Received January 10, 2020, accepted February 25, 2020, date of publication March 2, 2020, date of current version March 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2977683

A Machine Learning Auxiliary Approach for the Distributed Dense RFID Readers Arrangement Algorithm

PEIZHI YAN, SALIMUR CHOUDHURY, AND RUIZHONG WEI
Department of Computer Science, Lakehead University, Thunder Bay, ON P7B 5E1, Canada

Corresponding author: Peizhi Yan (pyan@lakeheadu.ca)

ABSTRACT This paper is an extended version of the work published. Radio-frequency identification (RFID) is widespread in industries such as supply-chain management and logistics due to its low-cost feature. In many real-world problems, one often needs to leverage a considerable amount of RFID readers to cover a large area. Many graph-based dense RFID readers system anti-collision algorithms were proposed to address the collision problems. However, state-of-the-art collision avoidance algorithms are centralized algorithms. In a dense RFID system, the graphs generated by the centralized algorithms could be very complicated. Therefore, a centralized algorithm increases the computational workload of the central server. We proposed a distributed anti-collision algorithm based on the idea of a centralized collision avoidance algorithm called MWISBAII. In our later research, we found that due to the lack of global information, there is a gap between the performance of our distributed algorithm and the centralized MWISBAII. To narrow this gap, we introduced machine learning into the proposed algorithm. The machine learning model is an empirical model that mitigates the deficiency of the lack of global information. The experimental results show that the proposed distributed algorithm with machine learning can get almost the same performance as the centralized MWISBAII in different experimental settings.

INDEX TERMS Large scale RFID network optimization, reader coverage collision avoidance (RCCA), maximum weight independent set (MWIS), machine learning (ML).

I. INTRODUCTION

RFID is an automatic identification and data capture technology, using radio frequency electromagnetic waves to transmit signals [3]. Because RFID tags are inexpensive and extremely portable, RFID technology is an essential part of the modern IoT world [4]. Some common usages of RFID systems are product identification (to replace the traditional bar code), theft-detection, and contact-less payment [5], [6]. In addition, RFID can also be applied in positioning [7]–[9]. A normal RFID system has three major types of components: RFID reader(s), RFID tag(s), and the host system (or central computer) [10]. Both RFID readers and tags have antenna for communication. Based on the type of power source, RFID tags can be categorized into active RFID tags and passive RFID tags. The antenna of a passive RFID tag serves as both the power receiver and the signal transmitter. To read the information recorded on a passive RFID tag, the RFID reader needs to emit the electromagnetic wave to power the passive RFID tag. Whereas, an active RFID tag should have an internal power source (usually a battery) to power the antenna and the microchip. Due to this reason, the transmission range of an active RFID tag is usually much more extensive than the transmission range of a passive RFID tag. However, active RFID tags are more expensive and larger than passive RFID tags [10]. In contrary, the passive RFID tags are much smaller (often as thin as paper), so they can be installed in a passport, a luggage tag, or even a book [11], [12]. We only consider the RFID systems for passive RFID tags in this paper.

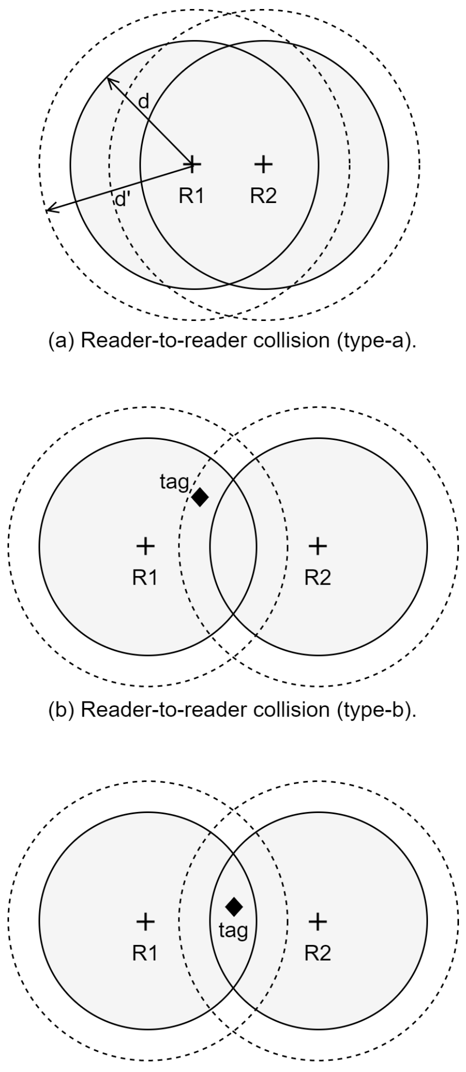
Because the RFID reader provides energy for passive RFID tags, and the valid energy transmission range is small, the coverage of a single RFID reader is limited. Therefore, in many real-world applications, one generally uses multiple readers to increase the coverage of the RFID system [13]–[16]. This kind of RFID systems is referred to as dense RFID readers systems. Tags within the activated

The associate editor coordinating the review of this manuscript and approving it for publication was Min Jia.

interrogation range of a reader can be read by the system if there is no collision. In reality, since the electromagnetic wave will not disappear beyond the interrogation range, there is an interference range, which is larger than the interrogation range [17]. In Figure 1 (a), for the reader R1, the radii of the interrogation range and the interference range are denoted by d and d', respectively. We can use a coefficient β to represent the relationship between d and d': $d' = d\beta$, where $\beta > 1.0$, means that the interference range includes the interrogation range. In this paper, we define the interference region as the region which is within the interference range but beyond the interrogation range. If we only deploy the RFID readers and tags on a two-dimensional plane, then we can abstract the interrogation and interference range to circles. To get full coverage of a field, the ranges of different readers may overlap (as shown in Figure 1), which may lead to some types of collisions.

Reader-to-reader collision (frequency interference) and reader-to-tag collision (tag interference) are two primary types of collisions in a dense RFID readers system [18]–[22]. There are two types of reader-to-reader collisions. Type-a reader-to-reader collision (see Figure 1a) occurs when a reader is within the interrogation range of another reader, and both readers are active. The radio-frequency electromagnetic wave emitted by the second reader prevents the first reader from communicating with tags within its interrogation range. Figure 1b depicts type-b reader-to-reader collision, which occurs when a tag is in the interrogation range of one reader (R1), but also in the interference region of another reader (R2). If R1 wants to read the tag and R2 is also active, then the signal of R2 may interfere with the signal of tag; meaning, R1 cannot read tag. Reader-to-tag collision occurs when one or more tags are in the activated interrogation ranges of more than one readers (see Figure 1c). In this example, if R1 and R2 attempt to communicate with the tag simultaneously, the reader-to-tag collision will occur. Furthermore, if the number of tags within an RFID reader's interrogation range is greater than the maximum number of tags that can be read by an RFID reader (we use α to represent this limit), this RFID reader cannot be activated. This scenario can also be considered as a type of collision.

The objective of most dense RFID readers system collision avoidance algorithms is to minimize the total time used for identifying (reading) all the tags without collision. Alternatively, to increase the read throughput (generally defined as the number of tags read per time slot). Based on access schemes, such algorithms are usually classified into time-division multiple access (TDMA), frequency-division multiple access (FDMA), and carrier-sense multiple access (CSMA) [23]–[28]. Considering TDMA has a relatively low implementation complexity and operational cost, most of such algorithms are TDMA-based, which can be further divided into ALOHA-based and tree-based algorithms [29]. The basic idea of access scheme-based algorithms is to reduce or eliminate collision by optimally allocating temporal or frequency resources. In [24],



(a) Reader-to-reader collision (type-a).

(b) Reader-to-reader collision (type-b).

(c) Reader-to-tag collision.

FIGURE 1. Some types of collisions in a dense RFID readers system.

Rezaie et al. proposed a centralized reader-to-reader collision avoidance protocol which combines TDMA and FDMA mechanisms. Ho et al. proposed a distributed hierarchical Q-learning (HiQ) algorithm for minimizing the collision rate of a dense RFID readers system [27]. HiQ makes the optimization by assigning different time and frequency resources to RFID readers. However, HiQ failed to consider the existence of the interference range, and not efficient to train when the network size is large [30]. A CSMA-based collision avoidance algorithm (named GENTLE) for mobile RFID networks was proposed in [28]. GENTLE assumes that the reader-to-reader collision problem is more severe than the reader-to-tag collision problem in a mobile RFID network (the RFID readers could be mobile phones). The basic idea of GENTLE is to use beacon messages to eliminate reader-to-tag collision and use the multi-channel solution to avoid the reader-to-reader collision. Reference [31] proposed a tree splitting-based anti-collision algorithm for ultra-high frequency RFID systems. This algorithm accelerates the splitting process as well as increases the system read throughput. A dense RFID readers system anti-collision protocol stack named Season was proposed in [32]. Season does not assume the existence of the interference range, which may lead to a different result than the theoretical expectation in practice. Season utilizes one phase (at the beginning) to collect data from the tags which are not within the overlapping interrogation ranges of different readers. After the first phase, Season converts the anti-collision problem to the Maximum-Weight-Independent-Set (MWIS) problem and employs two phases to selectively active some readers. Those two phases might be executed multiple iterations until all the tags have been read. The algorithm proposed in [33] requires a planned deployment of RFID readers, which makes it possible for the algorithm to get the accurate location information of each reader. Based on the interrogation and interference regions of RFID readers, this algorithm schedules the activation of readers to enable all the areas to be covered at least once at the end. Reference [34] presents a distributed approach (ADRA) for the scenarios when a central server does not exist. ADRA assumes that there could be multiple applications running in the system and issue identification requests. Based on the assumption, ADRA works in an adaptive way to make the idle readers not to participate in the coordination. A common limitation of the algorithms, as mentioned earlier, is that the readers cannot read tags within their overlapped interrogation ranges simultaneously, which causes delays. MRTI-BT addressed this issue through bit tracking [35]. Besides, MRTI-BT also prevents common tags from being identified multiple times by different readers.

