Journal of Ambient Intelligence and Humanized Computing (2019) 10:4305–4317
https://doi.org/10.1007/s12652-018-1114-7

ORIGINAL RESEARCH

A localized fault tolerant load balancing algorithm for RFID systems

Ahnaf Munir[1] · Md. Tahmid Rahman Laskar[2] · Md Sakhawat Hossen[1] · Salimur Choudhury[3]

Received: 24 September 2018 / Accepted: 22 October 2018 / Published online: 31 October 2018

Abstract
Radio frequency identification (RFID) is a unique scientific invention that comprises individually recognizable, low-cost tags and readers where the readers monitor the tags using frequencies from the radio spectrum. Uniform distribution of the tags for gaining a balanced load of the readers is a significant concern to ensure successful collection of data from all of the tags of an RFID system with multiple readers. Moreover, some of the readers in an RFID network may become defective during operation and stop working. As a result, information would not be collected from those tags which were associated with the defective readers and the network would operate with partial information. We target to maintain a balance among the load of the readers by placing the tags as evenly as possible to address the fast tag reading problem. We convert the addressed issue as a load balancing problem and introduce a cellular automaton inspired localized algorithm as a solution to this problem. Our proposed algorithm utilizes the local information of the readers to relocate tags from a heavily loaded reader to a lightly loaded reader. We develop our proposed algorithm as a fault tolerant one so that all of the tags in the network are always under surveillance even if some of the readers become defective. Numerical analysis and comparison results suggest that the proposed localized load balancing algorithm outperforms the existing localized solution and gives a competitive result compared to the centralized algorithm. Finally, we implement our proposed algorithm in the parallel programming platform Compute Unified Device Architecture that greatly improves the runtime of the proposed algorithm.

Keywords RFID · Cellular automaton · Fault tolerant · Load balancing · Reading time · Readers · Tags · Fairness index · CUDA

1 Introduction

Radio frequency identification (RFID) is one of the most widespread wireless technologies adopted in different different aspects of our daily life. Some of the notable applications of RFID technology include supply chain management, tracking and identification, health monitoring, environmental data logging, user access control, etc. The RFID technology has become one of the fundamental elements for 'Internet of Things' (Dong et al. 2011; Dominikus 2011) because of its ease of scaling and low cost implementation. The major building blocks of an RFID system includes the readers and the tags where the number of tags is significantly higher than that of readers. A reader is a transceiver which is capable of interrogating the tags and access the information stored in the tags. However, a reader is capable of accessing the tags only within its proximity and the proximity of a reader is defined according to its interrogation range which normally varies from reader to reader. On the other hand, a tag is a transponder that responds to the query of a reader. A tag is basically a passive entity with limited memory and energy to store a few but significant information. Compared to other states of the art identification systems like biometrics, bar-code systems, optical character recognition (OCR) system, etc., one of the

✉ Salimur Choudhury
salimur.choudhury@lakeheadu.ca

Ahnaf Munir
ahnaf@iut-dhaka.edu

Md. Tahmid Rahman Laskar
tahmedge@cse.yorku.ca

Md Sakhawat Hossen
sakhawat@iut-dhaka.edu

[1] Department of Computer Science and Engineering, Islamic University of Technology, Dhaka, Bangladesh

[2] Department of Electrical Engineering and Computer Science, York University, Toronto, Canada

[3] Department of Computer Science, Lakehead University, Thunder Bay, Ontario, Canada

main advantages of the RFID technology is that it does not ask for a line-of-sight communication between a reader and a tag for its proper functioning.

The main challenge of an RFID system with multiple readers is to eliminate the collision among the tags at the time of interrogation which primarily happens because of the ratio between the number of readers and the number of tags present in the system. In an RFID system, a reader handles a significant number of tags within its proximity or interrogation range which may result in a significant number of collisions among the tags at the time of responding to a query by a reader simultaneously. Eventually, the responding tags cancel out each other's responses which contributes a significant delay to the interrogation time of a tag by a reader. So, a balanced load among the readers is a necessity to address this problem of fast tag reading time. A number of anti-collision algorithms (Capetanakis 1979; Jihoon and Wonjun 2006; Lawrence 1975; Vinod and Lixin 2007; Yoon and Vaidya 2010) can be found in literature those deal with the load balancing of the readers. However, most of them fail to provide a fast tag interrogation time because they do not consider the reader to reader communication at the time of tag interrogation. A tag in the proximity of multiple readers is interrogated by all of them which results in redundant information gathering. This redundancy can be avoided if the readers communicate among themselves prior to interrogating the tags in an RFID system with multiple readers and a great improvement in tag reading time can be achieved. In Xie et al. (2012), a solution to the addressed problem is provided which considers the reader to reader communication which ensures that a tag is accessed by a lone reader. However, they employ a distributed load balancing algorithm which introduces an overhead to the system due to the distributed processing and message passing.

In this research article, our target is to balance the load of the readers by placing the tags as evenly as possible to address the fast tag reading problem. We express the addressed research issue as a load balancing problem and develop a localized algorithm as a solution to the load balancing problem. Our proposed algorithm is based on cellular automaton (CA) (Garzon 1995) that utilizes the local information of the reader to achieve the goal. The advantage of using CA based algorithm is that, it needs much less memory and computation power (since it computes solutions based on the state of itself and its neighboring cells). As of our concern, this is the first ever strictly localized solution for the addressed research problem which is based on CA.

Our proposed load balancing algorithm ensures that each of the tags is read by a single reader. Therefore, if any of the readers get damaged, the tags associated with that reader would be ignored by the network. Such tags whose information is not being read are called orphan tags. If an orphan tag has neighboring readers that are not defective then it can be redistributed to any one of them to help maintain the proper operation of the RFID network. We also address the problem that occurs when multiple readers of the network stop working and leave the tags in their proximity as an orphan. We incorporate a fault tolerant procedure with our proposed solution that can redistribute the orphan tags to the nearby functioning readers. We compare the simulation results of our proposed load balancing algorithm with other states of the art solution. The comparison results suggest that the proposed algorithm is superior to the existing algorithm (Xie et al. 2012) in terms of tags per reader (tag load). We compare the performance of our fault-tolerant procedure using two different metrics and observe that both of the metrics provide similar performance with some trade-offs between them. The initial version (without fault tolerance) of our proposed load balancing algorithm can be found in Munir et al. (2016).

