Come and Be Served: Parallel Decoding for COTS RFID Tags

Jiajue Ou, Mo Li
School of Computer Engineering
Nanyang Technological University
{jou1,limo}@ntu.edu.sg

Yuanqing Zheng
Department of Computing
The Hong Kong Polytechnic University
csyqzheng@comp.polyu.edu.hk

ABSTRACT

Current commodity RFID systems incur high communication overhead due to severe tag-to-tag collisions. Although some recent works have been proposed to support parallel decoding for concurrent tag transmissions, they require accurate channel measurements, tight tag synchronization, or modifications to standard RFID tag operations. In this paper, we present *BiGroup*, a novel RFID communication paradigm that allows the reader to decode the collision from multiple COTS (commodity-off-the-shelf) RFID tags in one communication round. In *BiGroup*, COTS tags can directly join ongoing communication sessions and get decoded in parallel. The collision resolution intelligence is solely put at the reader side. To this end, *BiGroup* examines the tag collisions at RFID physical layer from constellation domain as well as time domain, exploits the under-utilized channel capacity due to low tag transmission rate, and leverages tag diversities. We implement *BiGroup* with USRP N210 software radio that is able to read and decode multiple concurrent transmissions from COTS passive tags. Our experimental study gives encouraging results that *BiGroup* greatly improves RFID communication efficiency, i.e., 11× performance improvement compared to the alternative decoding scheme for COTS tags and 6× gain in time efficiency when applied to EPC C1G2 tag identification.

Categories and Subject Descriptors

C2.1[Network Architecture and Design]: Wireless communication

Keywords

RFID systems; physical layer; parallel decoding

1. INTRODUCTION

RFID (Radio Frequency IDentification) technology has been extensively used in various applications, such as warehouse inventory [8, 40, 41], object tracking [25–27, 32, 35], human-computer interaction [7, 33], powerless sensing [6, 38], etc. Existing RFID communication standards like EPCglobal C1G2 [1] employ slotted aloha channel access and sequentially read tags at random time slots. The communication efficiency in current RFID systems remains low for two main reasons. (1) Concurrent transmissions of more than one COTS tags would collide and none of the transmission can be decoded. (2) RFID tags send data at low data rates (e.g., 16 kbps for backscatter link frequency of 64 kHz and Miller 4 coding [1])) using on-off keying that cannot fully utilize the channel capacity even when the wireless channel quality is high.

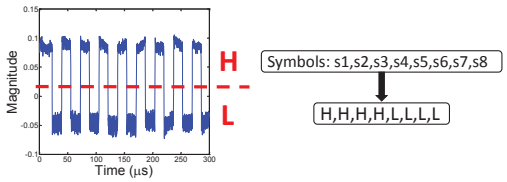
To improve the communication efficiency, recent works [3, 12, 31] explore the possibility of letting RFID tags transmit in parallel and separating the collided transmissions at the reader. These prior designs, however, require substantial modifications to COTS tags to enable collision resolution. For example, Buzz [31] needs to instrument tags to avoid inter-slot interference and perform channel measurements. BST [12] explicitly coordinates tag transmissions and misaligns tag signal edges to separate their signals. Some other designs [23, 24] require coding mechanisms (e.g., CDMA, rateless codes [11]) adopted on RFID tags to facilitate collision recovery. As a result, existing parallel decoding approaches cannot fully support standardized COTS tags in widely deployed RFID systems. While several billions of COTS RFID tags with globalized standards are already in use worldwide, these recently proposed schemes can hardly benefit them.

In this work, we consider an ideal parallel decoding scheme that is able to decode packet collisions from COTS tags. We ask the question: *Can we enable "come and be served" parallel transmissions without any extension to COTS RFID tags?* The design goal is twofold: (1) we aim to enable parallel tag transmissions without the need of any modification to the COTS tags; (2) we aim to support COTS tags to join on-going communication sessions without specific coordination. The "come and be served" design does not tamper C1G2 logics on COTS tags and can provide direct benefits within the C1G2 framework. For example, the tag identification procedure of C1G2 requires individual RFID tags to send RN16 for channel contention. A collision occurs when multiple tags send RN16 in the same slot and the slot is wasted. "Come and be served" parallel decoding allows the reader to acquire RN16 codes of multiple tags from the collision and thus improves identification efficiency. According to our experimental results (which we will detail in Section 4), parallel decoding of merely 2 – 5 tags suffices to improve the tag identification time efficiency by 6×. For another example, the READ command in C1G2 comes after tag identification and reads data from one tag at a time. With "come and be served" design, the READ command can be extended to concurrently read data from multiple tags. The READ sessions from different tags do not need to be synchronized. Later transmissions can ride on on-going sessions. The throughput can be easily multiplied.

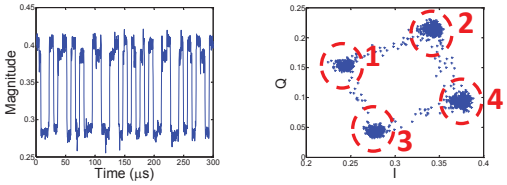
This paper presents *BiGroup* (Bipartite Grouping), that instantly decodes concurrent transmissions of COTS tags, by purely extend-

MobiCom'15, September 7–11, 2015, Paris, France.

DOI: http://dx.doi.org/10.1145/2789168.2790101.



(a) Decoding one tag transmission based on signal magnitude.



(b) The combined signal magnitude when two tags collide (Left), and the complex physical symbols on the I-Q plane (Right).

Figure 1: Decoding tag signals from single and multiple concurrent transmissions.

ing the decoding intelligence at the RFID reader. Unlike existing approaches, *BiGroup* examines collided tag transmissions at both time domain (time series signal transitions) and constellation domain (complex symbols on constellation plane). Regulated by standard RFID coding schemes, e.g., FM0 or Miller coding, RFID tags flip their reflection states at bit boundaries in backscatter transmission, which results in transitions of the combined signals in the time domain. *BiGroup* iteratively extracts the sequence of each tag's signal transitions in the time domain, and connects with the physical symbol clusters in the constellation domain. As one tag only alternates between reflecting or absorbing states, all clusters can be bipartitely grouped into two according to that tag. We can thus produce a bipartite grouping for each tag involved in the transmission, and the state transitions between the two groups give the transmitted data of that tag.

We implement *BiGroup* on a USRP N210 based RFID reader that concurrently reads different models of COTS tags as well as programmable RFID tags. To the best of our knowledge, *BiGroup* is the first practical design that is able to provide accurate parallel decoding for COTS RFID tags within the C1G2 framework. In particular, compared with source separation methods in [28], *BiGroup* improves the success rate by 11× on average for decoding the collisions of 3 – 5 C1G2 passive tags. Our experimental results also demonstrate huge performance gain over existing schemes for non-COTS tags. The decoding capacity of *BiGroup* is affected by the channel quality and limited when the number of colliding tags increases. Nevertheless, our experimental case study demonstrates that concurrently decoding a small number of tags can already significantly improve the efficiency of some standard EPC C1G2 operations.

In the rest of the paper, we describe the background and motivation of our design in Section 2. We present the design details in Section 3. We evaluate *BiGroup* and present experimental results in Section 4. We detail related work in Section 5 and conclude the paper in Section 6.

2. BACKGROUND AND MOTIVATION

A passive RFID tag encodes its data by reflecting or absorbing incident carrier waves, resulting in two possible states: "High (H)" and "Low (L)". The nth tag's two alternative states are denoted by,

$$S_n = H_n \; or \; L_n \tag{1}$$

which exhibit two distinct signal magnitudes when received at an RFID reader as shown in Figure 1(a). The reader can thus decode the data using a magnitude threshold.

When multiple tags transmit simultaneously, their signals add up at the receiver. One collision state is a combination of all the tags' signal states. The possible collision states of N tags are denoted by,

$$S_{col} = [S_1, S_2, ..., S_N], \tag{2}$$

[.] means all combinations of $S_m = H_m$ and L_m, $m = 1, 2, ...N$. In the example of the collision from two tags, the four collision states are denoted as "HH", "HL", "LH" and "LL", respectively.

