig. 3 shows the minimum achieved accuracy of features versus the wireless channel bandwidth w. The observations are similar as in Fig. 2. The minimum achieved feature accuracy increases with the wireless channel bandwidth in general and gradually becomes a constant as the bandwidth is sufficiently large and the ES computation capacity eventually becomes the performance bottleneck. It can be seen that the minimum achieved accuracy using the proposed approximate solutions is always higher than the optimum one. It is because the ES computation capacity used in this simulation is relatively small, i.e., 1.2 G CPU cycles/s, the ES CPU capacity constraint is always violated by a factor of at most $(1 + \varepsilon)$ in the approximate solutions.

B. Approximate Solution for the Generalization

In this section, we consider there are five PSs in the coverage of the BS, each PS requires five features, and each feature can be implemented by using one of the four models (with different accuracy). The input data $s_{i,k,m}$'s are randomly generated from $U[2, 200]$ M bits. The needed CPU cycles $f_{i,k,m}$'s are randomly generated from $U[5, 500]$ M CPU cycles. In this case, we assume the model with the highest accuracy of the first feature has to be used at least once. The models with the lowest accuracy of the first and the third features have to be used at most 4 times. The link gains include both the path loss and small-scale fading given as $g_i = 10^{-3}\rho_i{}^2 d_i^{-3}$, where d_i denotes the distance between PS i and the BS and ρ_i represents the additional channel small-scale fading which is assumed to be Rayleigh distributed [29], [30]. Thus, $\rho_i{}^2$ is an exponentially distributed random variable with unit mean. Note that a 30-dB average signal power attenuation is assumed at a reference distance of 1 m. The simulation results are averaged over 100 independent experiments, each of which is for one set of randomly generated PS locations and feature and model parameters.

Fig. 4 shows the minimum achieved accuracy of features versus the ES CPU capacity F, where the proposed approximate method is applied by running one round of dependent rounding. The figure shows that the minimum achieved accuracy using the proposed approximate solution is very much close to the optimum. The gap is slightly higher when the wireless channel bandwidth is smaller. Meanwhile, Fig. 5 shows the corresponding constraint violation resulted from dependent rounding in the proposed approximate solution. Both the wireless channel bandwidth constraints [i.e., (25)] and

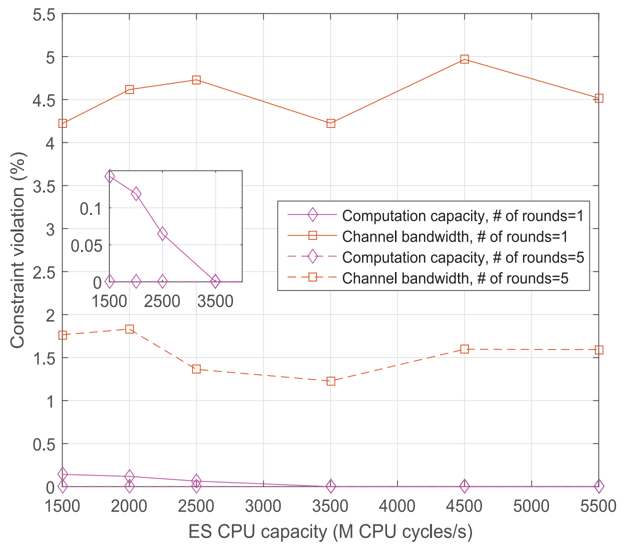


Fig. 5. Constraint violation versus ES computation capacity.

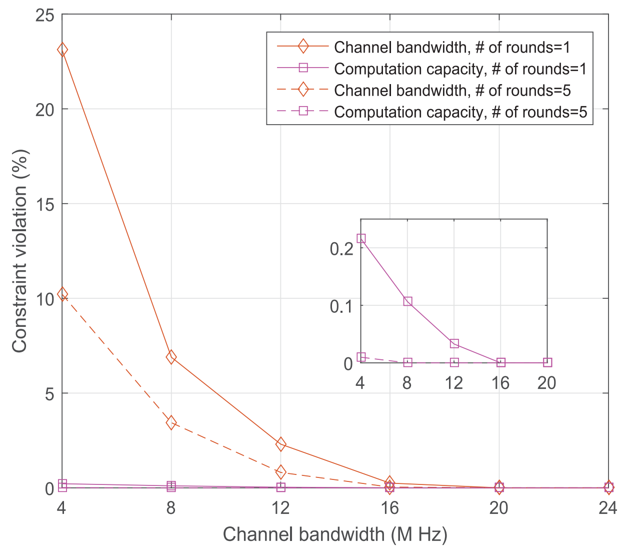


Fig. 7. Constraint violation versus wireless channel bandwidth.