The remaining part of this article is arranged in the following manner. A brief introduction of the cellular automaton is presented in Sect. 2. Section 3 presents some of the notable works addressing the same research problem. We present the system model along with the problem formulation in Sect. 4. We elaborate our proposed load balancing algorithm and fault tolerant algorithm in Sects. 5 and 6 respectively. Simulation results and the numerical analysis are discussed in Sect. 7 and finally, we conclude the paper in Sect. 8.

2 Cellular automata

A CA is a discrete model which is a popular way of modeling the various types of physical systems. It consists of a group of cells that form a grid. The state of the cells can change over discrete time steps based on certain rules which take the neighboring cells into consideration. In a two-dimensional environment, the CA is defined as the quadruple, $A = (C, S, \delta, N)$. Here, C denotes a grid. The grid is made of elements called cells. The cells of CA can exist in a finite number of states. The set of these states is denoted by S. δ is the set of transition rules that allows the cells to change from one state to another. The neighborhood of a cell is represented by N. At any time step t, each of the cells of C is assigned a state from S. At time step $t + 1$, the state of a certain cell $c \in C$ may or may not change based on the transition function δ. The functions consider the state of c and its neighbors at time step t to make this decision. The neighborhood of a cell can vary depending on the CA system. In a square grid, a cell has a total of eight adjacent cells. If a CA system uses the Moore

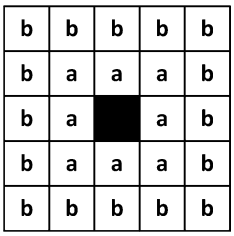

Fig. 1 Radius 1 and radius 2 Moore neighbors

neighborhood, then it considers all of the eight cells as a neighbor.

Figure 1 illustrates the Moore neighborhood for radii 1 and 2. In the figure, the radius 1 neighbors are marked as 'a' and the radius 2 neighbors are marked as 'b'. Elaborate descriptions regarding CAs can be found in Garzon (1995).

3 Related work

The application of collision avoidance protocols is one of the primary methods of improving the performance of RFID systems. A remarkable number of such protocols are present in the literature which can be divided into two major categories: (1) tree-based and (2) ALOHA based. Two tree based collision avoidance protocols are introduced in Capetanakis (1979) and Jihoon and Wonjun (2006) those use deterministic models based on either a binary or query tree. In these papers, all of the tags present in an RFID system are distributed into smaller subgroups preferably with a lone tag. Every subgroup of the model is associated with an ID which can be used to uniquely identify the tags. A subgroup having multiple tags acts as an inner node of the tree whereas a subgroup with a single tag acts as a leaf node of the tree. The main disadvantage of these type of protocols is the significant identification delay. In Bhatia and Sood (2018) a threshold based decision tree model is used to classify activities of defense personnel in IoT-based systems. On the other hand, the ALOHA based collision avoidance protocols introduced in Lawrence (1975) and Vinod and Lixin (2007) are probabilistic models where the tags randomly choose time slots within the query frame sent by the reader. Collisions cannot be avoided completely in ALOHA based protocols. These protocols are also susceptible to 'tag starvation' where a certain reader may remain unidentified for a prolonged amount of time. Estimating the number of tags present in an RFID system is one of the vital problems encountered by many anti-collision protocols. Chen et al. (2008) discuss a novel method of approximating the number of tags present in the proximity of a reader. An algorithm based on bitmap and hash function is used in this paper to develop a replica-intensive protocol that estimates the number of tags. This algorithm is suitable in cases where multiple readers are present in the network. This protocol eliminates the need for the unique identification of the tags and anti-collision protocols. An RFID authentication scheme based on hash functions and bitwise-XOR functions is introduced in Wu et al. (2018) for e-healthcare applications.

The identification of tags in an RFID system greatly depends on the transmission control strategy which controls the access to the broadcasting channel. In Floerkemeier (2006), one such scheme is presented which is based on framed ALOHA and adopts earlier works on Bayesian broadcast strategies. A couple of unified approximation algorithms are proposed in Murali and Thyaga (2006) which have complementary properties. The collision-based estimator proposed in this paper provides a high level of accuracy with the help of three estimators. On the other hand, the level of accuracy for the probabilistic estimators is constant as the running time is not dependent on the estimated tag set size. The challenges faced in an RFID system is increased further if the load of each of the readers in the system is not balanced. Such imbalanced are commonly found in systems like warehouses, modern airports, supermarkets etc. This common problem is not properly addressed in the literature stated above. A probabilistic distributed algorithm is suggested by Qunfeng et al. (2007) that utilizes the min-max tag count assignment to ensure the balance of tag loads in this type of topology. Dhas et al. (2010) try to maintain both the load balancing and the elimination of redundant readers by storing the load information of the reader inside every tag that is read by that reader. The drawback of this solution is that it cannot be implemented in large-scale RFID systems consisting of thousands of passive tags.

Jang and Lee (2008) consider a load balancing agent based on a fuzzy logic control. For any edge M/W, the fuzzy logic control is used to predict its volume and evaluate its workload. The agent is responsible for migrating the workload from overload edges to under-loaded edges M/Ws. In large-scale RFID systems, it is common for multiple readers to simultaneously read the same tags. In these scenarios, the elimination of the redundant readers may lead to faster tag reading times and balanced load for the active readers. Hsu et al. (2011) define fuzzy membership functions and fuzzy rules to determine different degrees of danger in RFID-based systems. A load balancing mechanism is introduced in Zhang et al. (2018) to increase the quality and the source utility of a WLAN. Multiple centralized and localized algorithms are proposed in Ali et al. (2011a, b) and Irfan et al. (2011) to eliminate the redundant readers in a RFID system. In Rashid et al. (2016, 2018), cellular automaton based localized algorithm has also been used to address this redundant reader elimination problem. However, none of the aforementioned algorithms guarantee a balanced load of the readers. Chiu and Jain (1989)