In some scenarios, passive RFID tags not only serve as the object identifiers but also have some complex functionalities. For instance, the wireless sensor could be integrated into an RFID tag, which leverages the energy harvested by the tag's antenna to drive the wireless sensor module and transmit the data collected by the sensor module [36], [37]. Because the wireless sensors need to work uninterruptedly, it is challenging (if not impossible) to leverage the above-mentioned anti-collision algorithms to keep all the wireless sensors online. In this case, one needs to sacrifice some sensor nodes (deactivating some of the RFID readers could help to avoid the collision, but some RFID tags might be inaccessible by the system), and allow the whole system to enable as many sensor nodes online as possible. The problem of selectively activate or deactivate the interrogation ranges in a dense RFID readers system to allow the system to read as many tags at the same time as possible is known as reader-coverage collision avoidance (RCCA) problem [38]. Figure 2 depicts an example RFID system, where only two readers are active to enable the system to keep communicating with the maximum number of tags. One of the state-of-the-art RCCA algorithms is MWISBAII [2]. This algorithm first transforms the RCCA problem into a MWIS problem, and uses graph theory to solve the MWIS problem. In a dense RFID reader system, the graph representation of the MWIS problem could be

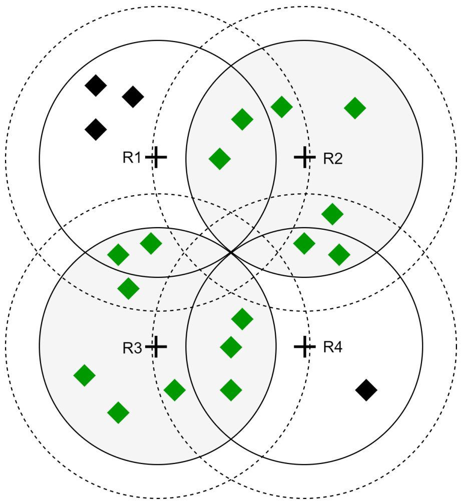


FIGURE 2. An example RFID system with four readers (R1, R2, R3, and R4). R2 and R3 are active, while R1 and R4 are off.

huge, increasing the burden on the central computer. To keep the performance of the centralized MWISBAII as much as possible and distribute the computing task into each reader, we proposed a machine learning assisted distributed RCCA algorithm. We leverage the MWISBAII to label the training data for machine learning and apply the trained model to our distributed algorithm. Different from our initial algorithm proposed in [1], the new algorithm first utilizes the trained model to predict whether to activate and deactivate some of the readers and then use our initial algorithm to handle the rest of the readers. To the best of our knowledge, no machine learning-based approach has been proposed to solve the RCCA problem. The contributions of this paper are summarized below:

1) We convert the RCCA optimization problem into a supervised learning problem. Thereby, the algorithm is more straightforward than most unsupervised reinforcement learning algorithms.
2) Our machine learning model is light-weighted and straightforward. Therefore, it can be efficiently deployed to RFID readers.
3) The proposed method is fully distributed after training, and the performance is close to the centralized algorithm without increasing the information type from one-hop to multi-hop.

The remainder of this paper is organized as follows. We state the RCCA problem in detail in Section II. In Section III, we discuss some MWIS algorithms. Because MWISBAII leverages GWMIN2 to solve the MWIS representation of the RCCA problem, we describe the GWMIN2 algorithm through a step-by-step example. Then we describe the transform and conquer algorithm MWISBAII. We review our distributed MWISBAII algorithm in Section IV. Section V presents the proposed machine learning auxiliary approach and gives an example of using this algorithm in solving the RCCA problem. Section VI presents the experimental results of the performance of MWISBAII and our distributed algorithm (with or without machine learning auxiliary). At the end (in Section VII), we conclude this paper and put forward some ideas of future work.

II. PROBLEM STATEMENT

Reader-coverage collision avoidance (RCCA) problem is about which reader(s) in a dense RFID readers system should be activated to allow the whole system read as many RFID tags without collision as possible at the same time [38].

In this paper, we suppose that the readers in the dense RFID readers system have identical technical specification such as the radius of the interrogation range d; the radius of the interference range $d' = d\beta$ ($\beta > 1.0$); and the maximum number of tags can be read by each reader (denoted by α). Assume the dense RFID readers system has N readers and M tags, then we define the set of readers as $\{r_i | 1 \leq i \leq N, i \in \mathbb{N}\}$, and the set of tags as $\{t_i | 1 \leq i \leq M, i \in \mathbb{N}\}$. For the i^{th} reader r_i, we use R_i and R'_i to represent the set of tags within its interrogation range and the set of tags within its interference range respectively. Because d' is greater than d, $R_i \subseteq R'_i$. $|R_i|$ is the number of tags in R_i, and $|R'_i|$ is the number of tags in R'_i. The result of the RCCA algorithm can be represented as a subset Υ of the set of all readers. Only the readers in Υ should be activated. The result should meet the following constraints:

1) if $r_i, r_j \in \Upsilon$, $i \neq j$, then $R_i \bigcap R'_j = \emptyset$ and $R'_i \bigcap R_j = \emptyset$;
2) if $r_i \in \Upsilon$, then $|R_i| \leq \alpha$ and $|R_i| > 0$.

We define that $T = \{R_i : r_i \in \Upsilon\}$, which is the set of tags can be read by the RFID system. The goal of the RCCA algorithm is to find a set Υ, such that $|T|$ is as large as possible. In practice, we also consider the total energy consumption of the system. Therefore, besides maximizing $|T|$, we also want to minimize the number of activated readers $|\Upsilon|$. An efficient RCCA algorithm should also derive a solution with a high T/R ratio (the number of readable tags divided by the number of activated readers): $|T|/|\Upsilon|$.

III. RELATED WORK

The purpose of the MWIS problem is to find a set of vertices for any given undirected graph, where no two vertices are adjacent in the original undirected graph and the total weight of vertices should be as large as possible. Since the MWIS problem is NP-hard [39], when the graph is complex, often we can only derive a near optimal solution.

A few simple, yet effective greedy algorithms for solving the MWIS problem are GMIN [40], GMAX [41], and GWMIN2 [42]. The GMIN algorithm repeats the following process until no vertex can be selected (the graph is empty): select a vertex of the minimum degree from the graph and put it into the set I (I is an empty set at the beginning of the algorithm), then remove this vertex and its neighbors. The GMAX algorithm deletes a vertex of the maximum degree at each step until no vertex can be deleted (no edge in the graph), then puts all the remaining vertices into the set I. The result set I is the MWIS. Sakai *et al.* [42] showed that both GMIN and GMAX can give a MWIS where the total weight is greater than or equal to $\sum_{v \in V(G)} \omega(v)/(deg(v)+1)$ ($\omega(v)$ is the weight of v, $deg(v)$ is the degree of v), while GWMIN2 can give a MWIS that the total weight is greater than or equal to $\sum_{v \in V(G)} \omega(v)^2 / \sum_{u \in neighbors(G,v)} \omega(u)$ ($neighbors(G, v)$ represents all the neighbors of v in G).

Du and Zhang [43] proposed a distributed MWIS algorithm. Their algorithm allows each node to make a partial solution, where each node broadcasts the partial solution as a message to each of its neighbors. To achieve the different trade-off between approximation accuracy and space complexity, they introduced a parameter H to lead the nodes to truncate some partial solutions before broadcasting the message. The higher the H value, the more accurate approximation their algorithm can achieve. When $H = +\infty$, the nodes will not truncate any partial solution. The problem of this algorithm is when the number of nodes is huge, the message size could be exponentially large.

A. GWMIN2 Algorithm

To find the MWIS on a given undirected graph $G = (E, V)$ (E is the set of edges, V is the set of vertices; we use the same graph representation in our proposed algorithm), the first step in GWMIN2 algorithm is to use an equation (Eq. 1) to evaluate the cost of each vertex $v \in V$.

$$cost(G, v) = \frac{\omega(v)}{\sum_{u \in neighbors(G,v)} \omega(u) + \omega(v)} \tag{1}$$

The second step in GWMIN2 is to pick the vertex v with the highest cost and then add v into the independent set I. Finally, delete v and its neighbors from G, and repeat the first and the second steps until no vertex can be selected. Set I is the solution.

In a simple example MWIS problem depicted in Figure 3, there are four vertices $\{v_1, v_2, v_3, v_4\}$ with weights $\{7, 7, 9, 6\}$ respectively. We use Eq. 1 to get the cost of each of them:

- $cost(G, v_1) = 7/23 \approx 0.3043478261$
- $cost(G, v_2) = 7/20 = 0.35$
- $cost(G, v_3) = 9/22 \approx 0.4090909091$
- $cost(G, v_4) = 3/11 \approx 0.2727272727$

Vertex v_3 has the highest cost value. Thus, we add v_3 to I, and remove v_3 and its neighbors from G. By repeating the above steps, we get the MWIS: $I = \{v_3, v_2\}$. The total weight of vertices in I is 16. Because this example is straightforward, the result is the best possible solution. However, when the graph is large, GWMIN2 can only guarantee a relatively optimal solution.

B. MWISBAII

Maximum Weight Independent Set Based Algorithm (MWISBA) [38] ideally assumes that the interference range does not exist. Thereby, in nature, it cannot detect and avoid type-b reader-to-reader collisions. Based on MWISBA, MWISBAII [2] considers the interference range, which allows the RFID system to avoid all types of collisions depicted in Figure 1. The main idea of the MWISBAII is to

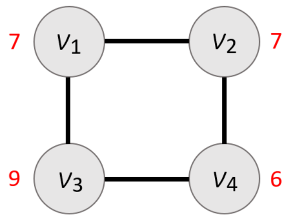


FIGURE 3. An example MWIS problem. Each circle represents a vertex; the red number beside each vertex represents the weight of that vertex.

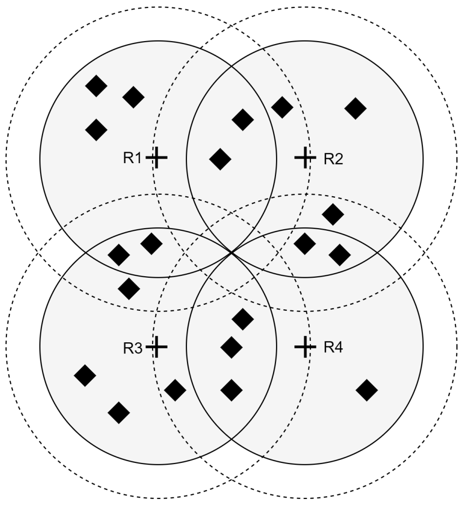


FIGURE 4. An example RFID system with four readers (R1, R2, R3, and R4).

transform the reader-coverage collision avoidance (RCCA) problem into a MWIS problem. The GEMIN2 algorithm is then used to solve this MWIS problem. Lastly, the solution of MWIS problem can be transformed back to the solution of RCCA problem.