In commodity RFID systems, the reader solely examines the received signal magnitudes. As a result, the reader cannot distinguish different collision states when multiple tags transmit at the same time. For example, Figure 1(b) presents the combined signal magnitude detected at a commodity reader when two tags transmit simultaneously. The reader cannot determine the threshold to detect the states of either individual tag. When we plot the received physical symbols in the In-phase and Quadrature (I-Q) coordinates, however, we see that the symbols form four separable clusters, each representing one collision state. Thus, if we can associate each cluster to a specific collision state ("HH", "HL", "LH", or "LL"), the collision can be recovered. Generalized to N tags, the decoding goal is to identify the collision state S_{col}, which is to derive in each collision state whether the separate tag transmission state $S_m = H_m$ or L_m for $m = 1, 2, \ldots, N$.

A variety of approaches have been proposed in view of above observation to decode concurrent transmissions. Buzz [31] first coordinates tags to measure their channel coefficients. Assuming that the collided signal of multiple tags is linear composition of individual signals, Buzz recovers the transmitted signal of each tag with channel measurement. In Buzz, the RFID tags apply rateless codes to encode transmitted data. Other coding mechanisms like [24] can also be used on tags to facilitate collision resolution.

In a recent approach BST [12], tags transmit with allocated initial offsets and data rates in order to create misaligned signal edges. During backscattering, tags monitor whether their signal edges overlap and adjust offsets and data rates. For bootstrapping decoding and correcting decoding errors, tags need to insert known bits called "sentinel" bits at specific intervals in data packets. These demanded tag operations are not supported by standard COTS tags.

Although existing approaches allow parallel decoding for RFID tags programmed with specific logics, they do not meet the "come and be served" requirement and cannot support decoding COTS tags of standard operating logics. The reasons are as follows. First, in order to bootstrap collision recovery, many works require controlled channel measurement (e.g., using preambles [3, 16, 39] or coordinated transmissions [31, 34]) to understand channel coefficients of individual tags. The channel measurement requires non-standard coordination that is not supported by COTS tags. Extra communication and coordination overhead also brings down the efficiency. Second, many designs [3, 4, 16, 31] rely on precise tag synchronization so collided tag signals can align with each other. In practice, however, due to manufacturing diversities, COTS tags have non-identical clocks and respond asynchronously with distinct bit durations. Third, many works require non-standard operations from RFID tags apart from those defined in COTS tags (e.g, adaptive transmission offsets and data rates [12], extra "sentinel" bits

[12], specific tag coding [24, 31], etc.). Affording those operations on ASIC chips for COTS tags remains elusive.

Unlike existing works, *BiGroup* is designed to work with COTS tags. *BiGroup* does not require channel measurement, tag transmission synchronization, or any special operation at tag side. The COTS tags simply follow the standardized response logics when interrogated by the reader. A basic observation is that when the constellation plane is examined, regardless of where the physical symbol clusters are located (which is determined by the channel coefficients), as long as the clusters can be divided into two groups corresponding to the two transmission states ("H" or "L") of one tag, we can decode the data transmitted from that tag. We call it bipartite grouping. For the example case shown in Figure 1(b), we can decode one tag by separating symbols from the group consisting of cluster 1, 2 and symbols from the other group consisting of cluster 3, 4. The other tag can be decoded with the group of cluster 1, 3 and the group of cluster 2, 4.

In the general case of N tags, symbol clusters can similarly be grouped for each tag (e.g., tag n) as follows,

$$\begin{aligned} H_{n,col} &= [S_1, S_2, ...H_n..., S_N] \\ L_{n,col} &= [S_1, S_2, ...L_n..., S_N]. \end{aligned} \tag{3}$$

As more tags collide, however, it becomes increasingly difficult to accurately group the clusters with respect to each tag. *BiGroup* examines time domain tag state transitions to address the problem. Regulated by standard RFID coding schemes, e.g., FM0 or Miller coding, each tag transits between the reflecting and absorbing states at bit boundaries during its backscatter transmission. As such, at the bit boundaries of tag n, the reader knows that tag n would definitely transit from one of the collision states in $H_{n,col}$ to one in $L_{n,col}$, or vice versa. *BiGroup* carries such state transitions into the constellation domain and can thus identify symbol clusters belonging to either $H_{n,col}$ or $L_{n,col}$. As more state transitions are detected along with the time, *BiGroup* is able to eventually bipartitely group all clusters into two, that corresponds to $H_{n,col}$ and $L_{n,col}$. The data from tag n can thus be recovered from the temporal transitions between the two groups.

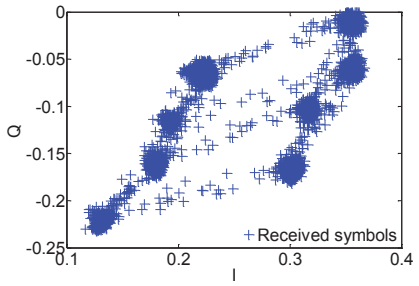
Designing and implementing *BiGroup* in practice entails substantial challenges. Without coordinated responses from COTS tags, it is challenging to identify state transitions of individual tags. The problem becomes more difficult if we cannot expect any prior knowledge of channel coefficients or linear dependency among collision states. At the same time, the design of *BiGroup* has to be efficiently implemented so the computational overhead imposed by operations like symbol clustering, boundary extraction, etc. can be properly accommodated. We detail the design and implementation of *BiGroup* in the next section.

3. DESIGN

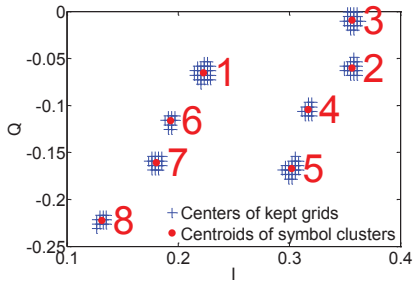
In this section, we first describe the symbol clustering method. We then introduce the design considerations and the principle of *BiGroup*. After that, we give implementation details of *BiGroup* in practice.

3.1 Symbol Clustering

One signal sample received at physical layer is represented as one complex symbol on the I-Q plane. Due to noises and interferences, received symbols of the same collision state are dispersed and scattered around a centroid position, forming a cluster. Before performing bipartite grouping for these clusters, we need to first identify the number of clusters and the symbols belonging to each of those clusters. We use the density based clustering algorithm which obviates the need for cluster number. To reduce the input size for faster clustering, *BiGroup* aggregates received symbols into grids (which are much fewer than symbols) and cluster those grids.



(a) Received symbols with three tags.



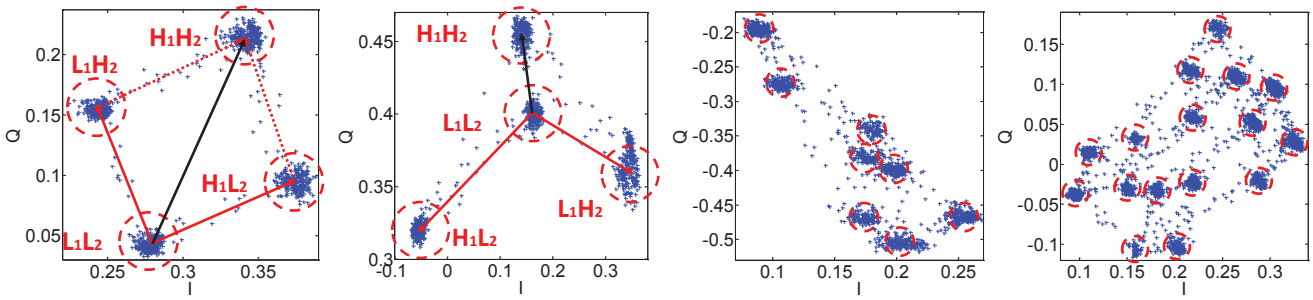
(b) Grids after filtering and cluster centroids.

Figure 2: Symbol clustering.