introduce a metric known as the fairness index that compares loads of the readers. A distributed load balancing algorithm was developed in Xie et al. (2012) for addressing the fast reading problem. This algorithm assumes that at each time step, each reader determines its neighboring reader. Next, each of the readers tries to find a neighbor that has a smaller load. If such a neighbor is found, then the reader transfers the control of a specific number of its own tags ($0 \leq \alpha \leq 1$) to its neighbor. The process is continued by each of the readers until either the fairness index rises to a fixed threshold or the fairness index is not affected by a further transfer of tags. The idea of a maximum flow algorithm is adapted in Dong et al. (2011) to develop a centralized optimal load balancing algorithm. In Meddeb and Jaballah (2017), an improved maximum weight independent set based algorithm (MWISBA) is proposed to address the reader coverage collision avoidance arrangement (RCCAA) problem where the interference range of a reader is considered along with the interrogation range in a multiple reader RFID systems. In Wang and Liu (2017), a cost-efficient method of reader deployment is proposed where the reader are adjustable to different communication ranges by tuning their transmission power. Campioni et al. (2018) propose one centralized and several localized algorithms for reader scheduling in an RFID system. An extended version of their algorithms are under the review process of the Journal of Ambient Intelligence and Humanized Computing (AIHC). Their algorithm is also inspired by CA and they consider the readers with limited residual energy and having dense deployment over the environment. In their proposed model, the readers communicate (direct or indirect via the tags) among themselves to maintain the sleep/awake scheduling with an aim to optimize network lifetime and energy usage. We need to make it clear that, though our proposed algorithm is also CA-based, we are addressing the fast tag reading time problem (completely different than their problem) by balancing the tag loads of the reader. Hence, our algorithm is also completely different. Moreover, we emphasize the fault-tolerant behavior of our proposed algorithm and implemented our algorithm in CUDA.

Most of the works done on fault-tolerant RFID networks are based on reader localization. Localization in RFID systems can be divided into two main categories: (1) tag localization and (2) reader localization (Sanpechuda and Kovavisaruch 2008). In tag localization, each object to be located is attached with an RFID tag while the readers are scattered in the environment. A server is responsible for gathering data from the readers, executing a localization algorithm, and notifying the localization result to the object. In reader localization, each object to be located is equipped with an RFID reader and a set of RFID tags are distributed in the area of interest.

Bulusu et al. (2000) apply radio communication to ensure outdoor localization in a low-cost method where the nodes of interest are distributed in the environment. The nodes are capable of transmitting radio signal as a beacon frame and when the beacon frame is received by an object of reference, it can locate itself as the centroid of the proximate reference node. In He et al. (2003), each of the target objects tries to form a triangle of three reference node by interacting with its neighboring target objects. The target object is capable of narrowing down the region where it is possibly located by following this method. Bouet and Pujolle (2008) use virtual reference nodes to determine the location where the object is located. A non-linear programming scheme is used in Wang et al. (2007) to located the readers. Here, the geometric knowledge of the reader's identification area is used. A real-time localization system is developed in Bilodeau et al. (2018) for enhanced tracking in smart homes. A process called ATI is proposed in Zhu et al. (2014) which can provide a quality index to determine the localization results. ATI is capable of tolerating regional permanent fault and works in both 2D and 3D environment.

4 System model and problem statement

We are considering a large-scale RFID system that consists of numerous tags and a number of readers. Figure 2 illustrates an instance of such systems. The readers are numbered as 1–6 in the figure and the tags are represented by small black dots. Our system consists of only passive tag that does not have any additional power source or processing ability. When a reader sends a signal to a tag, it uses the power from that signal to send back the contents of its memory to the reader. The comparatively low price

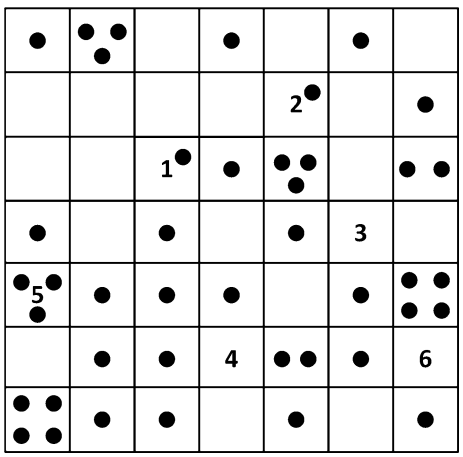

Fig. 2 Model of an RFID system with multiple reader in a single cell of a grid

of such passives tags makes them common in different applications.

Our environment is considered to be a two-dimensional grid where each cell of the grid can contain only one reader, but multiple tags. Each reader has a fixed interrogation range. A reader can only read the tags that are located within its interrogation range. It can also elect to not monitoring any of the tags placed in its range. The reader placed in the grid are capable of communicating and cooperating with each other to develop a reader network. Each of the tags in the grid can only be associated with a single reader. At the initial stage, the tags that fall under the interrogation range of multiple readers are randomly associated with any of the readers. Due to the randomness of the assignment process, the balance distribution of tags among the reader cannot be guaranteed. We assume that the tag reading time is constant. So, the time required to read from all the tags in the network is directly dependent on the maximum number of tags assigned to a single reader. Therefore, the feasible approach to reduce the tag reading time in the multi-reader RFID system is to minimize the maximum number of tags assigned to any of the readers. We can formally define this as:

$$minimize, maximum(w_1, w_2, \ldots, w_n)$$
$$where, w_i = |S_i|$$

where S_i is set of tags assigned to reader r_i and w_i is the cardinality of the set S_i. If any of the readers are assigned with multiple tags, it will try to shift a portion of its tags to the neighboring readers to ensure that the load of the tags is balanced among all the readers.

5 Proposed algorithm

Each of the readers in the RFID network is capable of executing both the load balancing algorithm and also the fault tolerant procedure. The fault tolerant procedure is only executed if a reader finds an orphan tag inside its radius. The reader executes the load balancing algorithm for all other tags. In this section, we introduce our load balancing algorithm and the fault tolerant procedure is discussed in Sect. 6.