For example (see Figure 4), there are four readers in a dense RFID reader system. Assume each reader only has one interrogation range, and the maximum number of tags that can be read by each reader is $\alpha = 10$. We use integer i to represent the i^{th} reader. In this example, $i \in \{1, 2, 3, 4\}$. To transform this RCCA problem into MWIS problem, first, the MWISBAII will append a vertex v_i to graph G if the number of tags within the interrogation range of the i^{th} reader is less than or equal to α. If there are any tags within both the interrogation range of the i^{th} reader and the interference/interrogation range of the j^{th} reader, an edge will be associated to v_i and v_j in G. Therefore, this RCCA problem will be transformed into the MWIS problem shown in Figure 3. The solution of the above MWIS problem is $I = \{v_3, v_2\}$, which was mentioned before. This means that R3 and R1 should be activated to allow the RFID system to read as many tags as possible without collision.

IV. THE DISTRIBUTED MWISBAII

To allow each RFID reader to be involved in the computation and decision process, we proposed a distributed MWISBAII in [1]. In our distributed algorithm, we assume that each reader already has the information on how many tags within its interrogation and interference range (in practice, one could use RFID positioning technology to collect that information [44]). Besides, each reader should have a local data field that contains the following components:

- A local undirected graph (or local graph) $G^* = (V^*, E^*)$.
- The status of reader (STAT) that has four possible values:
  - LOCK: The reader will not receive signal from neighbor readers.
  - OPEN: The reader is waiting for signals from neighbor readers.
  - ACTIVE: The reader is activated.
  - OFF: The reader cannot be activated.
- A sender buffer BUFFER$_{out}$.
- A receiver buffer BUFFER$_{in}$.

Readers communicate with each other through signals, which we define as a six-tuple: ($i\_, j\_$,CODE, VALUE). The main idea of the distributed MWISBAII is to let each reader build a local graph with the one-hop information. Then, let each reader compute and broadcast the cost. Readers make local decisions (whether to activate or not) regarding their local graphs. If a reader cannot decide on this iteration, it will move on to the next iteration until the decision been made.

The algorithm can be described as the following steps (suppose this reader is the i^{th} reader):

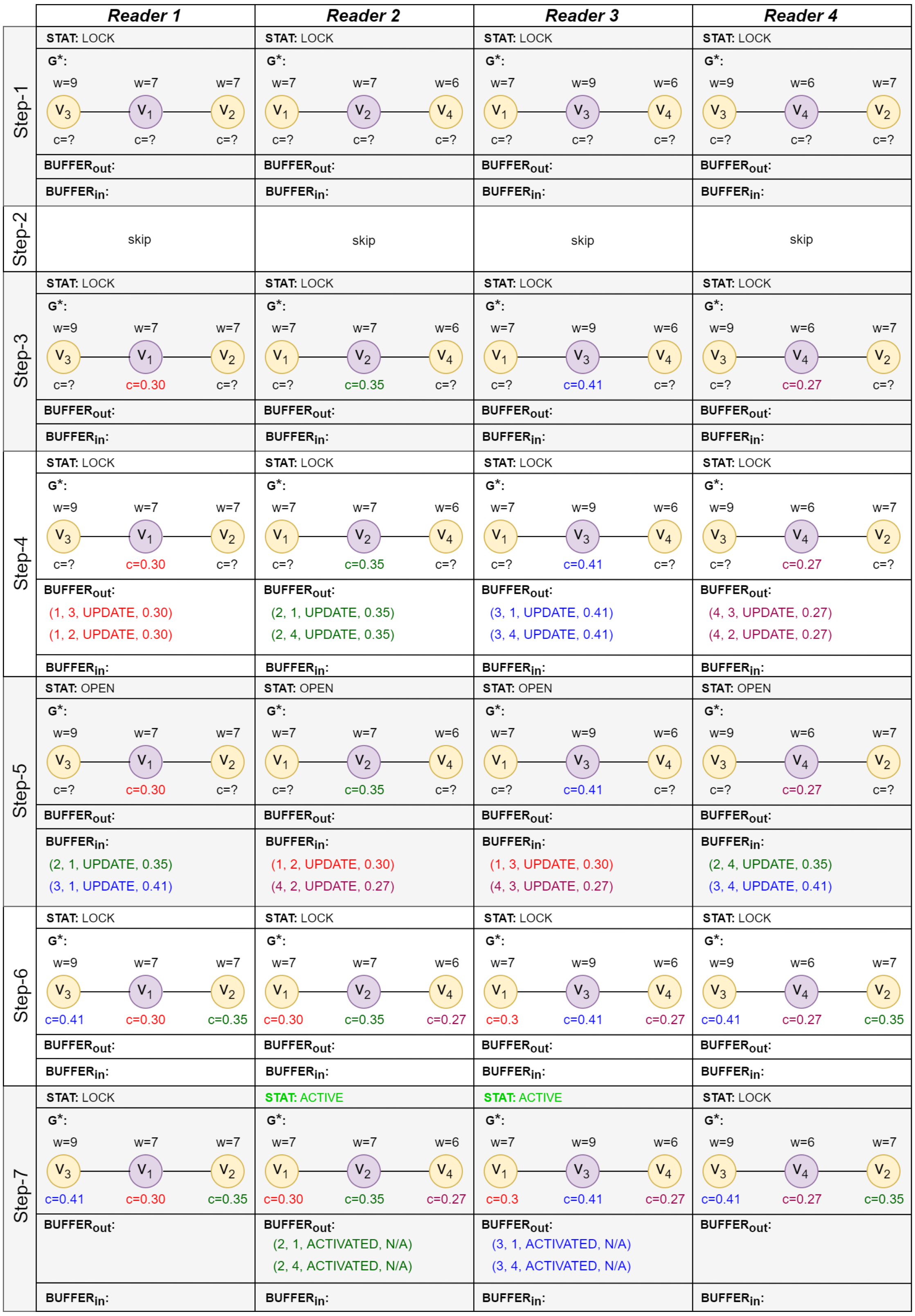
Step-1: Initialize the graph $G^* = (V^*, E^*)$. First, set STAT to LOCK. For the i^{th} reader, if $|R_i| \leq \alpha$, then we associate a vertex v_i in G^*. The weight of v_i is equivalent to $|R_i|$. We call v_i a local vertex. For all readers, other than the i^{th} reader ($\forall j \in \{j | 1 \leq j \leq N\}$): if $R'_i \bigcap R'_j \neq \emptyset$, we associate a vertex v_j (we call it non-local vertex) and an edge (v_i, v_j) to G^*. Go to step-3 (skip step-2).

Step-2: Remove all the redundant vertices in G^* (if the program just finished executing step 1, then this step will be skipped). For each non-local vertex $v_j (j \neq i, v_j \subseteq V^*)$, if the status (STAT) of the j^{th} reader is OFF, we simply remove v_j from G^*. If the status (STAT) of the j^{th} reader is ACTIVE, we remove v_j and its neighbors from G^*. Go to step-3.

Step-3: Compute the cost value of the local vertex. For each local vertex v_i, the cost value of it is calculated by Eq. 1. Go to step-4.

Step-4: Prepare the signals to be sent. For each non-local vertex v_j in G^*, if there is an edge (v_i, v_j) between it and a local vertex, we create a signal ($i\_ = i, j\_ = j$,CODE=UPDATE, VALUE= $cost(G^*, v_i)$). Afterwards, we put this signal into the sender buffer BUFFER$_{out}$. Go to step-5.

Step-5: Send and receive signals alternatively. Set the status (STAT) to LOCK. For each signal ($i\_, j\_$,CODE, VALUE) in BUFFER$_{out}$; if the $i\_^{th}$ reader's status (STAT) is ACTIVE, we send this signal to the $i\_^{th}$ reader (put into the $i\_^{th}$ reader's BUFFER$_{in}$), and remove this signal from BUFFER$_{out}$. Change the status (STAT) to OPEN. If there are non-local vertices in G^*, wait for a short period of time τ