BiGroup divides the constellation plane into grids and denotes all the symbols confined within a grid using the grid center. The grid size can be adapted according to the background noise level. After a reader sends the QUERY command, tags remain absorbing states until they harvest sufficient amount of energy. During the charging stage, the cluster of received symbols corresponds to the all "L" collision state. These symbols can be averaged to derive the cluster centroid and the symbol dispersions can be measured to derive the background noise level. We thereafter set the grid size as 1/3 of the noise level. We filter out the grids with small number of symbols and feed the centroids of the remaining grids into the density based clustering algorithm (DBSCAN [9] in our implementation), which outputs the final symbol clusters. Figure 2 gives an example where 3 tags transmit to the reader concurrently. The received symbol samples are plotted in Figure 2(a). The kept grids and output cluster centroids are depicted in Figure 2(b). After that, each symbol will be marked by the label of its cluster. The received sequence of physical symbols can then be transformed into a sequence of cluster labels for later decoding.

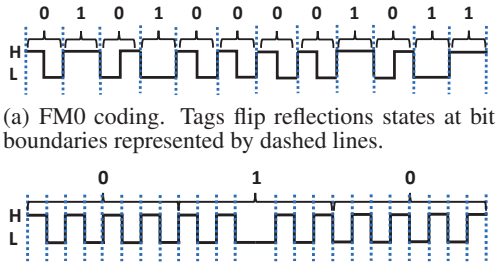
3.2 Bipartite Grouping using Linear Dependency?

Now we have the symbol clusters and each symbol is associated with one cluster. One possible approach to bipartitely group the clusters is to leverage the linear dependency among clusters. Theoretically when two complex signals collide, they add up linearly at the receiver. In the same way, when two tags backscatter the radio signal concurrently, the collided signal is the linear combination of these tags' channel coefficients. Thus we can observe the linear dependency among different collision states (i.e., symbols clusters). One example is shown in Figure 3(a), where



(a) Two tags with linear dependency. (b) Two tags with non-linear dependency. (c) Three tags with non-linear dependency. (d) Four tags with non-linear dependency.

Figure 3: Collision symbol clusters may exhibit non-linear dependency when multiple tags respond.



(a) FM0 coding. Tags flip reflections states at bit boundaries represented by dashed lines.

(b) Miller coding ($M = 4$). Similar periodic state changing patterns are observed. Dashed lines represent state flipping locations.

Figure 4: RFID coding property.

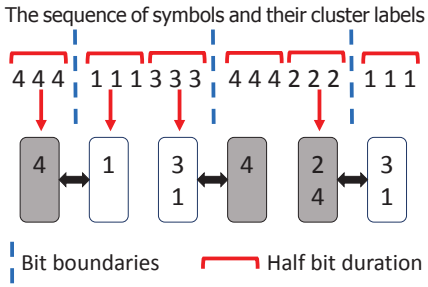


Figure 5: An illustrative example of bipartite grouping.

symbol vector "$L_1L_2 \rightarrow H_1H_2$" (illustrated with the black arrow) is approximately the linear addition of two vectors "($L_1L_2 \rightarrow H_1L_2$) + ($L_1L_2 \rightarrow L_1H_2$)" (illustrated with red arrows). The four cluster centroids roughly create a parallelogram. The group of clusters "H_1H_2" and "H_1L_2" are shifted from the group of clusters "L_1H_2" and "L_1L_2". The difference between the former two and latter two is the state of tag 1, meaning that they belong to opposite groups in bipartite grouping for tag 1. In a similar way, clusters "L_1H_2", "H_1H_2" and clusters "L_1L_2", "H_1L_2" are separated in terms of tag 2's state. In this case, we can leverage the linear addition property to do bipartite grouping. Same as other collision recovery methods [28, 31], such a bipartite grouping approach assumes the linear dependency among symbol clusters.

In practice, however, we also observe cases like Figure 3(b). We see that the vector '$L_1L_2 \rightarrow H_1H_2$" deviates from the linear addition of two vectors "($L_1L_2 \rightarrow H_1L_2$) + ($L_1L_2 \rightarrow L_1H_2$)". Such non-linear dependency is probably due to the change of tags' channel conditions by nearby tags. We hypothesize that when a tag backscatters alone, the reflected signal is mainly from the reader's carrier wave, but when multiple tags coexist, tags may backscatter the backscattered signals from nearby tags [21], that results in the non-linear dependency in the combined signals received at the reader. According to our measurement, the non-linear dependency becomes more obvious as more tags coexist as shown in Figure 3(c) and Figure 3(d) where 3 and 4 tags collide, respectively. Similar non-linear dependency has been reported in previous studies [12] as well. We note that the non-linear dependency is affected by various factors, e.g., the proximity and positions of tags, the data contents, etc., which are not under control of the RFID reader [12, 42]. In view of that, *BiGroup* has to perform bipartite grouping without assuming the linear dependency property.

3.3 *BiGroup* in Principle

In this section, we describe our key observations and the principle of *BiGroup*.

Data agnostic state flipping. COTS tags use standardized FM0 or Miller coding to encode data and the coding scheme in each communication round is specified by the reader [1]. They have a predictable state flipping pattern regardless of the transmitted data bits. We take FM0 coding (bi-phase space) for example. The bit-0 has an additional mid-bit phase inversion while bit-1 does not flip the phase as Figure 4(a) details. Nevertheless, FM0 inverts the phase at every bit boundary, so in the time domain we may always observe state transitions at bit boundaries independent of the data contents. The bit boundaries are dash lined in Figure 4(a). Similar deterministic state transitions can also be observed in Miller coding schemes as Figure 4(b) depicts. We note that such state transitions are compulsory for all CIG2 COTS tags so the RFID reader is able to track the backscatter frequencies. For brevity, we will later use FM0 codes as a vehicle to describe our method. Our approach can be generalized with slight modifications to handle Miller codes as well[1].

Overview of bipartite grouping. The bit boundaries identified for tag *i* divide the symbol clusters into two groups, corresponding to the "H" and "L" states of tag *i*. Since the state of the tag must flip at bit boundaries, we have the knowledge that the state before a bit boundary and state after the boundary must belong to the oppo-

[1]In Miller codes, the signal state flips with a fixed period proportional to the bit duration and with a small number of exceptions (less than 1/10). Majority voting technique in *BiGroup* tolerates such exceptions.

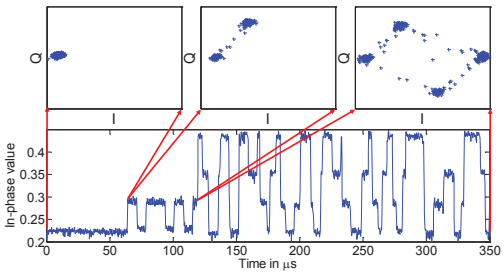


Figure 6: Tags join the transmission at different time.

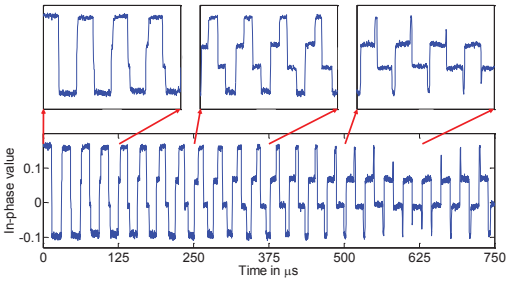


Figure 7: Two tags' signals misalign with each other due to different bit durations.

site groups, i.e., $H_{i,col}$ and $L_{i,col}$. Connecting to the constellation domain, the corresponding symbols (and their cluster labels) can thus be divided as well. As more bit boundaries of tag i are examined, more information can be accumulated on which clusters belong to opposite groups. All the clusters will eventually be bipartitely grouped.

Figure 5 illustrates the process using an example. The bit boundaries of tag i are identified and mapped onto the sequence of symbols and the associated sequence of cluster labels. According to the first bit boundary, cluster 4 and 1 are put in opposite groups. Then according to the second bit boundary, cluster 3 and 4 are put in opposite groups. Similarly, at the third bit boundary, cluster 2 is put in a group opposed to cluster 1's group. As we have already identified the cluster representing all "L" state when all tags are being charged, the group containing that cluster should be $L_{i,col}$, and the other group should be $H_{i,col}$. Note that the success of bipartite grouping does not rely on full detection of all bit boundaries. The process completes once an adequate number of bit boundaries are identified that allows all clusters to be distinguished. The process also tolerates inaccurate bit boundary detection, which we will detail in the next section.