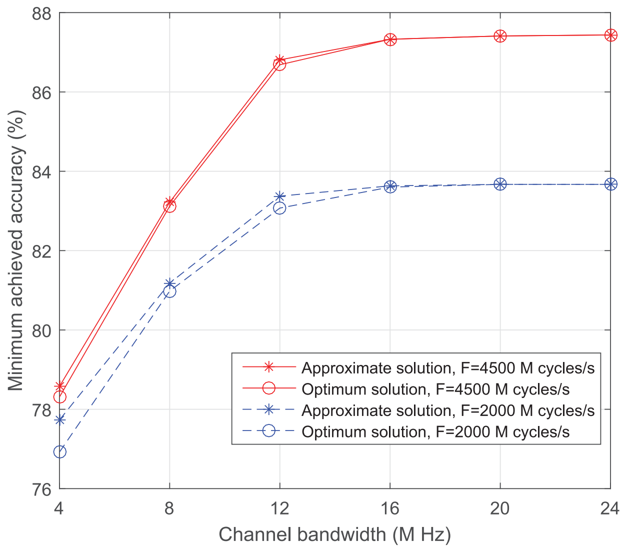


Fig. 6. Minimum achieved accuracy versus wireless channel bandwidth (Generalization case).

the computation capacity constraint [i.e., (26)] are considered. For the computing constraint, the violation is calculated as percentage changes to the right-hand value after running the algorithm. For the bandwidth constraints, the percentage change to the right-hand value is calculated for each of the N constraints and then the maximum is taken. The figure shows that as F increases, the computation constraint violation decreases in general. When F is sufficiently large, there is no violation in the computation constraint. Note that the results of the dependent rounding are random, and therefore the change of the constraint violation is not monotonic. The changes in bandwidth constraint violation are mainly due to the random effect of the dependent rounding. Fig. 5 shows for both the bandwidth and computation constraints, running additional rounds of the dependent rounding helps reduce the constraint violation. Running additional rounds of the dependent rounding also helps bring the approximate solution closer to the optimum. However, the results after running five rounds of dependent rounding are not shown in Fig. 4 because the approximate solution after running the dependent rounding for one round is already very close to the optimum.

Fig. 6 shows the minimum achieved feature accuracy versus the wireless channel bandwidth w, when one round of dependent rounding is run in the proposed approximate solution. It is seen that the minimum achieved feature accuracy using the proposed approximate solution is very close to the optimum, although the gap is slightly larger when the computation capacity is smaller. Fig. 7 shows the constraint violation as the wireless channel bandwidth changes. As the channel bandwidth increases, the bandwidth constraint violation decreases in general. When w is sufficiently large, there is no violation in the bandwidth constraint. When increasing the channel bandwidth, the computation constraint violation decreases in general, which is significantly smaller compared to the bandwidth constraint violation. However, increasing the number of rounds of running the dependent rounding helps reduce the violation in both types of constraints.

VII. Conclusion

DTs provide features that represent the real behavior of their associated PSs. This is done using models that yield differing levels of system accuracy. In this article, we have considered the DT model selection problem where the DTs of multiple PSs are hosted at the same ES. The objective is to maximize the minimum achieved accuracy among the requested features by making appropriate model selections subject to communication channel and ES resource availability. This problem was first formulated as an NP-complete integer program. This article then reduced it to a feasibility problem and decomposed it into multiple subproblems that each consist of a modified Knapsack problem. A polynomial-time approximation algorithm was proposed using DP (FPTAS) to solve it efficiently by violating the constraint by at most a given factor. A generalization of the model selection problem for DTs was then introduced. A polynomial-time approximation algorithm using relaxation and dependent rounding was proposed that finds approximate problem solutions that provide an asymptotic bound on constraint violation. A variety of simulation results were presented that demonstrate the excellent performance of the proposed algorithms. It was verified that our proposed FPTAS is highly efficient when the system size is large. The approximate solution can be obtained by violating its constraints no more than any given factor. The proposed solution to the generalization achieved close-to-optimum feature accuracy and its violation of constraints can be reduced by running additional rounds of the dependent rounding.

REFERENCES

[1] M. Vaezi et al., "Digital twins from a networking perspective," *IEEE Internet Things J.*, vol. 9, no. 23, pp. 23525–23544, Dec. 2022.

[2] Y. Lu, S. Maharjan, and Y. Zhang, "Adaptive edge association for wireless digital twin networks in 6G," *IEEE Internet Things J.*, vol. 8, no. 22, pp. 16219–16230, Nov. 2021.

[3] B. A. Talkhestani and M. Weyrich, "Digital twin of manufacturing systems: A case study on increasing the efficiency of reconfiguration," *at-Automatisierungstechnik*, vol. 68, no. 6, pp. 435–444, 2020.