		Reader 1	***Reader 2***	***Reader 3***	***Reader 4***
Step-1	**STAT:**	LOCK	LOCK	LOCK	LOCK
	G*: w	w=9 w=7 w=7	w=7 w=7 w=6	w=7 w=9 w=6	w=9 w=6 w=7
		V_3 — V_1 — V_2	V_1 — V_2 — V_4	V_1 — V_3 — V_4	V_3 — V_4 — V_2
	c	c=? c=? c=?	c=? c=? c=?	c=? c=? c=?	c=? c=? c=?
	$BUFFER_{out}$:				
	$BUFFER_{in}$:				
Step-2		skip	skip	skip	skip
Step-3	**STAT:**	LOCK	LOCK	LOCK	LOCK
	G*: w	w=9 w=7 w=7	w=7 w=7 w=6	w=7 w=9 w=6	w=9 w=6 w=7
		V_3 — V_1 — V_2	V_1 — V_2 — V_4	V_1 — V_3 — V_4	V_3 — V_4 — V_2
	c	c=? c=0.30 c=?	c=? c=0.35 c=?	c=? c=0.41 c=?	c=? c=0.27 c=?
	$BUFFER_{out}$:				
	$BUFFER_{in}$:				
Step-4	**STAT:**	LOCK	LOCK	LOCK	LOCK
	G*: w	w=9 w=7 w=7	w=7 w=7 w=6	w=7 w=9 w=6	w=9 w=6 w=7
		V_3 — V_1 — V_2	V_1 — V_2 — V_4	V_1 — V_3 — V_4	V_3 — V_4 — V_2
	c	c=? c=0.30 c=?	c=? c=0.35 c=?	c=? c=0.41 c=?	c=? c=0.27 c=?
	$BUFFER_{out}$:	(1, 3, UPDATE, 0.30) (1, 2, UPDATE, 0.30)	(2, 1, UPDATE, 0.35) (2, 4, UPDATE, 0.35)	(3, 1, UPDATE, 0.41) (3, 4, UPDATE, 0.41)	(4, 3, UPDATE, 0.27) (4, 2, UPDATE, 0.27)
	$BUFFER_{in}$:				
Step-5	**STAT:**	OPEN	OPEN	OPEN	OPEN
	G*: w	w=9 w=7 w=7	w=7 w=7 w=6	w=7 w=9 w=6	w=9 w=6 w=7
		V_3 — V_1 — V_2	V_1 — V_2 — V_4	V_1 — V_3 — V_4	V_3 — V_4 — V_2
	c	c=? c=0.30 c=?	c=? c=0.35 c=?	c=? c=0.41 c=?	c=? c=0.27 c=?
	$BUFFER_{out}$:				
	$BUFFER_{in}$:	(2, 1, UPDATE, 0.35) (3, 1, UPDATE, 0.41)	(1, 2, UPDATE, 0.30) (4, 2, UPDATE, 0.27)	(1, 3, UPDATE, 0.30) (4, 3, UPDATE, 0.27)	(2, 4, UPDATE, 0.35) (3, 4, UPDATE, 0.41)
Step-6	**STAT:**	LOCK	LOCK	LOCK	LOCK
	G*: w	w=9 w=7 w=7	w=7 w=7 w=6	w=7 w=9 w=6	w=9 w=6 w=7
		V_3 — V_1 — V_2	V_1 — V_2 — V_4	V_1 — V_3 — V_4	V_3 — V_4 — V_2
	c	c=0.41 c=0.30 c=0.35	c=0.30 c=0.35 c=0.27	c=0.3 c=0.41 c=0.27	c=0.41 c=0.27 c=0.35
	$BUFFER_{out}$:				
	$BUFFER_{in}$:				
Step-7	**STAT:**	LOCK	ACTIVE	ACTIVE	LOCK
	G*: w	w=9 w=7 w=7	w=7 w=7 w=6	w=7 w=9 w=6	w=9 w=6 w=7
		V_3 — V_1 — V_2	V_1 — V_2 — V_4	V_1 — V_3 — V_4	V_3 — V_4 — V_2
	c	c=0.41 c=0.30 c=0.35	c=0.30 c=0.35 c=0.27	c=0.3 c=0.41 c=0.27	c=0.41 c=0.27 c=0.35
	$BUFFER_{out}$:		(2, 1, ACTIVATED, N/A) (2, 4, ACTIVATED, N/A)	(3, 1, ACTIVATED, N/A) (3, 4, ACTIVATED, N/A)	
	$BUFFER_{in}$:				

FIGURE 5. The readers' local data fields at each step of the first iteration by applying the distributed MWISBAII algorithm on a simple example. "w" denotes vertex weight; "c" denotes vertex cost.

to receive signals. Repeat step 5 until BUFFER$_{out}$ is empty and the number of signals in BUFFER$_{in}$ equals the number of non-local vertices in G^*. Go to step-6.

Step-6: Process the signals in BUFFER$_{in}$. Set the status (STAT) to LOCK. For each signal ($i\_, j\_$, CODE, VALUE) in BUFFER$_{in}$: if the CODE = UPDATE, we update the cost value of $v_{i\_}$ to VALUE; else, if CODE = ACTIVATED, we remove the vertex $v_{i\_}$ and the vertex $v_{j\_}$ from G^* (just ignore it if $v_{j\_}$ has already been removed); else (CODE = DEACTIVATED), we simply remove $v_{i\_}$ from G^*. Go to Step-7.

Step-7: Make a local decision. If G^* is empty, change the status (STAT) to OFF, and for each non-local vertex v_j, send a signal ($i\_ = i, j\_ = j$, CODE=DEACTIVATED, VALUE=N/A) to the j^{th} reader. Following this, stop the algorithm. If G^* is not empty, find the vertex with the highest cost value. If this vertex is a local vertex v_i: change the status (STAT) to ACTIVATE, and for each non-local vertex v_j, send a signal ($i\_ = i, j\_ = j$,CODE=ACTIVATED, VALUE=N/A) to the j^{th} reader, and stop the algorithm. If the vertex with the highest cost value is not a local vertex, then go to step-2 (next iteration).

The deletion of vertices in a reader's local graph may cause the need for its neighbor readers to delete some of the vertices in their local graph. This may result in a domino effect. Due to the consideration of efficiency, we need to reduce the number of messages transmitted between readers. Therefore, in some cases, when a reader has made the decision, it will not send any signal to the other readers. This may finally cause a deadlock in the system [45]. To resolve the deadlock problem and assure a higher efficiency, we set a threshold of time limit for each reader. If the amount of time that a reader spends in any step is larger than the threshold, the reader will spontaneously set its status to OFF and stop the algorithm.

Define the average number of vertices in G^* as n. Then each step in this algorithm has a time complexity of $O(n)$, and the time complexity of each iteration is also $O(n)$. Moreover, because each reader only needs to get its one-hop neighbor readers updated, the complexity of message exchange is $O(n)$. In Figure 5, we show the execution of the distributed MWISBAII on the example in Figure 4 step by step. After the first iteration, reader 2 (R2) and reader 3 (R3) are activated. In the second iteration (which is not shown in Figure 5), reader 1 (R1) and reader 4 (R4) will receive and process the signals sent by reader 2 and reader 3. At the end of the second iteration, reader 1 and reader 4 will be deactivated.

V. THE PROPOSED MACHINE LEARNING AUXILIARY APPROACH

In practice, the tags are randomly distributed in the RFID system. Some RFID readers may have more tags, while other RFID readers may have fewer tags. This uneven distribution could be highly skewed. If the RFID readers merely make decision on their local (1-hop) information, some valuable RFID readers (contribute more tags to the whole system) may not be successfully activated. We assume that there are hidden patterns which can help the distributed algorithm to make a better decision if the geological position of each RFID reader is fixed. Which means that, with only the 1-hop local information and some empirical knowledge on the system environment, we could improve the performance of the distributed algorithm. Based on the assumption, we propose a machine learning auxiliary approach for our distributed MWISBAII algorithm previously introduced in [1].

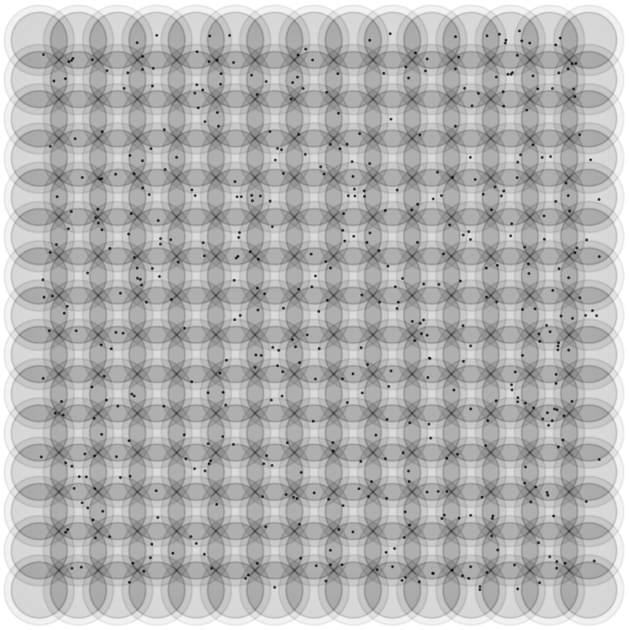


FIGURE 6. A 100m × 100m simulated area with 500 randomly assigned tags (tags are shown as dots). The interval of nearest readers is 7m; d = 5m; β = 1.25.

To implement the proposed approach, we require that each RFID reader has a neural network model and a local ego-network graph. The RFID readers could communicate with their nearby readers through a wired connection. However, if the wireless connection among RFID readers is required, the communication should leverage a channel that will not interfere with the RFID interrogation. Moreover, the wireless communication range should be at least $d + d'$. Because when the distance between two RFID readers is greater than $d + d'$, the collisions we mentioned previously will not happen, and these two readers do not need to contact each other directly.

There are two stages in the proposed approach. The first stage is the machine learning stage, which is required only when the dense RFID system is set up. This stage primarily happens on the central server because running simulations and training a neural network model require high-performance computing resources. At the end of the machine learning stage, the central server broadcasts the trained neural network weights to each reader, and each reader updates its neural network model with the received weights. The second stage is the application stage, which

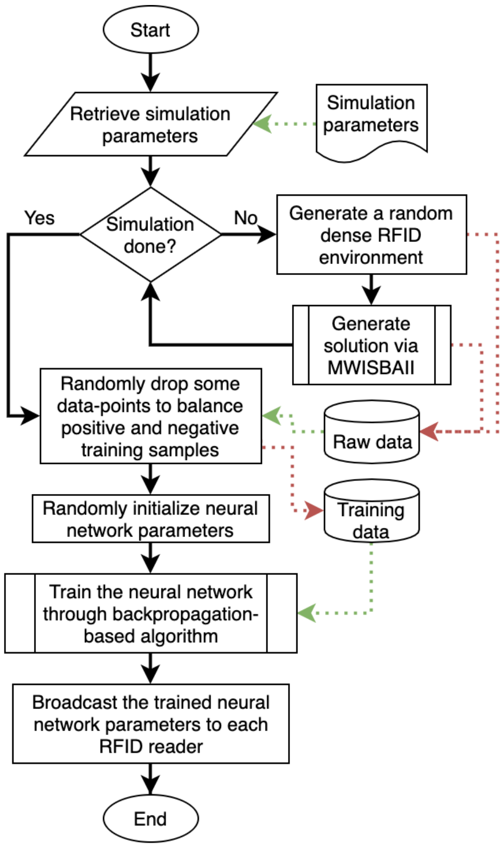

Start
Retrieve simulation parameters
Simulation parameters
Simulation done?
Yes
No
Generate a random dense RFID environment
Generate solution via MWISBAII
Raw data
Randomly drop some data-points to balance positive and negative training samples
Training data
Randomly initialize neural network parameters
Train the neural network through backpropagation-based algorithm
Broadcast the trained neural network parameters to each RFID reader
End

FIGURE 7. **The flowchart of machine learning stage.**

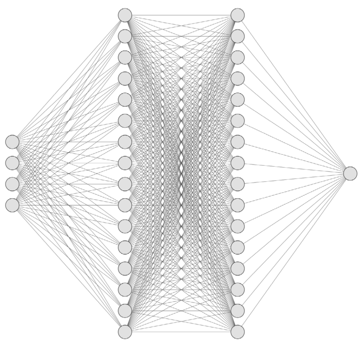


FIGURE 8. **Neural network architecture. From left to right are input layer (4 neurons), the first hidden layer (16 neurons), the second hidden layer (16 neurons), and the output layer (1 neuron).**

runs in each reader. In this stage, each reader collects the information from its neighbor readers to initialize a local graph. The neural network model is used to score each reader afterward. A reader will be activated if it has the highest score among its neighbor readers. Finally, the distributed MWISBAII algorithm proposed in [1] is used to arrange the rest of the readers. In this section, we describe the proposed approach in detail.