3.4 *BiGroup* in Practice

To obtain the bipartite groups of each tag, we need to accurately extract bit boundaries for each individual tag and deal with imprecise bit boundaries with background noise. We first describe the tag-to-tag unsynchronization. We leverage such tag diversities to extract bit boundaries. We then present the method of bipartite grouping.

Unsynchronized tag signals. In practice, tag responses from different tags are usually unsynchronized for two reasons: different response delays and different bit durations.

Different tags have different response delays. A tag responds to a reader's QUERY with a delay. The length of the delay is largely determined by the tag's clock rate, power charging rate and strength of incident radio power. Due to manufacturing and tag location diversities, tags generally respond to the reader's QUERY with different initial offsets as illustrated in Figure 6. In the figure, we see that the first tag responds at around 60μs while the second joins the concurrent transmission at around 120μs. The three I-Q planes plot the snapshots of physical symbols received when no tag, one tag, and two tags respond, respectively.

Different tags have different bit durations. A bit duration refers to the time period of transmitting one data bit, which refers to the gap between two closest dashed lines for FM0 in Figure 4(a). RFID tags use digital clocks to control the backscatter link frequency. Due to the clock diversity, the bit durations are not identical for different tags [13, 37]. Commodity RFID readers can tolerate the link frequency variations by tracking bit boundaries when one tag responds. Figure 7 plots the collided signal when two tags simultaneously transmit the same alternating data sequences. We see that two signals which initially align with each other gradually misalign due to different bit durations.

When tags collide, due to unsynchronized starting time and different bit durations, their signals do not always align with each other. Since tags' starting time and bit durations are not known in advance, it is hard to figure out how signals may collide with each other. Thus, previous schemes [4, 31] generally consider the unsynchronization harmful and try to avoid the inter-bit interference among tags with explicit coordination. Unlike previous schemes, we leverage such misalignment properties of RFID tags to identify bit boundaries for individual tags respectively.

Extracting bit boundaries. COTS tags need to invert their states at each bit boundary to allow the RFID reader to recover and track backscatter link frequencies. As a result, when we examine the clusters in an I-Q plane as we receive physical symbols, we observe that the symbols transit from one cluster to another at bit boundaries.

We first detect all the cluster transitions due to state transitions of the RFID tags. After the symbol clustering, the sequence of symbols is transformed to a sequence of cluster labels. Based on the sequence of cluster labels, we identify cluster transitions in the sequence. We may detect each cluster transition by detecting the symbol which has a different cluster label from the preceding symbol. In practical implementation, instead of detecting cluster changes of individual symbols, we detect the cluster changes of f symbols with the same cluster label to f symbols with different clusters as a cluster transition so as to enhance detection robustness.

The detected cluster transitions involve two parts: compulsory transitions at bit boundaries and occasional transitions in bit durations (e.g., the state flip in the middle of the duration of bit "0"). We need to extract bit boundaries by mapping those compulsory cluster transitions to each tag's bit boundaries in time series.

We denote the time points that the bit boundaries of N collision tags fall at as $T = \bigcup_{i=1}^{N} T_i$, where $T_i = \{t_{i,k} | 1 \leq k \leq L\}$ represents L bit boundaries due to state transitions of tag i. We note that L is determined by the packet size and is known in advance to the RFID reader. As the tags respond simultaneously, the bit boundaries are interleaved and mixed together. Fortunately, because of the unsynchronization among tags, the bit boundaries of colliding tags do not completely coincide with each other. We incorporate the tag response delay and bit duration and describe the bit boundaries of tag i (i.e., T_i) as follows

$$T_i = \{a_i + kb_i | 1 \leq k \leq L\}, \quad (4)$$

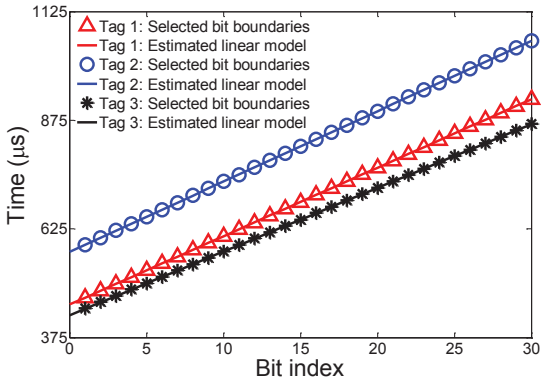


Figure 8: An illustrative example of identifying bit boundaries for three tags. The slope of a line represents the bit duration.

where a_i and b_i denote the starting transmission time and bit duration of tag i, and thus $a_i + kb_i$ represents the location of the kth bit boundary of tag i.

At first glance, we may determine a_i and b_i, and extract the bit boundaries of T_i with linear regression by optimizing a_i and b_i to minimize the residual errors as follows

$$\min_{\hat{a_i},\hat{b_i}} \sum_k \{||\hat{a_i} + k\hat{b_i} - t_{i,k}||^2\},\ k = 1, 2, 3, ..., L. \quad (5)$$

However, since the bit boundaries of different tags ($T_m, m = 1, 2, 3, ..N$) are mixed within detected cluster transitions and are unknown, we cannot trivially optimize a_i and b_i to extract T_i for tag i.

To solve the problem, we search over the possible ranges for a_i and b_i to fit L bit boundaries of T. For each candidate pair of a_i and b_i, specifically, we find the set of cluster transitions that minimizes the residual error. At the same time, we denote the corresponding achieved minimum residual error as R_i and the identified L bit boundaries as T_i, for each candidate pair respectively. We then find the pair of $\hat{a_i}$ and $\hat{b_i}$ and corresponding $\hat{T_i}$ that has the smallest $\hat{R_i}$ among all the R_is. After extracting $\hat{T_i}$ for tag i and decoding tag i, *BiGroup* removes both compulsory and occasional cluster transitions caused by tag i from the detected cluster transitions and iterates to extract bit boundaries for more tags. This iteration ends when there are not enough cluster transitions left.

Figure 8 plots one segment of the identified bit boundaries as well as their linear models for 3 tags. Taking tag 1 for example, the red marks represent the identified bit boundaries for tag 1 (i.e., T_1). The estimated linear model for tag 1 is depicted as the red line ($y = a_1 + xb_1$). We see that the identified bit boundaries fit well with the the linear model, indicating a very small residual error for tag 1. Similarly, our method extracts T_2 and T_3 for tag 2 and 3, respectively. Those identified bit boundaries for different tags will be used in determining bipartite grouping.

The search ranges for a_i and b_i can be determined in the following way. When tag i joins the transmission, its state switches from "L_i" to "H_i", resulting in new collision states and thus new symbol clusters in the I-Q plane (as the example in Figure 6 suggests). Therefore, we select each cluster's first a few transitions as the search range for a_i and let the fitting algorithm to search for the valid starting points. As the fitting algorithm is to find bit boundaries simply used for bipartite grouping not decoding, as long as a_i is one of the valid bit boundaries of tag i, the detected bit boundaries can be used to bootstrap bipartite grouping. The search range of bit duration b_i is set according to the possible link frequency ranges specified in C1G2 standard [1].

Algorithm 1 Majority voting in bipartite grouping

1: **INPUT**: vote for each cluster pair, e.g. V_{ij} for cluster i and cluster j
2: **OUTPUT**: classification of clusters for one certain tag
3: **PROCEDURE**:
4: Initialization:
5: Find $\hat{i}, \hat{j}$ s.t. $V_{\hat{i}\hat{j}} = \max\{V_{ij}\}$
6: group 1$\leftarrow$ cluster $\hat{i}$
7: group 2$\leftarrow$ cluster $\hat{j}$
8: **repeat**
9: **for** each cluster s that has not been grouped
10: **for** $k = 1, 2$
11: W_{sk} = mean $\{V_{sk_1}, V_{sk_2}, V_{sk_3}...\}$, for cluster $k_i \in$ group k
12: **end for**
13: **end for**
14: Find $\hat{s}, \hat{k}$ s.t. $W_{\hat{s}\hat{k}} = \max\{W_{sk}\}$
15: group $(3 - \hat{k})\leftarrow$ cluster $\hat{s}$ //Put cluster $\hat{s}$ into group $\hat{k}$
16: **until** all the clusters have been grouped

Bipartite grouping. After extracting bit boundaries for a tag, we determine bipartite grouping for the tag according to the principle introduced in Section 3.3. In practice, burst noise may result in wrong cluster labels and small fluctuations of the bit duration may cause shifted bit boundaries. Therefore, direct bipartite grouping may mistakenly separate some cluster pairs of the same group to opposite groups, that results in decoding errors. We apply a majority voting algorithm to improve bipartite grouping robustness.