[4] E. Glaessgen and D. Stargel, "The digital twin paradigm for future NASA and US air force vehicles," in *Proc. 53rd AIAA/ASME/ASCE/AHS/ASC Struct., Struct. Dyn. Mater. Conf. 20th AIAA/ASME/AHS Adaptive Struct. Conf. 14th AIAA*, 2012, p. 1818.

[5] B. R. Barricelli, E. Casiraghi, and D. Fogli, "A survey on digital twin: Definitions, characteristics, applications, and design implications," *IEEE Access*, vol. 7, pp. 167653–167671, 2019.

[6] N. Wickramasinghe et al., "A vision for leveraging the concept of digital twins to support the provision of personalized cancer care," *IEEE Internet Comput.*, vol. 26, no. 5, pp. 17–24, Sep./Oct. 2022.

[7] T. Erol, A. F. Mendi, and D. Doğan, "The digital twin revolution in healthcare," in *Proc. 4th Int. Symp. Multidiscip. Stud. Innov. Technol. (ISMSIT)*, 2020, pp. 1–7.

[8] A. Ricci, A. Croatti, and S. Montagna, "Pervasive and connected digital twins–A vision for digital health," *IEEE Internet Comput.*, vol. 26, no. 5, pp. 26–32, Sep./Oct. 2022,

[9] S. Chakrabarty, D. W. Engels, and L. Wood, "Consumer frameworks for smart environments," in *Proc. IEEE 10th Int. Conf. Consum. Electron. (ICCE-Berlin)* 2020, pp. 1–5.

[10] L. Cascone, M. Nappi, F. Narducci, and I. Passero, "DTPAAL: Digital twinning pepper and ambient assisted living," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1397–1404, Feb. 2022.

[11] A. Lee, J. Kim, and I. Jang, "Movable dynamic data detection and visualization for digital twin city," in *Proc. IEEE Int. Conf. Consum. Electron. Asia (ICCE-Asia)* 2020, pp. 1–2.

[12] L. Raes et al., "DUET : A framework for building secure and trusted digital twins of smart cities," *IEEE Internet Comput.*, pp. 1–9, 2021.

[13] O. E. Marai, T. Taleb, and J. Song, "Roads infrastructure digital twin: A step toward smarter cities realization," *IEEE Netw.*, vol. 35, no. 2, pp. 136–143, Mar./Apr. 2021.

[14] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic network virtualization and pervasive network intelligence for 6G," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 1–30, 1st Quart., 2022.

[15] F. Tang, X. Chen, T. K. Rodrigues, M. Zhao, and N. Kato, "Survey on digital twin edge networks (DITEN) toward 6G," *IEEE Open J. Commun. Soc.*, vol. 3, pp. 1360–1381, 2022.

[16] V. Stegmaier, G. Ghasemi, N. Jazdi, and M. Weyrich, "An approach enabling accuracy-as-a-service for resistance-based sensors using intelligent digital twins," *Procedia CIRP*, vol. 107, pp. 833–838, 2022.

[17] Y. Gao, X. Li, and G. Liang, "A deep lifelong learning method for digital twin-driven defect recognition with novel classes," *J. Comput. Inf. Sci. Eng.*, vol. 21, no. 3, 2021, Art. no. 031004.

[18] R. Lu, C. Rausch, M. Bolpagni, I. Brilakis, and C. T. Haas, "Geometric accuracy of digital twins for structural health monitoring," *Structural Integrity and Failure*. London, U.K.: IntechOpen, 2020.

[19] G. M. Paldino, F. De Caro, J. De Stefani, A. Vaccaro, D. Villacci, and G. Bontempi, "A digital twin approach for improving estimation accuracy in dynamic thermal rating of transmission lines," *Energies*, vol. 15, no. 6, p. 2254, 2022.

[20] S. Wang, Y.-C. Wu, M. Xia, R. Wang, and H. V. Poor, "Machine intelligence at the edge with learning centric power allocation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 11, pp. 7293–7308, Nov. 2020.

[21] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Mar. 2003.

[22] Y. Dai and Y. Zhang, "Adaptive digital twin for vehicular edge computing and networks," *J. Commun. Inf. Netw.*, vol. 7, no. 1, pp. 48–59, Mar. 2022.

[23] Z. Zhou et al., "Secure and latency-aware digital twin assisted resource scheduling for 5G edge computing-empowered distribution grids," *IEEE Trans. Ind. Informat.*, vol. 18, no. 7, pp. 4933–4943, Jul. 2022.

[24] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W.H. Freeman, 1979.

[25] R. Gandhi, S. Khuller, S. Parthasarathy, and A. Srinivasan, "Dependent rounding and its applications to approximation algorithms," *J. ACM*, vol. 53, no. 3, pp. 324–360, 2006.