A. MACHINE LEARNING STAGE

This stage has three sub-phases: (1) data collection, (2) training the neural network, and (3) broadcasting the

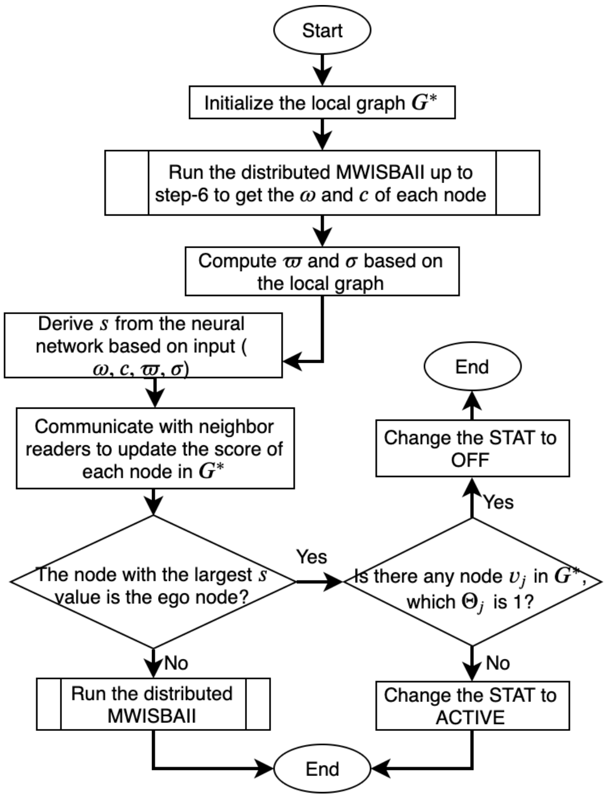

Start
Initialize the local graph G^*
Run the distributed MWISBAII up to step-6 to get the ω and c of each node
Compute ϖ and σ based on the local graph
Derive s from the neural network based on input ($\omega, c, \varpi, \sigma$)
Communicate with neighbor readers to update the score of each node in G^*
The node with the largest s value is the ego node?
Yes
No
Is there any node v_j in G^*, which Θ_j is 1?
Yes
No
Change the STAT to OFF
End
Change the STAT to ACTIVE
Run the distributed MWISBAII
End

FIGURE 9. **The flowchart of application stage.**

trained weights. We run simulations in the data collection phase to collect training data. We assume the simulated area is $100m \times 100m$ (m denotes unit meter), each reader has the identical specification (same α and β), and the readers are uniformly distributed, (see Figure 6). We assume that the maximum number of deployed tags is $\Psi = 100\psi$, where ψ is a positive integer value. We run ten random simulations for each number of tags in $\{100\rho \mid 1 \leq \rho \leq \psi, \rho \in \mathbb{N}\}$. At the start of each simulation, we record the following information of each reader (i denotes the i^{th} reader): weight (ω_i), cost (c_i), average weight of neighbors (ϖ_i), and average cost of neighbors (σ_i). Weight ω_i denotes the number of tags within the i^{th} reader; cost c_i is computed through Eq. 1. Then, we run the MWISBAII to get the solution. The solution is recorded in Λ, where $\Lambda_i \in \{0, 1\}$ ($\Lambda_i = 0$ represents the i^{th} reader is not active; $\Lambda_i = 1$ represents the i^{th} reader is active;). Each sample is a quintuple: $(\omega_i, c_i, \varpi_i, \sigma_i, \Lambda_i)$, where $(\omega_i, c_i, \varpi_i, \sigma_i)$ is the input to the neural network, and Λ_i is the target output. If the number of positive samples ($\Lambda_i = 1$) and the number of negative samples ($\Lambda_i = 0$) are not equal, we randomly drop some samples from the majority group of samples to make the number of samples in both groups are equal.

The neural network architecture we use is a fully-connected feedforward network with two hidden layers. Each hidden layer has 16 neurons; the input layer has 4 neurons which match the 4 input values $(\omega_i, c_i, \varpi_i, \sigma_i)$; the output layer only has 1 neuron (see Figure 8). Because the neural network

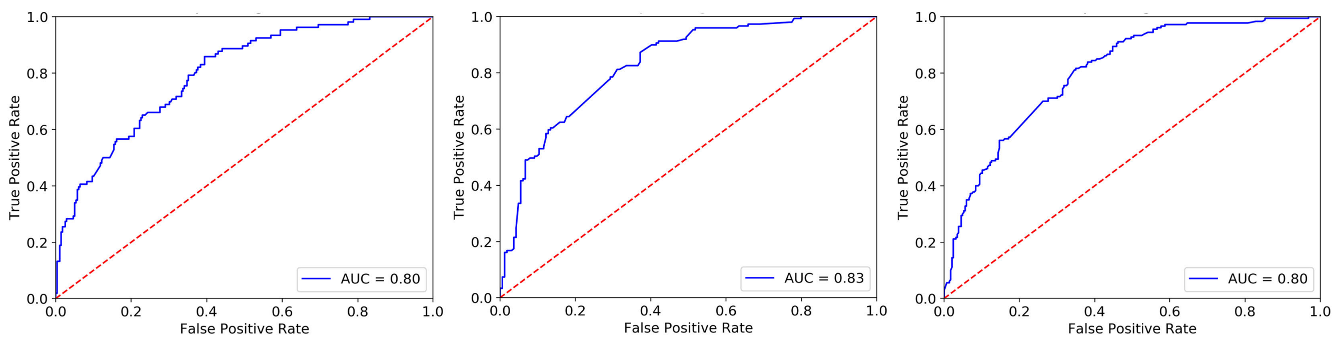


FIGURE 10. The testing ROC curves on setting 2 (left column), setting 5 (middle column), and setting 8 (right column).

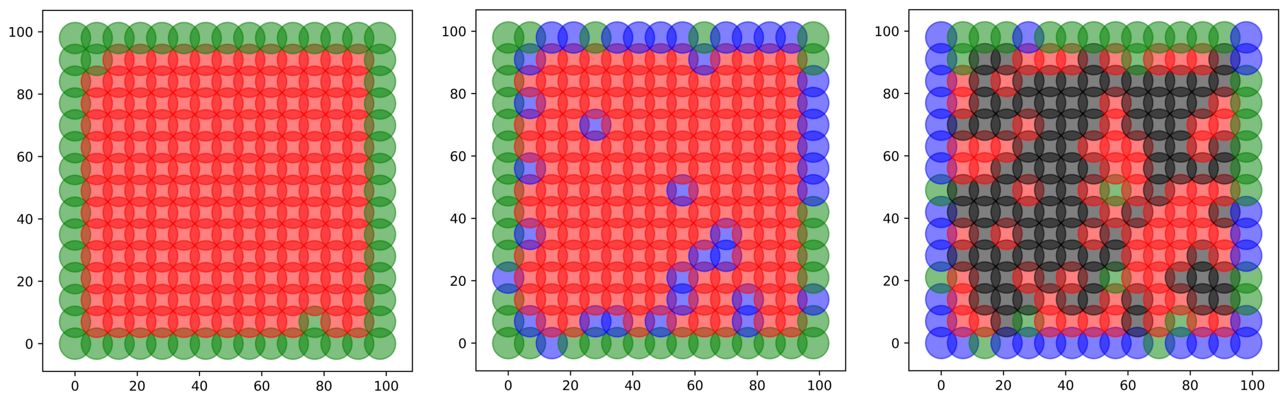


FIGURE 11. Readers clustered by k-means clustering algorithm. From left to right, the number of clusters (k) are two, three, and four.

weights can be represented as matrices, the total amount of weights of this neural network is $4 \times 16 + 16 \times 16 + 16 \times 1 = 336$ (not consider bias weights). We use 32-bit floating-point data type for neural network weights. Therefore the network weights take 10,752 bits (equivalent to 1,344 bytes) memory. We use rectified linear units (ReLU) [46] as the activation function of the hidden layers, and use the sigmoid activation function in the output layer. The neural network loss value is the mean squared error (MSE) of the neural network outputs and the target outputs. We use Adam optimizer [47] with a fixed learning rate (10^{-3}) to optimize the neural network (minimize the loss value). We train the neural network on the training samples 500 times (epochs), and the training batch size is 16 (each training step takes 16 samples). After training, we broadcast the trained neural network weights to each reader, and each reader assigns the received weights to its neural network model. Figure 7 depicts the flowchart of machine learning stage.

B. APPLICATION STAGE

This stage has two sub-phases. In the first sub-phase, we first let each reader establish its local ego-network graph G^*. The center node (ego node) of the local graph represents the reader itself, where the other nodes (external nodes) represent the neighbor readers that have a conflict with this reader (if we active this reader, all of its neighbor readers should not be activated; otherwise, if any of its neighbor readers is activated, this reader cannot be activated). Each node in the local graph should contain the following information (i denotes the i^{th} reader): weight (ω_i), cost (c_i), score (s_i), and status (Θ_i). The score s_i is initialized to 0 and will be generated by the neural network once this reader has all the input information ready. The status Θ_i is an indicator of the corresponding reader's status (STAT). $\Theta_i = -1$ represents the STAT of the i^{th} reader is either LOCK or OPEN; $\Theta_i = 0$ or 1 represent the STAT of the i^{th} reader is OFF or ACTIVE, respectively. If ω_i is either 0 or greater than α, the i^{th} reader should be deactivated. Here, we skip the detail of the communication between readers. We can run our previously proposed distributed algorithm up to step-6 (include step-6) to complete the node cost information in the local graph.