For one tag, we first count the number of each cluster pair's occurrences on opposite sides of the tag's corresponding bit boundaries as the vote of this cluster pair. The vote of one cluster pair is denoted by $V_{i,j}$ for cluster i and j, $i \leq j$. For example, if there are a samples that belong to cluster i on one side of a bit boundary (within half duration), and b samples that belong to cluster j on the other side of the bit boundary (within half bit duration), we increase $V_{i,j}$ by $a \cdot b$.

We present the pseudocode of bipartite grouping for each tag in Algorithm 1. The algorithm initializes the classification with the cluster pair of the highest votes (line 4-7), i.e., if $V_{\hat{i}\hat{j}}$ has the largest value among all the votes, cluster $\hat{i}$ and cluster $\hat{j}$ are assigned to opposite groups. The remaining clusters are iteratively assigned to the two groups (line 8-16). In one iteration, for each ungrouped cluster s, we calculate the vote of cluster s and existing bipartite group k ($k = 1\ and\ 2$) as a whole, denoted by W_{sk}. W_{sk} is the average value of $\{V_{sk_1}, V_{sk_2}, V_{sk_3}...\}$, where clusters $k_1, k_2, k_3...$ are already assigned to group k. Then we find the highest value of W_{sk}, denoted by $W_{\hat{s}\hat{k}}$. If $\hat{k} = 1$, cluster $\hat{s}$ will be assigned to group 2, otherwise, cluster $\hat{s}$ will be assigned to group 1. The process iterates until all the clusters are put into one of the two groups.

In the end, the cluster group with the known all "L" state is identified as "L" group, and the opposite group is identified as "H" group. The received symbol sequence, with the knowledge of which cluster a symbol belongs to, is thus turned into a sequence of "H" and "L" states for each individual tag, and input to the conventional single tag decoder.

3.5 Put It Together

Figure 9 illustrates the key workflow of *BiGroup*, which comprises the following three main steps:

Symbol clustering. *BiGroup* first clusters the physical symbols in the I-Q plane. Each symbol is classified into one cluster and marked by the corresponding cluster label. The received symbol

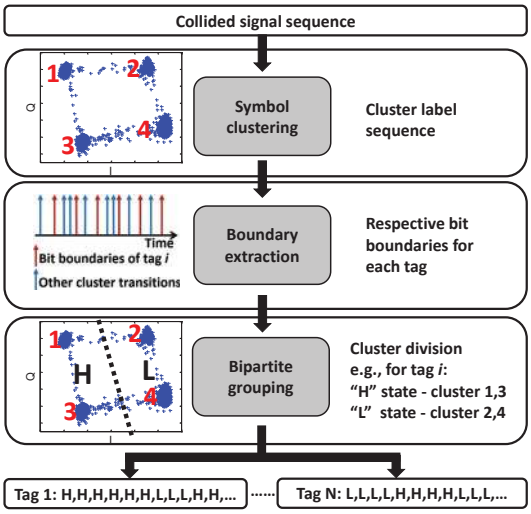


Figure 9: Workflow of *BiGroup*.

sequence is thus converted into a sequence of cluster labels for further processing.

Bit boundary extraction. *BiGroup* detects the time points of symbol transitions among clusters. *BiGroup* then extracts bit boundaries from cluster transitions and map to time series bit boundaries of different tags respectively.

Bipartite grouping. For each tag, *BiGroup* then divides the symbol clusters into bipartite groups of "H" state and "L" state. By examining the "H" or "L" states in the sequence of cluster labels, *BiGroup* outputs a sequence of binary states that represents the transmitted signal of that tag. Different tags have different bipartite groups and thus give different binary sequences. A conventional single tag decoder can then be applied on the binary sequence to decode the data of each tag.

3.6 Discussion

Computation overhead and time. The computation overhead of *BiGroup* mainly lies on symbol clustering and the time series search of bit boundaries. The symbol clustering algorithm used in *BiGroup* is performed on aggregated grids and its computation overhead is $O(k \log k)$, where k is the number of grids. The number of grids is influenced by the cluster dispersion and the size of a grid. Given cluster dispersion, a smaller grid size means higher resolution, but in the mean time results in more grids, thus higher computation overhead. To strike a balance between resolution and overhead, we set the side length of a grid as 1/3 of the cluster dispersion, and the number of grids is around 200. The computation overhead of time series search is $O(n^2p^2 \log(np))$, where n is the number of colliding tags and p is the number of packet bits used in searching. The above procedure does not have to apply to the whole packet, as it suffices to inspect part of the packet bits to obtain the bipartite grouping result (100 bits per packet in our experiments). Our current implementation of *BiGroup* is based on USRP N210 connected to a general PC equipped with 3.2GHz CPU and 16G memory, and the algorithm takes hundreds of μs to decode one collided packet. The computation time can be easily brought down by hardware implementation in the radio.

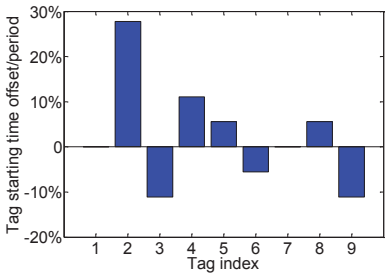
Decoding capacity and the gain. The decoding capacity of *BiGroup* is inevitably limited when the number of colliding tags increases. The channel quality essentially limits the resolution in symbol clustering. Ideally, received symbols of different collision states will fall at different positions on the I-Q plane and can always be separated. As SNR decreases or tag number increases, symbol clusters may overlap with each other, making symbol clustering more difficult and error prone. In our experiments, the throughput improvement from *BiGroup* peaks when 4 – 5 tags transmit simultaneously (more details in Section 4). Nevertheless, such decoding capacity can already significantly improve the efficiency of some standard EPC C1G2 operations. In these operations, the maximum number of colliding tags in most transmission slots can be easily controlled by methods such as frame slotted ALOHA access protocol. Take tag identification - the most widely used operation as an example. The reader sends a QUERY command and each tag contends for the channel by responding a random RN16 packet at a randomly selected time slot within the frame. The reader ACKs the RN16 it hears and the corresponding tag of the particular RN16 responds its EPC (tag ID). The conventional reader can only retrieve the RN16 code from a slot with single tag response, and thus has very low efficiency, i.e., with the optimized frame setting more than 63% of the slots are collided and thus wasted. With *BiGroup*, the reader is able to retrieve RN16 from colliding slots (up to 5 concurrent transmissions in our experiment) and thus less than 12% of the slots are wasted. Our experimental study shows that with the help of *BiGroup*, around 6× speedup can be achieved for C1G2 tag identification. Detailed results are presented in Section 4.

Figure 10: The Open RFID testbed.

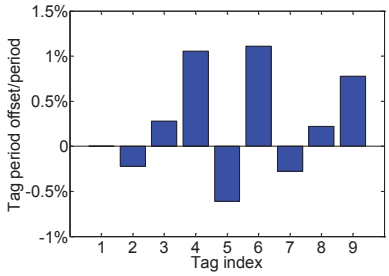
Impact of channel variation. In *BiGroup*, received symbols of different states are discriminated according to their clustering on I-Q plane. Channel variation may change the positions of received symbols and thus the cluster distribution. The transmission time of a typical RFID uplink packet, however, is as short as a few milliseconds, which is usually within channel coherence time. The *BiGroup* clustering algorithm naturally tolerates some level of channel variation, which is determined by the Euclidean distances among cluster centroids. In future work, we may explore the possibility of splitting one transmission into multiple time windows during decoding to address significant channel variations.