5.1 Algorithm metric

The fairness index introduced in Chiu and Jain (1989) is used in this paper to measure the fairness of tag distribution among the readers. This metric can be used to determine the fairness either in a specific region of the RFID system or in the entire environment. Let, $\mathbf{w} = (w_1, w_2, \ldots, w_n)$ be the vector representing the load of the readers present in the RFID system. Now, we can express the fairness index of the system by the following equation:

$$fairness\_index = \frac{(\sum_{i=1}^{n} w_i)^2}{n \sum_{i=1}^{n} w_i^2}.$$

The fairness index can hold any value between 0 and 1 where a higher value indicates better fairness (a value of 1 indicates complete fairness). Here, all the readers in the system would be assigned an equal number of tags. A value of $\frac{1}{n}$ would represent that the system is completely unfair with all the tags assigned to a single reader. In an ideal scenario, our goal would be to improve the fairness index of the entire RFID system. However, this goal is difficult to achieve with a localized solution where a reader is allowed to get connected with its neighboring readers only. Through the exchange of the load information with its neighbors, a reader can calculate the fairness index of its local area only. It is to be noted that an optimal solution of load minimization of a reader does not guarantee the optimal fairness index of a system.

5.2 Proposed CA based load balancing algorithm

Our proposed algorithm is modeled based on cellular automaton and we name our algorithm as CA based load balancing algorithm (CALBA). Any cell of the model can contain both readers and tags. A cell can contain one reader at most but may have multiple tags. The algorithm is described below:

5.2.1 State representation

A single tag can be in the interrogation range of a maximum of nine readers. The state of the tag can be used to determine which reader has been assigned a tag to and also its direction in respect to the tag. This is shown in Fig. 3. For example, if the state of a tag is 'C', then the tag is under the surveillance of a reader residing in the same cell as the tag. A reader state defines the number of tags under its surveillance at any given time.

5.2.2 Neighborhood

Our model assumes that all of the readers present in the system have a fixed communication range of radius 2. Readers within each others communication range can exchange load information among them. Each reader also has an interrogation range. A reader can only read and write contents on the tags that

Fig. 3 Possible positions of a reader to interrogate a tag

NW	N	NE
W	C	E
SW	S	SE

are within this range. We consider two different interrogation ranges for the readers (radius 1 and 2) in our model. This is discussed elaborately in our simulation results in Sect. 7.

5.2.3 Transition rules

At the initial stage, we deploy a random number of tags and readers in the environment and the tags are randomly assigned to the readers. The state of both the reader and tags are then updated based on this initial distribution. From the next time step, each reader adheres to the algorithm shown in 1. The steps of the algorithm are discussed below:

1. (lines 2–13) The reader checks the state of each of the tags that are in its proximity. If any of the tags are orphan, it executes the fault tolerant procedure which is described in Sect. 6. This is done by calling the *faultTolerantProcedure*(T_k, *metric*) function where the parameter T_k is an orphan neighboring tag of the reader and *metric* determines the type of metric that is going to be used in the fault tolerant procedure. The details of the fault tolerant procedure are described in the next section. If the reader does not find any orphan tags then it calculates its own load and a load of its nearby readers. Moreover, The reader calculates the fairness index of its own neighborhood.
2. (lines 14–20) The current reader now examines each of its neighboring readers and identifies which of the tags assigned to the current reader is also within the interrogation range of its neighbors. Next, the reader determines whether transferring its own tag to a specific neighbor can improve the local fairness index. If such improvement is possible, the reader updates the tag so that it is transferred to the neighboring reader. For example, suppose R_j is the neighbor of the current reader R_i which has a tag T_l assigned to it. R_i and T_l are located in the same grid while R_j is located to the 'NW' position with respect to T_l. So the current state of T_l is 'C'. Now, if R_i determines that the local fairness index can be improved by transferring T_l to R_j, then it updates the state of T_l from 'C' to 'NW' in order to reflect the change of the tag assignment.
3. Steps 1 and 2 are repeated by the readers in each time step. During a time step, a reader can only transfer its own load to other readers. However, the load of the reader can also be increased during this time as the neighboring readers might transfer their load to the current reader. So at the beginning of each time step, the reader determines its load by checking the state of all the tags residing inside its proximity.

All of the readers present in the system simultaneously run the aforementioned algorithm which may cause the overall degradation of the fairness index of the system in first few time steps. However, as the tags are more evenly distributed over the course of multiple time steps, the fairness index improves significantly. These behaviors are discussed furthers in Sect. 7.

Algorithm 1 CA Based Load Balancing Algorithm (CALBA)

```
1: procedure BALANCELOAD(Reader R_i)        ▷ reader R_i will run the next steps at time t
2:     L_i ← 0.                              ▷ initialize the current load to 0 before updating
3:     for each T_k in neighborTags do
4:         if getStatus(T_k) == R_i then     ▷ check if the tag is assigned to R_i
5:             L_i ← L_i + 1
6:         else if T_k is orphan then
7:             faultTolerantProcedure(T_k, metric)
8:         end if
9:     end for
10:    for each R_j in neighborReaders do
11:        getStatus(R_j)                    ▷ load information of readers is stored in their state
12:    end for
13:    fairnessIndex ← localFairnessIndex(R_i)
14:    for each R_j in neighborReaders do
           ▷ R_i tries to transfer its own tags to its neighbors to improve fairness index
15:        for each common tag T_l between R_i and R_j do
16:            if getStatus(T_l) == R_i then
17:                if updatedLocalFairnessIndex(L_i − 1, L_j + 1) > fairnessIndex then
                       ▷ check if transferring T_l to R_j improves the fairness index
18:                    setStatus(T_l, R_j)   ▷ update the status of the tag for the new reader
19:                    L_i ← L_i − 1         ▷ update the load information of R_i
20:                end if
21:            end if
22:        end for
23:    end for
24: end procedure
```

Table 1 The counter-clockwise distance assigned to a tag for each of the feasible positions around the reader when $IR = 1$

x − 1, y − 1 (distance 2)	x − 1, y (distance 1)	x − 1, y + 1 (distance 8)
x, y − 1 (distance 3)	x, y (distance 0)	x, y + 1 (distance 7)
x + 1, y − 1 (distance 4)	x + 1, y (distance 5)	x + 1, y + 1 (distance 6)

6 Fault tolerant procedure

6.1 Algorithm metric

The proposed fault tolerant procedure is designed in such a way that it can operate using two different metrics. The first metric is the tag load of a reader. If a tag becomes orphan then our fault tolerant procedure assigns the orphan tag to the neighboring reader that has the lowest number of tags assigned to it.