Based on the information (ω and c of each node) stored in G^*, each reader computes ϖ_i and σ_i. Then, each reader (reader i) inputs $(\omega_i, c_i, \varpi_i, \sigma_i)$ to its neural network, and use the neural network output as its score s_i. Once a reader updated the score of all the graph nodes (both the ego node and the external nodes), this reader will find the node that has the highest score in its local graph. If the node with the highest score is the ego node, this reader is activated

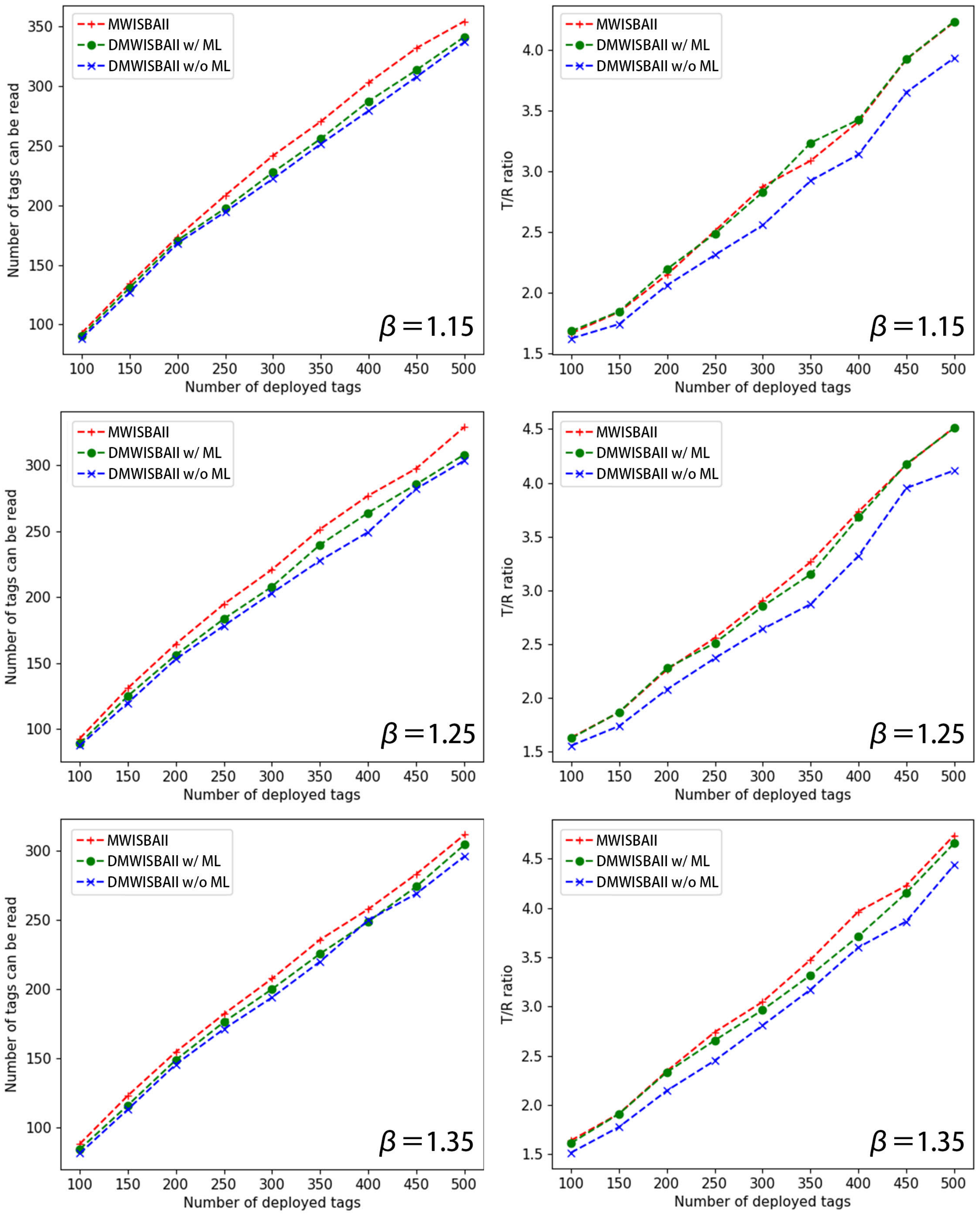


FIGURE 12. Performance evaluation results when reader interval is 6*m*.

(set Θ_i to 1). If any neighbor readers of a reader are activated before this reader, this reader should be deactivated (set Θ_i to 0). If there are more than one nodes (include the ego node) have the highest score, or the only node with the highest score is an external node, this reader will enter the second sub-phase which leverages the algorithm proposed in [1] to

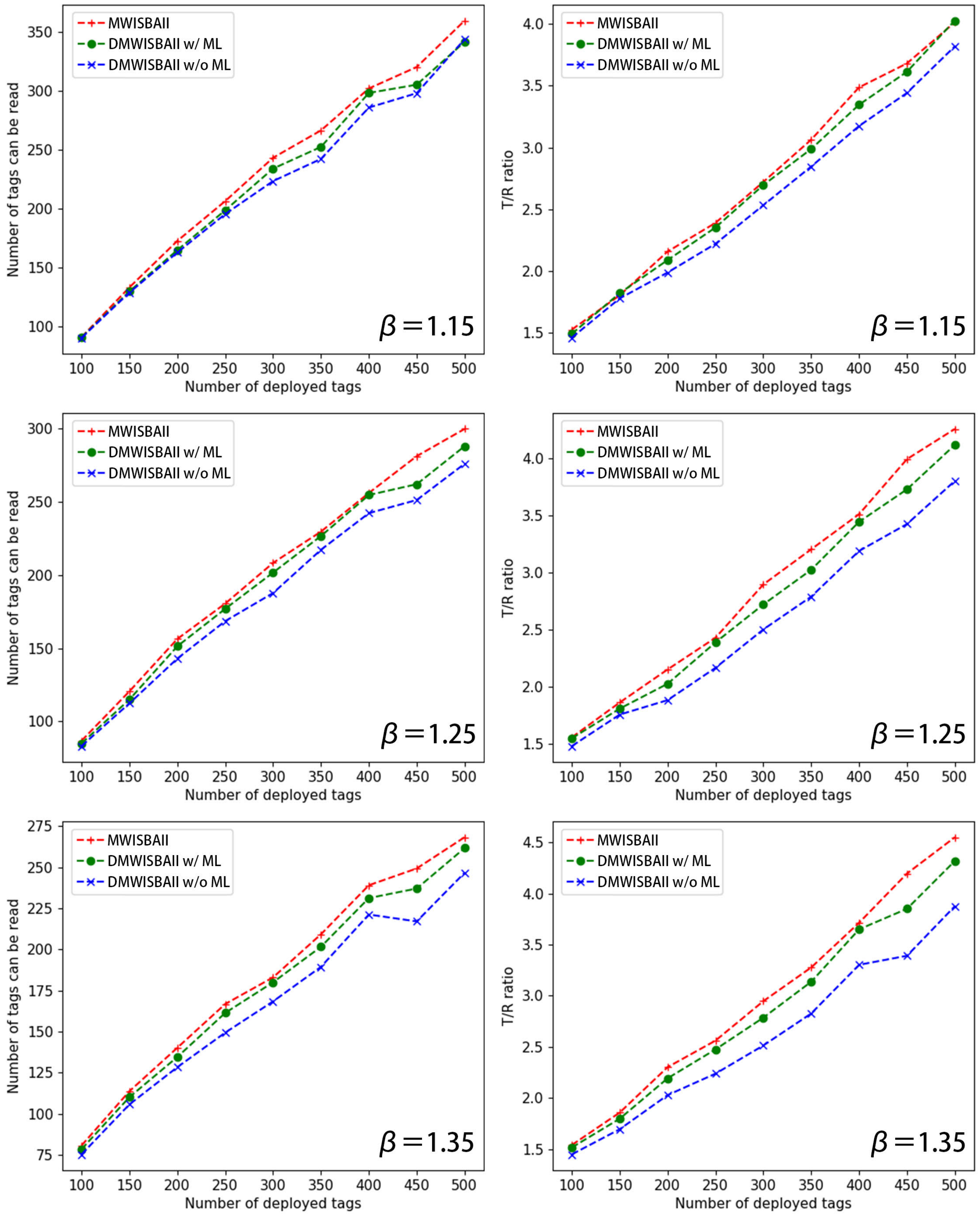


FIGURE 13. Performance evaluation results when reader interval is 7*m*.

let the rest of the readers whose STAT is neither OFF nor ACTIVE ($\Theta = -1$) make decision. Figure 9 depicts the flowchart of application stage.