4. IMPLEMENTATION AND EVALUATION

We implement *BiGroup* on top of our Open RFID Lab (ORL) [2] with the USRP N210 software defined radio (SDR) to read various COTS RFID tags and programmable tags. Figure 10 depicts the testbed. The SDR reader is connected to two USRP RFX900 daughterboards and operates in the 900MHz band. The SDR reader samples physical layer signals at 4MHz. The COTS tags follow the *de facto* EPCglobal C1G2 protocol [1]. The SDR reader interrogates different types of COTS tags (AD-833, ALN-9740 'G', and ALN-9640 Squiggle tags from left to right in Figure 10(b)). We

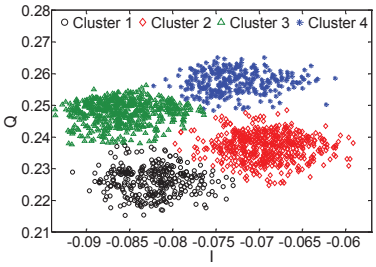


(a) Response starting time offset with respect to Tag 1.

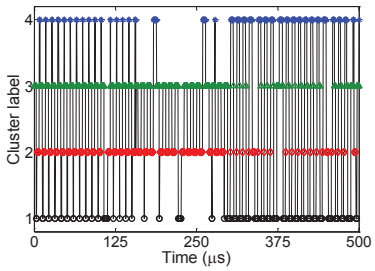


(b) Bit duration offset for each bit with respect to Tag 1.

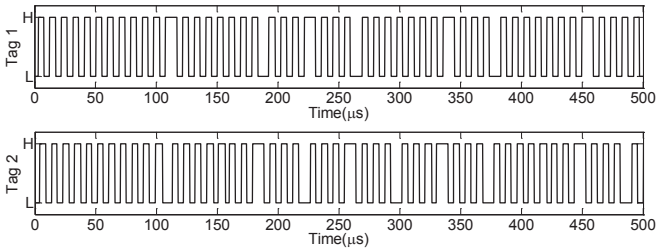
Figure 11: Response starting time and bit duration of 9 COTS tags.



(a) When two tags collide the physical samples exhibit four different clusters.



(b) Received samples transit among the four symbol clusters.



(c) Separated reflection states for the two tags.

Figure 12: Decoding two COTS tags using *BiGroup*.

also test with programmable RFIDs (WISP tags in Figure 10(c)) which implement the same commodity protocol. The reader sends QUERY commands and specifies the frame length f to be 1 by setting the contention parameter Q to 0 ($f = 2^Q$) [1]. Receiving such commands, the tags respond concurrently with RN16 packets, each consisting of a preamble followed by a 16-bit random payload. The RN16 packets are encoded with either the FM0 or Miller encoding scheme as specified by the reader in the QUERY commands. The backscatter link frequency (BLF) is specified as 100kHz.

We evaluate *BiGroup* in comparison with the following RFID concurrent transmission schemes.

- **LA (linear addition) based decoding scheme:** The approach [28] recovers tag collisions assuming that tags' channel coefficients remain static in different collision states. It assumes the linear dependency among cluster centroids to determine collision states and consequently does not perform well in practical scenarios. We compare the performance of *BiGroup* and the LA scheme in decoding the programmable tags.

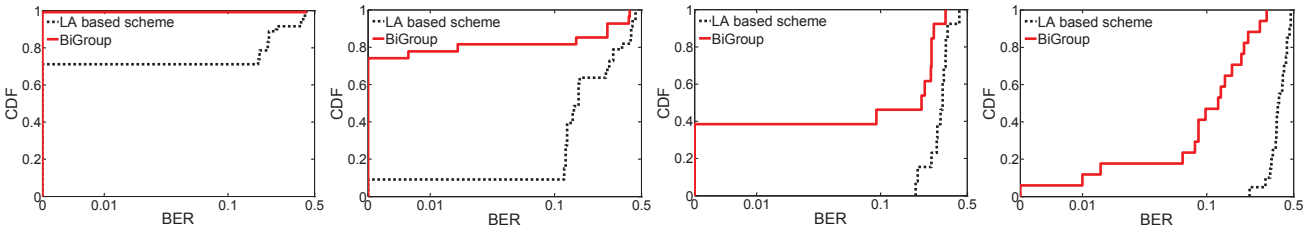
- **Buzz:** Buzz [31] requires tight synchronization among tags so that the bit alignment can be guaranteed for successful decoding. The channel coefficient of each tag is individually measured, assuming that the channel coefficients would linearly combine at the reader during concurrent transmissions. As a result, Buzz cannot decode COTS tags within C1G2 framework. We compare *BiGroup* with Buzz in trace-driven simulations of decoding ideally synchronized tags. The maximum data rate of Buzz is specified the same as *BiGroup*.

- **BST:** In BST [12], the response delay and the bit duration of each tag are pre-assigned by the reader. BST does not conform to C1G2 standard either. We compare *BiGroup* with BST in trace-driven simulations. We tune parameters of BST (e.g., its signal edge detection threshold, "sentinel" bits, etc.) and report its optimum performance.

4.1 Decoding COTS Tags

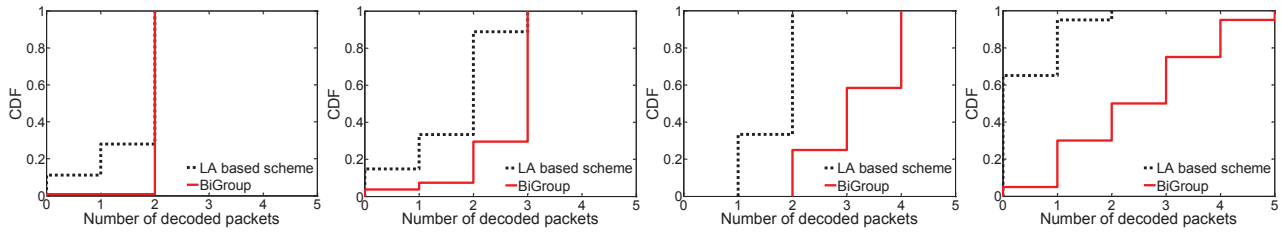
Characterizing COTS tag unsynchronization. We first experiment with the ALN-9640 Squiggle COTS tags. We randomly choose 9 tags of the same batch (labeled as tag $1-9$) and measure their response delays and bit durations. We plot the measured ratio of starting time offset and bit duration offset in Figure 11(a) and Figure 11(b), respectively. We see that different tags have different response delays and bit durations, which result in bit misalignment and non-overlapped state transition boundaries. In particular, the response delay offset and the bit duration offset (normalized by period) can be up to 30% and 1% for each bit, respectively. We also observe similar tag diversities among other tag batches. Although the misalignment due to tag diversities was generally considered harmful in previous schemes [4, 31], *BiGroup* leverages such inherent properties to detect bit boundaries and bootstrap bipartite grouping.

Decoding COTS tags. We experiment with COTS tags and illustrate the process of *BiGroup* decoding. Figure 12 presents an example of decoding 2 colliding tags. *BiGroup* first clusters the received samples on the I-Q plane into 4 clusters labeled in different



(a) Two tags. (b) Three tags. (c) Four tags. (d) Five tags.

Figure 13: CDF for BERs of different schemes with different numbers of colliding tags.



(a) Two tags. (b) Three tags. (c) Four tags. (d) Five tags.

Figure 14: CDF for successfully decoded packets of different schemes with different numbers of colliding tags.

colors as Figure 12(a) depicts. Each cluster represents one collision state. To better understand cluster transitions in time domain, we plot in Figure 12(b) the cluster label of each symbol sample during a time period of 500μs.

By combining the observed time domain state transitions and state flippings at bit boundaries, *BiGroup* performs bipartite grouping. As a result, cluster 3 and 4 represent "H" state of tag 1, and cluster 1 and 2 represent its "L" state, respectively. Similarly, *BiGroup* decodes tag 2 by grouping cluster 2 and 4 which represent state "H", and cluster 1 and 3 which represent state "L", respectively. In Figure 12(c), we see that *BiGroup* separates the individual signals for the two tags from the collision. After the separation, each stream of "H" and "L" states can be decoded to bits by a conventional threshold-based decoder.

In our experiment, *BiGroup* is able to decode all RN16 packets replied from COTS tags to the QUERY command. One RN16 packet includes a predefined 22-bit Miller preamble, followed by a random 16-bit data payload which was instantaneously generated at the tag. While we have no ground truth to assess the decoded random data payloads, our experiment shows that *BiGroup* can correctly recover the fixed 22-bit Miller preambles in most cases.