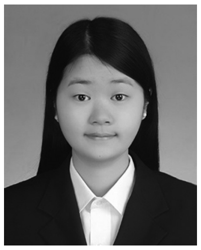
[26] A. Panconesi and A. Srinivasan, "Randomized distributed edge coloring via an extension of the Chernoff–Hoeffding bounds," *SIAM J. Comput.*, vol. 26, no. 2, pp. 350–368, 1997.

[27] B. Saha and A. Srinivasan, "A new approximation technique for resource-allocation problems," *Random Struct. Algorithms*, vol. 52, no. 4, pp. 680–715, 2018.

[28] H. Chen, T. D. Todd, D. Zhao, and G. Karakostas, "Digital twin model selection for feature accuracy in wireless edge networks," in *Proc. IEEE 34th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, 2023, pp. 1–6.

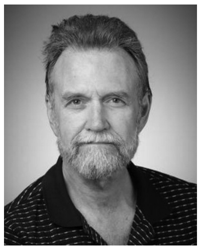
[29] H. Ju and R. Zhang, "Throughput Maximization in wireless powered communication networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 1, pp. 418–428, Jan. 2014.

[30] J. Chen, L. Zhang, Y.-C. Liang, X. Kang, and R. Zhang, "Resource allocation for wireless-powered IoT networks with short packet communication," *IEEE Trans. Wireless Commun.*, vol. 18, no. 2, pp. 1447–1461, Feb. 2019.



Hong Chen (Student Member, IEEE) received the B.E. and M.S. degrees from Chongqing University of Posts and Telecommunications, Chongqing, China, in 2016 and 2019, respectively, and the Ph.D. degree in electrical and computer engineering from McMaster University, Hamilton, ON, Canada, in 2023.

Her current research interests include mobile-edge computation, digital twin, machine learning, and resource management in wireless communication networks.



Terence D. Todd (Life Member, IEEE) received the B.A.Sc., M.A.Sc., and Ph.D. degrees in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1978, 1980, and 1984, respectively.

He is currently a Professor of Electrical and Computer Engineering with McMaster University, Hamilton, ON, Canada. He spent 1991 on research leave with the Distributed Systems Research Department, AT&T Bell Laboratories, Murray Hill, NJ, USA. During that time, he worked on the characterization of one of the first ATM switches and developed techniques for mesh network media access control. He also spent 1998 on research leave with The Olivetti and Oracle Research Laboratory (ORL), Cambridge, U.K. While at ORL, he worked on the Piconet Project which was an early embedded wireless network testbed. His research interests include local area networks, wireless communications, and the performance analysis of computer communication networks. His current work includes the design of energy-efficient wireless infrastructure, such as which operates from energy sustainable sources such as solar power.

Prof. Todd is a past Chairholder of the NSERC/RIM/CITO Chair on Wireless Networking and is a former Editor of the IEEE TRANSACTIONS ON NETWORKING and the IEEE TRANSACTIONS ON MOBILE COMPUTING. He is a Professional Engineer in the province of Ontario.

Dongmei Zhao (Senior Member, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in June 2002.

In July 2002, she joined the Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON, Canada, where she is a Full Professor. From April 2004 to March 2009, she was an Adjunct Assistant Professor with the Department of Electrical and Computer Engineering, University of Waterloo. Her current research areas are mainly in mobile computation offloading, energy-efficient wireless networking, and vehicular communication networks.

Dr. Zhao is an Associate Editor of the IEEE INTERNET OF THINGS JOURNAL. She served as an Associate Editor for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2007 to 2017. She also served as an Editor for *EURASIP Journal on Wireless Communications and Networking* and *Journal of Communications and Networks*. She is a Co-Chair of the Mobile and Wireless Networks Symposium of IEEE GLOBECOM Conference 2020, the Wireless Networking Symposium in IEEE GLOBECOM Conference 2007, the Green Computing, Networking, and Communications Symposium in International Conference on Computing, Networking and Communications 2020, and the Technical Program Committee for IEEE International Workshop on Computer Aided Modelling and Design of Communication Links and Networks 2016. She has been on the technical program committee of many international conferences in her fields. She is a Professional Engineer of Ontario.

George Karakostas received the Eng. Diploma degree in computer engineering and informatics from the University of Patras, Patras, Greece, in 1995, and the M.A.Sc. and Ph.D. degrees in computer science from Princeton University, Princeton, NJ, USA, in 1997 and 2000, respectively.

He is an Associate Professor with the Department of Computing and Software, McMaster University, Hamilton, ON, Canada. His main research interests are in the field of theoretical computer science, and, more specifically, in the design and analysis of approximation algorithms, algorithmic game theory, and their applications in practical fields, such as energy consumption and networks.