The second metric is the fairness index which is already applied in our proposed load balancing algorithm. In this case, the orphan tag is selected by the reader for which the local fairness index will be maximized.

6.2 Proposed fault tolerant procedure for CALBA

Since the fault tolerant procedure is added as a part of the proposed load-balancing algorithm, the CA model used in this case is almost identical to the previous case. There are some minor changes and those are described below:

6.2.1 State representation

The state of each tag indicates the reader which is currently monitoring the tag. A reader's state signifies its current load (the number of tags currently assigned to it). If a reader is defective, the state of that reader cannot be read. A reader can check the state of its neighboring tags and readers to determine whether a tag is orphan or not.

6.2.2 Neighborhood

We consider a radius of 1 or 2 for the communication between a tag and a reader. The communication range between two readers is the double of that between a reader and a tag. This ensures that a reader will be able to check whether any other reader which is currently interrogating a neighboring tag is defective or not.

6.2.3 Transition rules

For each of the neighboring orphan tags, a reader executes the following steps:

1. (line 2) The reader gets information about all the readers that are the neighbor of the orphan tag.
2. (lines 3–21) If tag load is selected as the metric of the algorithm then the reader checks which of the neighboring readers of the orphan tag has the lowest number of tags associated with them. If the reader itself has the minimum tag load then it becomes associated with the orphan tag by updating its own state and also the state of the tag. If more than one reader has the minimum tag load, the reader checks the distance between the readers and the orphan tag. The function *nDistance*() is used to check this distance. If the reader is closest to the orphan tag then it becomes associated with the tag. The *nDistance*() function has been described in algorithm 3. If there are other readers that also have the minimum distance, the reader finds out which reader comes first in the counter-clockwise direction from the orphan tag. This is checked using the *cDistance*() function which is described in algorithm 4. For the counter clock-wise case, distance is measured by the procedure stated in Table 1 (suppose, the tag is on the x, y co-ordinate of the RFID grid and the readers are on any co-ordinate from $[(x-1, y-1), (x+1, y+1)]$.)
3. (lines 22–35) If fairness index is selected as the metric of the algorithm, the reader checks for each neighboring reader and what the local fairness index would be if the orphan tag was associated with that reader. If the reader finds that the maximum fairness index gain would be achieved by associating the tag with the current reader, it updates the state of the orphan tag and its own state accordingly.

Algorithm 2 The Fault Tolerant Procedure for CALBA

procedure FAULTTOLERANTPROCEDURE($T_k, metric$) ▷ A reader R_i executes this algorithm to determine if it should be associated with the orphan tag T_k
  $neighborReaders \leftarrow getNeighbors(T_k)$ ▷ Since current reader R_i is also the neighbor of the orphan tag T_k it is also considered in the list of $neighborReaders$
  if $metric$ is tag load **then**
    $minLoad \leftarrow \infty$
    $associatedReader \leftarrow -1$
    for each R_j in $niehgborReaders$ **do**
      $tempLoad \leftarrow getLoad(R_j)$
      if $tempLoad < minLoad$ **then**
        $minLoad \leftarrow tempLoad$
        $associatedReader \leftarrow R_j$
      else if $tempLoad == minLoad$ **then**
        if $nDistance(associatedReader, T_k) > nDistance(R_j, T_k)$ **then**
          $associatedReader \leftarrow R_j$
        else if $nDistance(associatedReader, T_k) == nDistance(R_j, T_k)$ **then**
          if $cDistance(associatedReader, T_k) > cDistance(R_j, T_k)$ **then**
            $associatedReader \leftarrow R_j$
          end if
        end if
      end if
    end for
    $setStatus(T_k, associatedReader)$
  else if $metric$ is fairness index **then**
    $maxLocalFairnessIndex \leftarrow -1$
    $associatedReader \leftarrow -1$
    for each R_j in $niehgborReaders$ **do**,
      $tempAssociatedReader \leftarrow associatedReader$
      $associatedReader \leftarrow R_j$
      $tempLocalFairnessIndex \leftarrow localFairnessIndex(associatedReader)$
      if $tempLocalFairnessIndex > maxLocalFairnessIndex$ **then**
        $maxLocalFairnessIndex \leftarrow tempLocalFairnessIndex$
      else
        $associatedReader \leftarrow tempAssociatedReader$
      end if
    end for
    $setStatus(T_k, associatedReader)$
  end if
end procedure

Algorithm 3 The Reader and Tag Distance Calculating Algorithm

procedure NDISTANCE(R_i, T_j) ▷ The distance between reader R_i and tag T_j is calculated
  return $max(abs(R_i[row] - T_j[row]), abs(R_i[column] - T_j[column]))$
end procedure

Algorithm 4 The Reader and Tag Counter-Clockwise Distance Calculating Algorithm

```
1: procedure cDistance(R_i, T_j) ▷ The couter-clockwise distance between reader R_i and
   tag T_j is calculated
2:     if R_i[row] < T_j[row]&&R_i[column] == T_j[column] then
3:         return 1
4:     end if
5:     if R_i[row] < T_j[row]&&R_i[column] < T_j[column] then
6:         return 2
7:     end if
8:     if R_i[row] == T_j[row]&&R_i[column] < T_j[column] then
9:         return 3
10:    end if
11:    if R_i[row] > T_j[row]&&R_i[column] < T_j[column] then
12:        return 4
13:    end if
14:    if R_i[row] > T_j[row]&&R_i[column] == T_j[column] then
15:        return 5
16:    end if
17:    if R_i[row] > T_j[row]&&R_i[column] > T_j[column] then
18:        return 6
19:    end if
20:    if R_i[row] == T_j[row]&&R_i[column] > T_j[column] then
21:        return 7
22:    end if
23:    if R_i[row] < T_j[row]&&R_i[column] > T_j[column] then
24:        return 8
25:    end if
26: end procedure
```

7 Simulation result

7.1 Simulation environment

All the algorithms are implemented using the Python programming language. We vary the grid size, the number of readers and tags and also the interrogation and communication radii to simulate different scenarios. In all the scenarios, the placement of the readers and the tags in the network is selected at random. The tags are also randomly assigned to any of their neighboring readers. Each of the algorithms is run 20 times in every scenario to calculate their average performance.