4.2 Decoding C1G2 Programmable Tags

For more detailed evaluation of *BiGroup* decoding performance, we experiment with programmable passive RFID tags that implement the commodity C1G2 protocol [1]. In particular, we generate random bits offline and load them into WISP tags (Figure 10(c)) as RN16 data payloads that serve as ground truths.

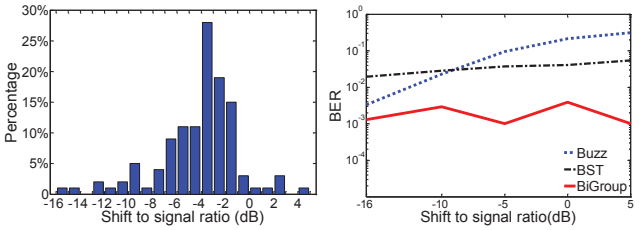
We compare the performance of *BiGroup* and LA based decoding scheme [28] in decoding the WISP tags. We experiment with 2 – 5 tags which respond concurrently to QUERY commands. We repeat the experiment with varied number of tags for 500 times. For each measurement, we vary the channel conditions by manually placing the tags in different locations. The experiment is carried out both during the daytime with people moving around as well as in the relatively stable settings. We evaluate two performance metrics: the BER (Bit Error Rate) and the number of successfully decoded packets. We measure the BER to evaluate the collision recovery capability. We also measure the number of successfully decoded packets in each concurrent transmission to evaluate the goodput.

BER. Figure 13 plots the CDFs of BERs for different numbers of colliding tags. In case of collisions, a recovery scheme may decode and output one or more packets. We compare each output packet with the transmitted packets and record the minimum BER. The BER of undecoded packet is set to 0.5. We measure the average BERs of all packets and report this average.

In Figure 13, we see that *BiGroup* greatly outperforms the LA based decoding scheme. When three tags transmit concurrently, more than 70% of collisions have 0 bit errors in *BiGroup*, while less than 10% have 0 bit errors in the LA based scheme. Around 40% of cases have 0 bit errors in *BiGroup* when four tags transmit concurrently, and around 20% have an average BER below 0.05 in *BiGroup* when five tags transmit concurrently. In contrast, the lowest average BERs when four and five tags collide for the LA based scheme are much higher (>0.1 BERs for all collisions). *BiGroup* achieves much lower BERs compared with the LA based scheme, because *BiGroup* tolerates the variation of channel coefficients and does not assume the linear dependency of signal combinations in practice.

Number of successfully decoding. Figure 14 plots the CDFs of successfully decoded packets of *BiGroup* and the LA based scheme in each collision.

We see that *BiGroup* significantly outperforms the LA based scheme, especially with more colliding tags. When two tags collide, *BiGroup* decodes both for 99% of collisions, while the LA based scheme decodes the two tags in only 70%. When three tags collide, *BiGroup* decodes all three tags in around 71% of cases, and two tags in around 22%, while the LA based scheme decodes the same number of tags in only 11% and 55%, respectively. When four tags collide, *BiGroup* decodes all four tags in around 42% of collisions, three tags in around 33%, and two tags in around 25%, while the LA based scheme is only able to decode at most two tags (in around 62%). When five tags collide, *BiGroup* decodes all five



(a) Shift to signal ratio (SSR) distribution.

(b) BER of different schemes with varied SSR.

Figure 15: SSR and its influence to bit error rates.

		SNR=10dB	SNR=20dB	SNR=30dB
Two tags	Buzz	90.6	98.7	99.0
	BST	84.3	104.8	105.8
	BiGroup	91.8	125.8	128.0
Three tags	Buzz	119.4	143.0	145.0
	BST	92.5	148.6	155.1
	BiGroup	113.8	180.8	192.0
Four tags	Buzz	140.2	178.7	184.2
	BST	103.5	166.0	196.2
	BiGroup	137.6	224.8	256.0
Five tags	Buzz	155.9	203.8	213.3
	BST	125.4	172.5	245.3
	BiGroup	162.4	236.3	320.0

Table 1: Goodput of different schemes (kbps).

tags in around 5% of collisions, four tags in around 20%, and three tags in around 25%, while the LA scheme is only able to decode at most two tags (in less than 5%). For more than 60% cases in the LA based scheme for five tags, no packets can be successfully decoded. Overall, for the concurrent transmissions of more than two tags, *BiGroup* is able to successfully decode 11× more concurrent transmissions than the LA scheme.

4.3 Trace-driven Evaluation for Non-standard Tags

We perform the trace-driven evaluation to compare *BiGroup* with Buzz and BST. The collision decoding performance is mainly influenced by the following factors: the noise level and the level of cluster non-linear dependency influenced by channel coefficients.

To investigate channel coefficient distributions in practice, we first characterise the backscatter channel of multiple tags using the SDR testbed. In particular, the WISP tags are programmed to backscatter known preambles and payloads, so we can directly identify the states of all the tags in each symbol cluster. We measure how the centroids of the collided symbol clusters are shifted away from the linear combinations of individually backscattered symbols. We quantify based on such shift to signal ratio (SSR) the non-linear dependency, which is the ratio of the shift of cluster centroid to the average signal strength. Figure 15(a) plots the distribution of measured SSRs in all experiments. We expect a small SSR (e.g., -16dB) if the non-linear dependency is weak, and a big SSR (e.g., 0dB) if the non-linear dependency is strong. According to our measurements, we see that SSR ranges from -16dB to 5dB, with majority of SSRs (>70%) concentrated in the range from -6dB to -2dB.

Performance comparison. We then let the SDR reader QUERY the programmed tags and record the traces of backscattered signals in 1000 rounds. Following the protocol specifications of Buzz and BST, we synthesize the collected signals of up to five tags and test the performance of different protocols. We take into consideration the non-linear dependency and incorporate the SSR in the tests.

In Figure 15(b), we display BERs of different decoding schemes with varied SSRs for concurrent transmissions of four tags. We fix the SNR to 24dB. The result suggests that the performance of Buzz is highly related to the SSR. The BER of Buzz significantly increases from 10^{-3} to 0.5 when the SSR changes from -16dB to 5dB. In contrast, the BERs of BST and *BiGroup* remain comparatively stable across different SSRs. *BiGroup* consistently outperforms BST by one order of magnitude.

We further compare the decoding schemes with different number of colliding tags and under different SNRs and plot the results in Figure 16. SSR is fixed at -4dB. Comparing *BiGroup* and Buzz, we see that Buzz cannot achieve low BER even with high SNRs (e.g., 25 – 35dB), while the BER of *BiGroup* decreases with higher SNRs and reaches 0 when the SNR is around 25dB. Comparing Figure 16 (a) – (d), we find that the lowest BER of Buzz increases as the number of tags increases, suggesting more severe performance degradation due to stronger non-linear dependency with more tags. We notice that Buzz has a slightly lower BER than *BiGroup* at low SNRs (e.g., 5 – 20dB), where the noise level is too high for *BiGroup*'s clustering algorithm. The BER provided by Buzz in such cases (e.g., 0.1 for four tags), however, cannot support reliable transmissions.

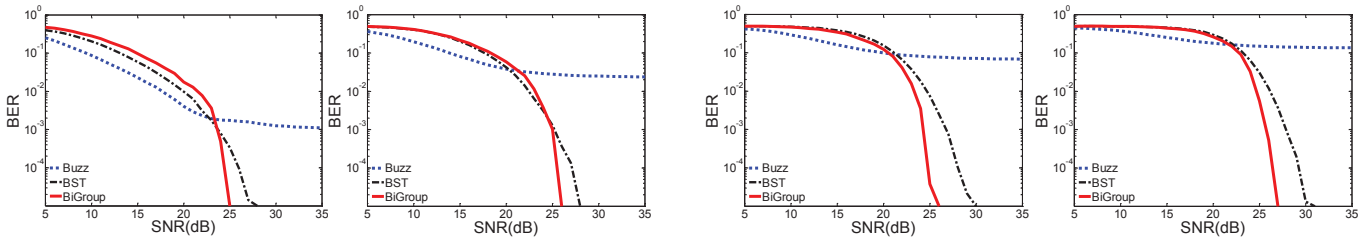
BiGroup also consistently provides better performance compared with BST. A 3 – 4dB SNR gain is achieved by *BiGroup* for low BER situations. The reason is that BST measures the distance between consecutive symbols to detect a signal edge (bit value transition). Due to noises, such signal edges may not be accurately captured. *BiGroup* only requires majority of the bit boundaries to be identified for the purposes of bipartite grouping, while each misdetected signal edge in BST may cause bit error(s). *BiGroup* is inherently more robust to noises.