The time complexity (each iteration) of the previously proposed distributed algorithm is $O(n)$, where n is the average number of vertices in G^*. Thereby, the time complexity

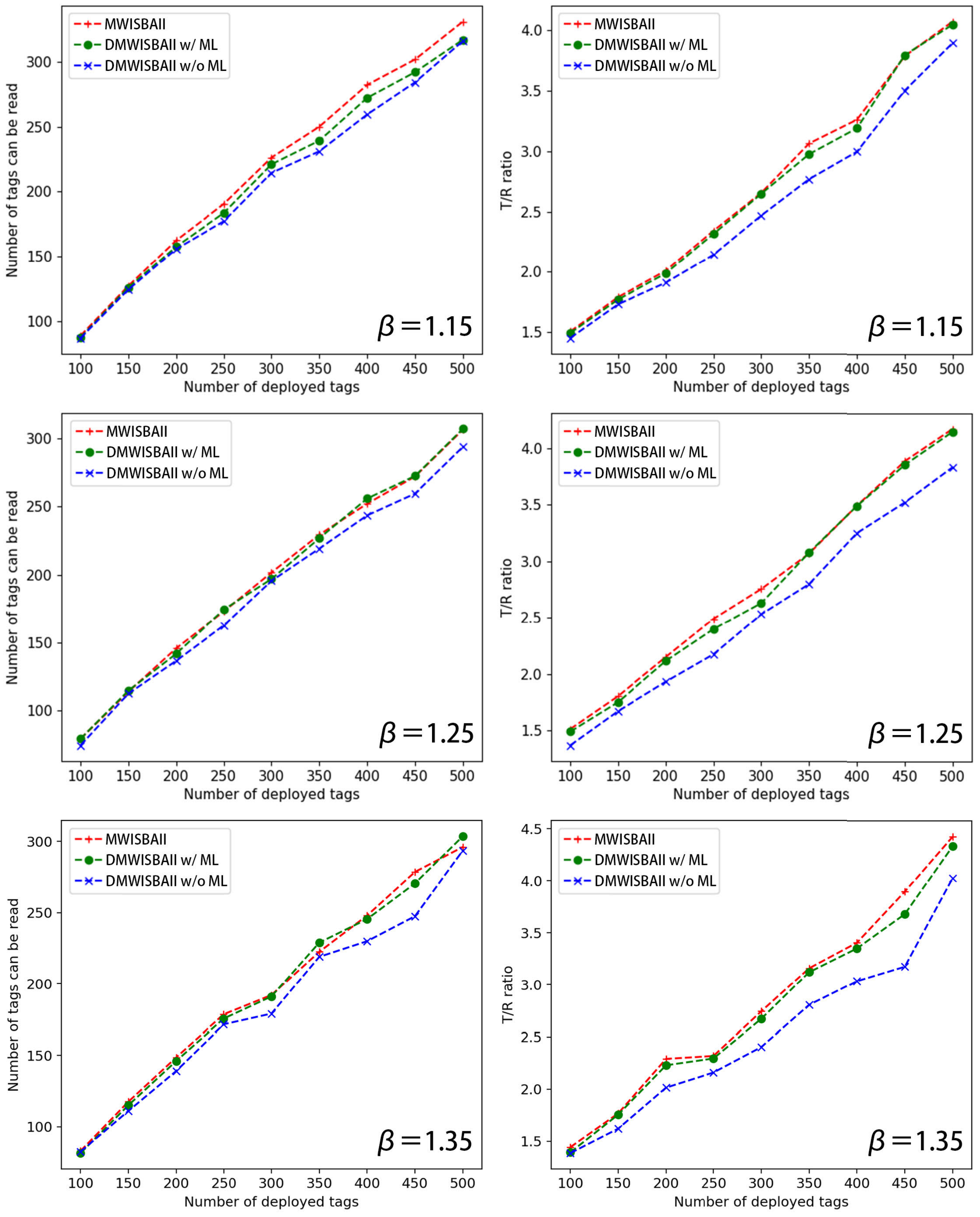


FIGURE 14. Performance evaluation results when reader interval is 8*m*.

for initializing G^* in the application stage is $O(n)$. The time complexity for computing either ϖ_i or σ_i is also $O(n)$, because there are $n-1$ external nodes in G^* on average. The neural network has a fixed number of parameters and operations. Thus, the time complexity for computing s_i is $O(1)$. In summary, the time complexity of the application

TABLE 1. Simulation settings for training.

Setting	Interval	β value	Number of tags
setting 1	$6m$	1.15	
setting 2	$6m$	1.25	
setting 3	$6m$	1.35	
setting 4	$7m$	1.15	
setting 5	$7m$	1.25	$\{100, 200, 300, 400, 500\}$
setting 6	$7m$	1.35	
setting 7	$8m$	1.15	
setting 8	$8m$	1.25	
setting 9	$8m$	1.35	

stage is equivalent to the previously proposed distribute algorithm.

VI. EXPERIMENTAL RESULTS

In the experiments, we simulated a square area ($100m \times 100m$) to deploy the readers and tags. We assume the interrogation range d of each reader is $5m$; the interference range d' of each reader is $d' = d\beta$, where β can be 1.15, 1.25, or 1.35. Each reader can read at most 10 tags ($\alpha = 10$). The positions of tags are randomly generated. The readers are uniformly assigned. The interval (horizontally and vertically) between two nearest readers can be $6m$, $7m$, or $8m$. Figure 6 depicts an example simulated area where the number of tags is 500, the reader interval is $7m$, and β is 1.25.

Table 1 depicts the settings of the simulations for training the neural network. To collect data with more variety, for each setting, we simulate ten times for each number of tags in $\{100, 200, 300, 400, 500\}$. Therefore, in the later experiments of evaluating the performance of the proposed algorithm, the number of deployed tags can be from 100 to 500. We randomly sampled 60% of the collected data points as training data, 20% data points as the validation data, and use the other 20% data points as testing data. Figure 10 shows the testing receiver operating characteristic (ROC) curves under setting 2, 5, and 8. In all of the experiments, the area under the curve (AUC) is greater than or equal to 0.8.

To get some visual insights into the simulation, under simulation setting 5, we apply k-means clustering algorithm on the readers ($k = 2, 3, 4$). Note well, each reader has the average ω_i, c_i, ϖ_i, and σ_i over simulations as its fingerprint. When $k = 2$, as shown in Figure 11 (the leftmost sub-figure), the readers at the edge are classified into one cluster. The reason is that in our simulations, the readers at the edge has a different number of neighbors than the other readers. When $k = 3$ or $k = 4$, the pattern of clusters is not obvious. Because theoretically, under our simulation setting, only the readers at the edge and the readers surrounded by the readers at the edge have an evident difference.

In the performance evaluation experiments, we use all the simulation settings in Table 1. For each setting, we ran the centralized MWISBAII, the proposed machine learning auxiliary approach (DMWISBAII w/ ML), and the distributed MWISBAII algorithm without machine learning assistance (DMWISBAII w/o ML) separately. The number of deployed tags is from set $\{100, 150, 200, 250, 300, 350, 400, 450, 500\}$, for each, we ran the simulation ten times and show the average results in Figure 12, Figure 13, and Figure 14. The evaluation metrics are the number of tags can be read by the RFID system and the T/R ratio. From the experiments, we can see that the proposed algorithm with machine learning assistance can get almost the same performance as the centralized MWISBAII. Besides, the proposed algorithm with machine learning assistance is always better than the one without machine learning assistance.

An interesting phenomenon in the performance evaluation results is that the number of tags that can be read by the system is less than the number of deployed tags. Also, with the number of deployed tags increases, the number of tags can be read increases slower. The explanation is that since each reader can read at most 10 tags ($\alpha = 10$), once the number of deployed tags within the interrogation range of a reader is greater than 10, this reader cannot be activated (this situation can be seen as a type of collision), the system might fail to read those tags.

VII. CONCLUSION AND FUTURE WORK

In our previous research, we found that due to the lack of global information, there is a gap in performance between the distributed MWISBAII and the centralized MWISBAII. The centralized MWISBAII algorithm will active the reader with the highest cost value in each iteration, and this highest cost value is a global highest cost. However, in [1], the distributed algorithm tends to miss some critical readers which should be activated. To solve this problem, we proposed a machine learning auxiliary approach to help each reader make a better decision. The machine learning model is a multi-layer neural network, which is trained on a central server only when the whole RFID system is set up for the first time. The training data is generated by running MWISBAII on multiple simulations. Therefore, the neural network is a supervised learning empirical model. The trained neural network model will be broadcast to each reader to predict the score of each reader. Although the training process is binary classification learning, we can interpret the output generated by the trained neural network model as some confidence level or score. Because the constraint of sigmoid activation function, the score is a continuous value between 0 and 1. A higher score means the neural network predicts that there is a higher probability to activate this reader to achieve better performance. The experimental results proved that the machine learning auxiliary approach helps the proposed distributed algorithm to make a more effective solution.

The experiments in this paper followed the assumption that RFID readers are uniformly distributed. Nevertheless, in many real-world scenarios, the distribution of RFID readers could be manifold. Therefore, merely training a single neural network model and broadcast the trained model to every RFID reader might not improve the performance significantly. In the future, we can employ unsupervised learning clustering algorithm such as k-means and mean-shift clustering to divide the RFID readers into groups based on the features $(\omega_i, c_i, \varpi_i, \sigma_i)$, and train a neural network model for each group of RFID readers individually. One challenge of this approach is that it is difficult to find an appropriate k value (the number of clusters). The other challenge is that, once the neural network model is trained for each cluster of readers, distributing the trained models to each cluster of readers increases the communication overhead.

Furthermore, extending the proposed approach to the dense RFID readers system where each RFID reader has more than one interrogation (and interference) ranges is also our future work.

REFERENCES

[1] P. Yan, S. Choudhury, and R. Wei, "A distributed graph-based dense RFID readers arrangement algorithm," in *Proc. ICC–IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, May 2019, pp. 1–6.

[2] A. Meddeb and A. Jaballah, "Algorithm for readers arrangement without collision in RFID networks," in *Proc. 18th Int. Conf. Parallel Distrib. Comput., Appl. Technol. (PDCAT)*, Dec. 2017, pp. 316–321.

[3] S. J. Udoka, "Automated data capture techniques: A prerequisite for effective integrated manufacturing systems," *Comput. Ind. Eng.*, vol. 21, nos. 1–4, pp. 217–221, Jan. 1991.

[4] L. Da Xu, W. He, and S. Li, "Internet of Things in industries: A survey," *IEEE Trans Ind. Informat.*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.

[5] D.-H. Shih, P.-L. Sun, D. C. Yen, and S.-M. Huang, "Taxonomy and survey of RFID anti-collision protocols," *Comput. Commun.*, vol. 29, no. 11, pp. 2150–2166, Jul. 2006.

[6] W. S. Johnson, Jr., and H. B. Reisgies, "Radio frequency identification (RFID) payment terminal with display-embedded RFID antenna," U.S. Patent 8 011 594, Sep. 6, 2011.

[7] J. Wang, Y. Guo, L. Guo, B. Zhang, and B. Wu, "Performance test of MPMD matching algorithm for geomagnetic and RFID combined underground positioning," *IEEE Access*, vol. 7, pp. 129789–129801, 2019.

[8] J. Wang, W. Wei, W. Wang, and R. Li, "RFID hybrid positioning method of phased array antenna based on neural network," *IEEE Access*, vol. 6, pp. 74953–74960, 2018.