7.2 Result comparison for the load balancing algorithms

We name our proposed algorithm as "cellular automaton based load balancing algorithm" (CALBA). To validate our proposed algorithm, we compare it to the load-balancing tag assignment algorithm (LBTA) of Xie et al. (2012) and also the centralized maximum flow (Max Flow) algorithm proposed in Dong et al. (2007). Each reader in LBTA uses a diffusion parameter (α, *where* $0 \leq \alpha \leq 1$) to transfer a portion of its tag load to its neighboring readers. Let, two neighboring readers R_i and R_j have tag loads w_i and w_j respectively where $w_i > w_j$. The common tags between R_i and R_j is the set S_{ij} where $S_{ij} > \alpha \times (w_i - w_j)$. According to LBTA, reader R_i can transfer $\alpha \times (w_i - w_j)$ to reader R_j in this case. Here, varying the value of α provides different levels of performance for LBTA. There is no fixed value of the parameter that provides the best performance in all possible scenarios. This is one of the drawbacks of LBTA. Moreover, LBTA cannot specify the exact tags that should be transferred from one reader to another. It can only determine the total number of tags that can be transferred.

The reader to reader communication performed in LBTA can be completely avoided by storing additional information in each of the tags. In this case, the readers write down their current load in each of the tags residing within its proximity. Other readers can read this information from the tags to perform the necessary calculations. However, the high cost associated with the read/write operations usually dissuades the use of such models. Adopting the reader to reader communication in our algorithm results in communication complexity similar to LBTA. We compare the performance of LBTA, CALBA, and the centralized maximum flow algorithm through their fairness index, the number of iterations required to achieve that fairness index and the maximum reading time for a single reader in Table 2. For our simulations, we consider four different values of the diffusion parameter, α (0.1, 0.3, 0.5, 0.7) in LBTA. The results are shown for readers with communication radius 2 and interrogation radius 1.

Table 2 Comparison among LBTA, CALBA and Max Flow for Interrogation Radius 1

Grid size	Numbeer of readers	Number of tags	Algorithms		Initial fairness index	Iterations	Final fairness index	Maximum reading time
70×70	3500	7000	LBTA	$\alpha = 0.1$	0.66	1	0.66	9.1
				$\alpha = 0.3$		2	0.77	5.5
				$\alpha = 0.5$		5	**0.89**	5.2
				$\alpha = 0.7$		6	0.88	6.2
			CALBA			5	**0.94**	3.4
			Max flow			N/A	**0.93**	2.8
60×60	2500	5000	LBTA	$\alpha = 0.1$	0.65	1	0.65	9.3
				$\alpha = 0.3$		2	0.80	6.5
				$\alpha = 0.5$		5	**0.89**	4.9
				$\alpha = 0.7$		6	0.89	5.6
			CALBA			4	**0.94**	3.3
			Max flow			N/A	**0.92**	2.8
50×50	2000	4000	LBTA	$\alpha = 0.1$	0.66	1	0.66	8.8
				$\alpha = 0.3$		2	0.79	5.2
				$\alpha = 0.5$		5	**0.91**	3.8
				$\alpha = 0.7$		7	0.90	5.0
			CALBA			5	**0.94**	3.2
			Max flow			N/A	**0.94**	2.5
40×40	1200	2400	LBTA	$\alpha = 0.1$	0.66	1	0.66	7.8
				$\alpha = 0.3$		2	0.78	5.6
				$\alpha = 0.5$		5	**0.91**	3.95
				$\alpha = 0.7$		7	0.90	4.0
			CALBA			4	**0.95**	3.2
			Max flow			N/A	**0.92**	3.05
30×30	700	1500	LBTA	$\alpha = 0.1$	0.65	1	0.65	8.2
				$\alpha = 0.3$		2	0.79	5.8
				$\alpha = 0.5$		5	**0.91**	4.6
				$\alpha = 0.7$		7	0.90	4.8
			CALBA			5	**0.94**	3.6
			Max flow			N/A	**0.93**	3.4

Our simulation results confirm that none of the values of the diffusion parameter, α for LBTA constantly performs well. The values 0.5 and 0.7 appear to exhibit comparatively better performance. Our proposed algorithm CALBA constantly outperforms LBTA by $\approx$ 4% which is quite a significant margin for the large RFID systems. CALBA also requires less number of iteration to achieve its maximum fairness index levels. Similar comparison results are shown in Table 3 for communication radius 1 and interrogation radius set to 2. The performance gap between CALBA and LBTA is less significant ($\approx$ 2%) in this case. The reason behind this scenario is that, when we use an interrogation radius of 2, a tag under the surveillance of two readers positioned in opposite direction may fail to get migrated to improve the fairness index.

7.3 Performance of CALBA with fault tolerant procedure

We compare the performance of CALBA with our proposed fault-tolerant procedure for two different metrics. We also observe the performance of LBTA where no fault tolerant approach is implemented. All of the comparisons are done by executing the algorithm in an identical instance of the network.

For our simulation, we consider 5 different scenarios. In every scenario, the number of readers, tags and the grid size is fixed. Now each of the scenarios is run for 10 different instances. At the beginning of each instance, the readers and tags are placed randomly. We also randomly select some of the readers to be damaged for every instance. Afterwards, we observe the performance of our fault-tolerant procedure

Table 3 Comparison among LBTA, CALBA and max flow algorithm for interrogation radius 2

Grid size	Number of readers	Number of tags	Algorithms		Initial fairness index	Iterations	Final fairness index	Maximum reading time
70×70	3500	7000	LBTA	$\alpha = 0.1$	0.66	1	0.66	8.7
				$\alpha = 0.3$		2	0.80	5.2
				$\alpha = 0.5$		6	**0.94**	4.9
				$\alpha = 0.7$		8	0.94	4.65
			CALBA			5	**0.95**	3.3
			Max flow			N/A	**0.95**	2.5
60×60	2500	5000	LBTA	$\alpha = 0.1$	0.65	1	0.65	8.8
				$\alpha = 0.3$		2	0.80	4.67
				$\alpha = 0.5$		6	**0.94**	4.1
				$\alpha = 0.7$		6	0.94	4.0
			CALBA			5	**0.95**	3.65
			Max flow			N/A	**0.93**	2.05
50×50	2000	4000	LBTA	$\alpha = 0.1$	0.66	1	0.66	8.2
				$\alpha = 0.3$		2	0.81	4.8
				$\alpha = 0.5$		7	0.94	3.9
				$\alpha = 0.7$		7	**0.94**	4.6
			CALBA			4	**0.95**	3.2
			Max flow			N/A	**0.92**	2.3
40×40	1200	2400	LBTA	$\alpha = 0.1$	0.65	1	0.65	8.0
				$\alpha = 0.3$		2	0.81	4.8
				$\alpha = 0.5$		5	**0.94**	4.0
				$\alpha = 0.7$		7	0.94	4.05
			CALBA			4	**0.96**	3.3
			Max flow			N/A	**0.94**	2.6
30×30	700	1500	LBTA	$\alpha = 0.1$	0.67	1	0.67	8.3
				$\alpha = 0.3$		2	0.83	4.6
				$\alpha = 0.5$		9	**0.93**	3.7
				$\alpha = 0.7$		11	0.93	3.5
			CALBA			6	**0.95**	3.3
			Max flow			N/A	**0.91**	3.0

using the two different metrics and also the performance of LBTA where no fault tolerance is used. Our simulation results (Table 4) indicate that in all cases, our fault tolerant procedure assigns the orphan tags successfully to other readers while LBTA algorithm fails to do so.

7.4 CUDA implementation of the proposed load balancing algorithm

To take the greater advantage of all the available cores of a graphics processing unit (GPU), we implement our algorithm in parallel programming environment. NVIDIA CUDA (Garland et al. 2008) is used as the parallel programming tool.

For the experiment, we initially use NVIDIA GeForce GT 525 m as the GPU which has compute capability 2.1. It is observed from our experiment that our algorithms implementation in CUDA provides identical results compared to the implementation in Python (Munir et al. 2016). However, the computation time in CUDA is faster than the alternate implementation in Python or any other serial implementation based on the central processing unit (CPU) (Nickollsa et al. 2008). CUDA allows programmers to write host code (CPU) and device code (GPU) in a single source program (Garland et al. 2008; NVIDIA 2017). The device code is known as Kernel. The multiple GPU threads can execute these kernels in parallel. All the threads that are in a parallel execution phase form a grid. It is observed that CUDA implementation of our algorithm is relatively faster than the Python implementation. Dynamic Parallelism can further improve the time complexity. However, Dynamic parallelism can cause an overhead issue for huge kernel launch which will degrade the speed (Wang and Yalamanchili 2014; Yang et al. 2015; Chen and Shen 2015). Moreover, requirement of expensive GPU makes it cost inefficient.

Table 4 Comparison of fault tolerant procedure with different metrics

Grid size	Number of readers	Number of tags	Algorithm	metric used	Orphan tags
70×70	3500	7000	CALBA	Tag load	0
				Fairness index	0
			LBTA	N/A	404
60×60	2500	5000	CALBA	Tag load	0
				Fairness index	0
			LBTA	N/A	289
50×50	2000	4000	CALBA	Tag load	0
				Fairness index	0
			LBTA	N/A	231
40×40	1200	2400	CALBA	Tag load	0
				Fairness index	0
			LBTA	N/A	140
30×30	700	1500	CALBA	Tag load	0
				Fairness index	0
			LBTA	N/A	81

8 Conclusion

In this paper, we develop and implement (in a parallel environment) a strictly localized load balancing algorithm as a solution to the fast tag reading problem of an RFID system. We express the addressed problem as a load balancing problem of the RFID system and exploit cellular automaton (CA) to propose a solution to the problem. We consider an RFID system with multiple readers where the readers communicate among themselves prior to tag interrogation to avoid collision among the tags. We also introduce a set of rules to ensure the successful collection of information even if some of the readers become defective which confirms the fault tolerance feature of our proposed load balancing algorithm. We validate our proposed algorithm through rigorous simulation and compare the performance of our algorithm with state of the art algorithms. Comparison results suggest that the proposed localized load balancing algorithm outperforms the existing localized solution and gives a competitive result compared to the centralized algorithm. If any existing reader becomes faulty, our algorithm successfully assigns its associated tags to other readers while balancing the load. Currently, our work only addresses the RFID systems with static tags. In the future, we intend to build upon this work so that similar problems can be solved with mobile tags.

References

Ali K, Alsalih W, Hassanein H (2011a) Set-cover approximation algorithms for load-aware readers placement in RFID networks. In: 2011 IEEE international conference on communications (ICC), pp 1–6. https://doi.org/10.1109/icc.2011.5963396

Ali K, Hassanein HS, Alsalih W (2011b) Using neighbor and tag estimations for redundant reader eliminations in RFID networks. In: 2011 IEEE wireless communications and networking conference, pp 832–837

Bhatia M, Sood SK (2018) Internet of things based activity surveillance of defence personnel. J Ambient Intell Hum Comput 9(6):2061–2076. https://doi.org/10.1007/s12652-017-0507-3

Bilodeau JS, Bouzouane A, Bouchard B, Gaboury S (2018) An experimental comparative study of rssi-based positioning algorithms for passive RFID localization in smart environments. J Ambient Intell Huma Comput 9(5):1327–1343. https://doi.org/10.1007/s12652-017-0531-3

Bouet M, Pujolle G (2008) A range-free 3-d localization method for RFID tags based on virtual landmarks. In: PIMRC, pp 1–5

Bulusu N, Heidemann J, Estrin D (2000) Gps-less low cost outdoor localization for very small devices. In: IEEE pervasive computing magazine, pp 28–34

Campioni F, Choudhury S, Al-Turjman F (2018) Readers scheduling for RFID networks in the IOT era. In: 2018 IEEE international conference on communications workshops (ICC Workshops), pp 1–6. https://doi.org/10.1109/ICCW.2018.8403753

Capetanakis J (1979) Tree algorithms for packet broadcast channels. IEEE Trans Inf Theory 25(5):505–515

Chen G, Shen X (2015) Free launch: optimizing GPU dynamic kernel launches through thread reuse. In: In Proceedings of the 48th international symposium on microarchitecture (MICRO-48). ACM, New York, NY, USA, pp 407–419

Chen Q, Hoilun N, Yunhao L (2008) Cardinality estimation for large-scale RFID systems. In: Sixth annual IEEE international conference on pervasive computing and communications (PerCom), pp 30–39