[9] X. Feng, J. Zhang, J. Chen, G. Wang, L. Zhang, and R. Li, "Design of intelligent bus positioning based on Internet of Things for smart campus," *IEEE Access*, vol. 6, pp. 60005–60015, 2018.

[10] C. M. Roberts, "Radio frequency identification (RFID)," *Comput. Secur.*, vol. 25, no. 1, pp. 18–26, 2006.

[11] M. J. Brady, T. Cofino, H. K. Heinrich, G. W. Johnson, P. A. Moskowitz, and G. F. Walker, "Radio frequency identification tag," U.S. Patent 5 682 143, Oct. 28, 1997.

[12] M. J. Brady, D.-W. Duan, and V. S. Kodukula, "Radio frequency identification system," U.S. Patent 6 100 804, Aug. 8, 2000.

[13] F. Campioni, S. Choudhury, and F. Al-Turjman, "Readers scheduling for RFID networks in the IoT era," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2018, pp. 1–6.

[14] F. Campioni, S. Choudhury, and F. Al-Turjman, "Scheduling RFID networks in the IoT and smart health era," *J. Ambient Intell. Humanized Comput.*, vol. 10, no. 10, pp. 4043–4057, Jan. 2019.

[15] A. Munir, M. T. R. Laskar, M. S. Hossen, and S. Choudhury, "A localized fault tolerant load balancing algorithm for RFID systems," *J. Ambient Intell. Humanized Comput.*, vol. 10, no. 11, pp. 4305–4317, Oct. 2018.

[16] H. Chen, L. Liu, R. Che, K. Lin, X. Ai, and Y. Li, "On using sampling Bloom filter for unknown tag identification in large-scale RFID systems," *IEEE Access*, vol. 6, pp. 57095–57104, 2018.

[17] D.-Y. Kim, B.-J. Jang, H.-G. Yoon, J.-S. Park, and J.-G. Yook, "Effects of reader interference on the RFID interrogation range," in *Proc. Eur. Microw. Conf.*, 2007, pp. 728–731.

[18] D. W. Engels and S. E. Sarma, "The reader collision problem," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, vol. 3, Oct. 2002, Art. no. 6 pp. vol.3.

[19] Y. Fu, X. Wang, E. Wang, and Z. Qian, "A bit arbitration tree anti-collision protocol in radio frequency identification systems," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 11, Nov. 2017, Art. no. 155014771774157.

[20] Y. Kang, M. Kim, and H. Lee, "A hierarchical structure based reader anti-collision protocol for dense RFID reader networks," in *Proc. 13th Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2011, pp. 164–167.

[21] B. Zhi, W. Sainan, and H. Yigang, "A novel anti-collision algorithm in RFID for Internet of Things," *IEEE Access*, vol. 6, pp. 45860–45874, 2018.

[22] G. Zhang, S. Tao, W. Xiao, Q. Cai, W. Gao, J. Jia, and J. Wen, "A fast and universal RFID tag anti-collision algorithm for the Internet of Things," *IEEE Access*, vol. 7, pp. 92365–92377, 2019.

[23] H. Saadi, R. Touhami, and M. C. E. Yagoub, "TDMA-SDMA-based RFID algorithm for fast detection and efficient collision avoidance," *Int. J. Commun. Syst.*, vol. 31, no. 1, p. e3392, Aug. 2017.

[24] H. Rezaie and M. Golsorkhtabaramiri, "A fair reader collision avoidance protocol for RFID dense reader environments," *Wireless Netw.*, vol. 24, no. 6, pp. 1953–1964, Jan. 2017.

[25] Z. J. Tang, Y. Guo, and Q. Liu, "Research on an improved fusion RFID collision avoidance algorithm," *J. Commun. Technol., Electron. Comput. Sci.*, vol. 22, pp. 6–19, Feb. 2019.

[26] A. A. Mbacke, N. Mitton, and H. Rivano, "A survey of RFID readers anticollision protocols," *IEEE J. Radio Freq. Identificat.*, vol. 2, no. 1, pp. 38–48, Mar. 2018.

[27] J. Ho, D. W. Engels, and S. E. Sarma, "HiQ: A hierarchical Q-learning algorithm to solve the reader collision problem," in *Proc. Int. Symp. Appl. Internet Workshops (SAINTW)*, Jan. 2006, pp. 4 and 91.

[28] J. Yu and W. Lee, "GENTLE: Reducing reader collision in mobile RFID networks," in *Proc. 4th Int. Conf. Mobile Ad-Hoc Sensor Netw.*, Dec. 2008, pp. 280–287.

[29] J. Su, Z. Sheng, V. C. M. Leung, and Y. Chen, "Energy efficient tag identification algorithms for RFID: Survey, motivation and new design," *IEEE Wireless Commun.*, vol. 26, no. 3, pp. 118–124, Jun. 2019.

[30] H. Su, Y. Li, Z. Siyi, H. Yan, and H. Chen, "Multichannel reader collision avoidance mechanism in RFID-sensor integrated networks," *J. Comput.*, vol. 29, no. 5, pp. 260–271, 2018.

[31] J. Su, Z. Sheng, L. Xie, G. Li, and A. X. Liu, "Fast splitting-based tag identification algorithm for anti-collision in UHF RFID system," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2527–2538, Mar. 2019.

[32] L. Yang, J. Han, Y. Qi, C. Wang, T. Gu, and Y. Liu, "Season: Shelving interference and joint identification in large-scale RFID systems," in *Proc. Proc. IEEE INFOCOM*, Apr. 2011, pp. 3092–3100.

[33] Z. Zhou, H. Gupta, S. R. Das, and X. Zhu, "Slotted scheduled tag access in multi-reader RFID systems," in *Proc. IEEE Int. Conf. Netw. Protocols*, Oct. 2007, pp. 61–70.

[34] W. Zhu, Y. Hong, V. Raychoudhury, R. Zhao, and D. Wang, "Adaptive distributed reader activation approach for large-scale RFID systems," in *Proc. IEEE 12th Int. Conf. Mobile Ad Hoc Sensor Syst.*, Oct. 2015, pp. 82–90.

[35] A. Fahim and T. ElBatt, "Multi-reader RFID tag identification using bit tracking (MRTI-BT)," in *Proc. IEEE Int. Conf. RFID (RFID)*, May 2016, pp. 1–7.

[36] R. M. Ferdous, A. W. Reza, and M. F. Siddiqui, "Renewable energy harvesting for wireless sensors using passive RFID tag technology: A review," *Renew. Sustain. Energy Rev.*, vol. 58, pp. 1114–1128, May 2016.

[37] J. Zhang, G. Tian, A. Marindra, A. Sunny, and A. Zhao, "A review of passive RFID tag antenna-based sensors and systems for structural health monitoring applications," *Sensors*, vol. 17, no. 2, p. 265, Jan. 2017.

[38] B.-H. Liu, N.-T. Nguyen, V.-T. Pham, and Y.-H. Yeh, "A Maximum-Weight-Independent-Set-Based algorithm for reader-coverage collision avoidance arrangement in RFID networks," *IEEE Sensors J.*, vol. 16, no. 5, pp. 1342–1350, Mar. 2016.

[39] M. R. Garey, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. San Francisco, CA, USA: Freeman, 1997.

[40] P. Erdos, "On the graph theorem of Turán," *Mat. Lapok*, vol. 21, nos. 249–251, p. 10, 1970.

[41] J. R. Griggs, "Lower bounds on the independence number in terms of the degrees," *J. Combinat. Theory B*, vol. 34, no. 1, pp. 22–39, Feb. 1983.

[42] S. Sakai, M. Togasaki, and K. Yamazaki, "A note on greedy algorithms for the maximum weighted independent set problem," *Discrete Appl. Math.*, vol. 126, nos. 2–3, pp. 313–322, Mar. 2003.

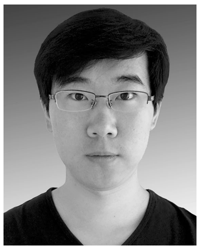
[43] P. Du and Y. Zhang, "A new distributed approximation algorithm for the maximum weight independent set problem," *Math. Problems Eng.*, vol. 2016, 2016, Art. no. 9790629, doi: 10.1155/2016/9790629.

[44] Z. Wang, N. Ye, R. Malekian, F. Xiao, and R. Wang, "TrackT: Accurate tracking of RFID tags with mm-level accuracy using first-order Taylor series approximation," *Ad Hoc Netw.*, vol. 53, pp. 132–144, Dec. 2016.

[45] E. G. Coffman, M. Elphick, and A. Shoshani, "System deadlocks," *ACM Comput. Surv. (CSUR)*, vol. 3, no. 2, pp. 67–78, Jun. 1971.

[46] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980



PEIZHI YAN received the B.Sc. degree in computer science from Algoma University, Sault Ste. Marie, ON, Canada, in 2018, and the B.Eng. degree in computer science from the University of Jinan, Jinan, Shandong, China, in 2019. He has been a Graduate Student with Lakehead University, Thunder Bay, ON, Canada, since 2018. His research interests are in machine learning, artificial neural networks, computer vision, and AI-aided network optimization. He was a recipient of the Vector Institute 2018–2019 Vector Scholarship in Artificial Intelligence. He has served as a Reviewer for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY.

SALIMUR CHOUDHURY received the Ph.D. degree in computing from Queen's University, Kingston, ON, Canada, in 2012. He is currently an Assistant Professor with the Department of Computer Science, Lakehead University, Thunder Bay, ON, Canada. He is also the Director of the Lakehead Optimization Research Group (LORG). His research interests include designing algorithms for wireless communications, optimization, cellular automata, and approximation algorithms. He was the Technical Program Chair of the SGIoT 2017, SGIoT 2018, and iThings 2018 Conferences. He is also an Editor of *Parallel Processing Letters*.

RUIZHONG WEI received the B.Sc. degree from Suzhou University, China, in 1982, and the Ph.D. degree from the University of Nebraska–Lincoln, in 1998. He held academic positions at Suzhou University, from 1982 to 1995. He worked at Mount Saint Vincent University, the University of Nebraska–Lincoln, and the University of Waterloo. He joined Lakehead University, Thunder Bay, ON, Canada, in 2000. He is currently a Professor in computer science. His research interests are in combinatorial mathematics, computer security, and information theory. He is also a Fellow of the Institute of Combinatorics and its Application.