Chiu D, Jain R (1989) Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. Comput Netw ISDN Syst 17(1):1–14

Dhas V, Muthukaruppan R, Balakrishnan K, Ganesan R (2010) Optimal solution for RFID load balancing. Communications in computer and information science. Springer, Berlin

Dominikus S (2011) Medassist—a privacy preserving application using RFID tags. In: IEEE international conference on RFID-technologies and applications (RFID-TA), pp 370–375

Dong C, Guiran C, Jiajia L, Jie J (2011) Study on the interconnection architecture and access technology for internet of things. In: International conference on computer science and service system (CSSS), pp 1744–1748

Dong Q, Shukla A, Shrivastava V, Agrawal D, Banerjee S, Kar K (2007) Load balancing in large-scale RFID systems. In: IEEE INFOCOM 2007—26th IEEE international conference on computer communications, pp 2281–2285

Floerkemeier C (2006) Transmission control scheme for fast rfid object identification. In: Fourth annual IEEE international conference on pervasive computing and communications workshops (PerCom workshops 2006), pp 457–462

Garland M, Grand SL, Nickolls J, Anderson J, Hardwick J, Morton S, Phillips E, Zhang Y, Volkov V (2008) Parallel computing experiences with CUDA. In: IEEE micro 28, vol 4, pp 13–27

Garzon M (1995) Models of massive parallelism: analysis of cellular automata and neural networks. Springer, Berlin

He T, Huang C, Blum BM, Stankovic JA, Abdelzaher T (2003) Range-free localization schemes for large scale sensor networks. In: MobiCom, pp 81–95

Hsu HH, Chen BK, Lin CY, Barolli L, Takizawa M (2011) Danger warning via fuzzy inference in an RFID-deployed environment. J Ambient Intell Hum Comput 2(4):285–292. https://doi.org/10.1007/s12652-011-0047-1

Irfan N, Yagoub MCE, Hettak K (2011) Redundant reader elimination approaches for RFID networks. Springer, Berlin, pp 396–405

Jang S, Lee J (2008) Fuzzy logic control-based load balancing agent for distributed RFID systems, lecture notes in computer science, vol 5226. Springer, Berlin

Jihoon M, Wonjun L (2006) Adaptive splitting protocols for RFID tag collision arbitration. In: 7th ACM international symposium on mobile ad hoc networking and computing, Florence, Italy

Lawrence G (1975) Aloha packet system with and without slots and capture. ACM SIGCOMM Comput Commun Rev 5:28–42

Meddeb A, Jaballah A (2017) Algorithm for readers arrangement without collision in RFID networks. In: 2017 18th international conference on parallel and distributed computing, applications and technologies (PDCAT), pp 316–321. https://doi.org/10.1109/PDCAT.2017.00059

Munir A, Hossen MS, Choudhury S (2016) Localized load balancing in RFID systems. In: TPNC, pp 34–45

Murali K, Thyaga N (2006) Fast and reliable estimation schemes in RFID systems. In: 12th annual international conference on mobile computing and networking, Los Angeles, CA, USA

Nickollsa J, Buck I, Garland M, Skadron K (2008) A performance study of general-purpose applications on graphics processors using CUDA. In: Queue—GPU computing, pp 40–53

NVIDIA (2017) CUDA C Programming Guide. http://docs.nvidia.com/cuda/cuda-c-programming-guide

Qunfeng D, Shukla A, Shrivastava V, Agrawal D, Banerjee S, Kar K (2007) Load balancing in large-scale RFID systems. In: IEEE INFOCOM 2007—26th IEEE international conference on computer communications, pp 2281 – 2285

Rashid N, Choudhury S, Salomaa K (2016) Carre: Cellular automaton based redundant readers elimination in RFID networks. In: 2016 IEEE international conference on communications (ICC), pp 1–6. https://doi.org/10.1109/ICC.2016.7510604

Rashid N, Choudhury S, Salomaa K (2018) Localized algorithms for redundant readers elimination in RFID networks. Int J Parallel Emerg Distrib Syst 20:1–12. https://doi.org/10.1080/17445760.2017.1419242

Sanpechuda T, Kovavisaruch L (2008) A review of RFID localization: applications and techniques. In: ECTI-CON, p 769–772

Vinod N, Lixin G (2007) Energy-aware tag anti-collision protocols for RFID systems. In: Fifth IEEE international conference on pervasive computing and communications, pp 23 – 36

Wang C, Wu H, Tzeng NF (2007) RFID-based 3-d positioning schemes. In: INFOCOM, pp 1235–1243

Wang J, Yalamanchili S (2014) Characterization and analysis of dynamic parallelism in unstructured GPU applications. 2014 IEEE international symposium on workload characterization (IISWC), Raleigh, NC, pp 51–60

Wang YC, Liu SJ (2017) Minimum-cost deployment of adjustable readers to provide complete coverage of tags in RFID systems. J Syst Softw 134:228–241. https://doi.org/10.1016/j.jss.2017.09.015. http://www.sciencedirect.com/science/article/pii/S0164121217302054

Wu F, Xu L, Kumari S, Li X, Das AK, Shen J (2018) A lightweight and anonymous RFID tag authentication protocol with cloud assistance for e-healthcare applications. J Ambient Intell Hum Comput 9(4):919–930. https://doi.org/10.1007/s12652-017-0485-5

Xie K, Cao J, Wen J (2012) Distributed load-balancing algorithm for fast tag reading. Int J Parallel Emerg Distrib Syst 28(5):434–448

Yang Y, Li C, Zhou H (2015) CUDA-NP: realizing nested thread-level parallelism in GPGPU applications. J Comput Sci Technol 49:93

Yoon W, Vaidya N (2010) RFID reader collision problem: performance analysis and medium access. Wirel Commun Mob Comput 2010:1–24

Zhang W, He ZY, Ma LM, Salem R, Han L (2018) Ap load balance strategy in face of high user density. J Ambient Intell Hum Comput 2018:1. https://doi.org/10.1007/s12652-018-0690-x

Zhu W, Cao J, Xu Y, Yang L, Kong J (2014) Fault-tolerant RFID reader localization based on passive RFID tags. IEEE Trans Parallel Distrib Syst 25:2065