In Table 1, we present the goodput of different schemes in different channel conditions. We calculate the goodput from the correctly decoded information bits over the total communication time. The data transmission rate of each tag is 64 kbps. We observe that *BiGroup*'s gain over Buzz and BST in goodput is even larger than the gain in BER. Unlike *BiGroup* which requires zero communication overhead, Buzz spends extra time in identification and channel measurement, and BST transmits extra "sentinel" bits and needs to retransmit when signal edges overlap. *BiGroup* provides up to 1.4× and 1.5× goodput gain over BST and Buzz, respectively.

4.4 *BiGroup* for EPC Tag Identification

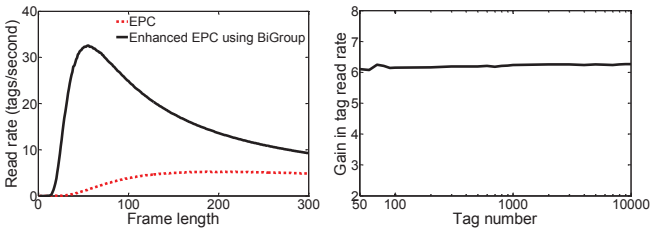
We study how *BiGroup* may benefit the EPC tag identification by decoding RN16 parallel transmissions. We use the SDR reader to QUERY 8 C1G2 passive tags. The frame length is set to 4. In an arbitrary collision slot, *BiGroup* tries to recover at least one RN16 packet out of the collision and then the corresponding tag is ACKed for identification. The read rate is calculated as the average number of identified tags in one second. According to the experimental results, the read rate of the enhanced EPC using *BiGroup* is around 11.6 tags/second, while the read rate of the conventional EPC is 3.7 tags/second. Note that we conduct experiments with the USRP/GnuRadio testbed which has a lower read rate than most commercial readers due to the communication delay between the USRP device and the PC. Nevertheless, we see 3× performance improvement when *BiGroup* is applied in the reader to enhance the EPC procedure. Higher performance gain is expected if *BiGroup* is implemented in hardware of commercial readers.

We now investigate the performance gain with a larger number of tags. In this simulation, the reader is capable of decoding RN16



(a) Two tags. (b) Three tags. (c) Four tags. (d) Five tags.

Figure 16: BER of different decoding schemes across a range of channel conditions.



(a) Read rate for varied frame lengths (200 tags).

(b) Performance gain provided by *BiGroup* for different numbers of tags.

Figure 17: *BiGroup* speeds up EPC tag identification.

collisions of up to five tags and subsequently ACK these colliding tags. We use the time traces measured in our SDR testbed to calculate read rate. We first plot in Figure 17(a) the read rate of EPC identification with and without *BiGroup* for 200 tags (frame length from 0 to 300). We observe that the enhanced EPC using *BiGroup* achieves a maximum read rate of 32.6 tags/second, when frame length is 55. On the other hand, the maximum read rate of conventional EPC is only 5.2 tags/second, when frame length is 200. The performance gain in maximum achievable tag read rate by using *BiGroup* in EPC identification is thus 6.3× for 200 tags. We further plot in Figure 17(b) the gain in tag read rate for varied tag numbers from 50 – 10000. The result demonstrates that consistent 6× to 6.5× gain is achieved by *BiGroup* for a large scale of RFID tags.

5. RELATED WORK

A variety of approaches have been proposed to enable multiple access in RFID communications. Existing commodity RFID systems typically adopt the frame slotted aloha scheme [1] or the tree-based arbitration [17, 29]. Besides the TDMA based approaches, FDMA/SDMA/CDMA based approaches [19, 20, 23, 36] have also been explored to avoid concurrent tag transmissions in the same collision domain, which incurs high coordination overhead.

Recent works improve communication efficiency by supporting concurrent backscatter transmissions. Buzz [31] identifies all tags and decodes tag collisions bit by bit. It assumes the linear combination of reflecting tags' channel coefficients independent of coexisting tags. Buzz also requires the bit-level synchronization among tags as well as channel measurements which are not supported by COTS tags. The linear addition based scheme proposed in [28] also assumes the linear dependency among symbol clusters to map symbol clusters to collision states. Some designs [3, 4, 16] require the knowledge of channel coefficients (e.g. using predefined preambles) and stringent tag synchronization to recover collisions of up to two concurrent tags. The scheme [10] theoretically explores to extract tags with strong signals by correlating with known preambles. A most recent work BST [12] detects signal edges when distances between consecutive symbols exceed a predefined threshold and separates signal edges of multiple tags. BST, however, requires the tags to transmit with assigned initial offsets and bit durations and insert known bits at specific intervals in data packets, all of which are not supported by COTS tags. Some other works assign orthogonal codes to RN16 packets [14, 18] to recover collisions, which are application specific and non-standard. Unlike all these works, *BiGroup* aims to recover collisions without modifying COTS tags and provide general decoding benefits within EPC C1G2 framework. To the best of our knowledge, *BiGroup* is the first effort made to target such a goal.

Some other recent works explore using higher order modulation schemes to improve single tag transmission rate [5, 30], which provides another way of improving RFID communication throughput. Nevertheless, higher order modulations require more complex tag circuits, higher power supply, and are not compatible with existing EPC standards.

It is demonstrated that battery-free devices (similar to COTS tags) can harvest energy and communicate by backscattering ambient RF signals from TV, cellular [21], and WiFi stations [15]. Moreover, backscatter networks can benefit from multi-antenna designs [24], advanced coding mechanisms [24] and full-duplex communications [22]. *BiGroup* is motivated to enable concurrent transmissions for backscatter devices and orthogonal to those works.

6. CONCLUSION

BiGroup enables parallel transmissions and decoding without any extension to COTS RFID tags. To achieve this, *BiGroup* exploits the RFID upper-layer communication patterns and leverages bipartite grouping to substantially improve the performance of physical layer collision recovery. *BiGroup* does not require modifications to C1G2 logics on COTS tags, channel measurements, or stringent time synchronization. We experiment on the software defined testbed and the results show that *BiGroup* significantly improves performance of COTS RFID systems in comparison with other alternative schemes and directly benefits the C1G2 framework. Future work includes further study on scalability of our scheme as well as hardware speedup for better time efficiency.

7. ACKNOWLEDGEMENTS

We would like to thank anonymous shepherd and reviewers for their constructive feedback and suggestions. This work is supported by Singapore MOE AcRF Tier 1 grant MOE2013-T1-002-005, and Nanyang Assistant Professorship (NAP) grant M4080-738.020 of NTU. This work is also supported in part by Hong Kong ECS (PolyU 252053/15E).

References

[1] EPCglobal Inc. EPCglobal Class 1 Generation 2 V. 1.2.0, www.gs1.org/gsmp/kc/epcglobal/uhfc1g2.

[2] Open RFID Lab (ORL), http://pdcc.ntu.edu.sg/wands/ORL/.

[3] C. Angerer, R. Langwieser, and M. Rupp. RFID Reader Receivers for Physical Layer Collision Recovery. *Communications, IEEE Transactions on*, 58(12), 2010.

[4] A. Bletsas, J. Kimionis, A. Dimitriou, and G. Karystinos. Single-Antenna Coherent Detection of Collided FM0 RFID Signals. *Communications, IEEE Transactions on*, 60(3), 2012.

[5] C. Boyer and S. Roy. Coded QAM Backscatter Modulation for RFID. *Communications, IEEE Transactions on*, 60(7), 2012.

[6] M. Buettner, B. Greenstein, and D. Wetherall. Dewdrop: An Energy-aware Runtime for Computational RFID. In *USENIX NSDI*, 2011.

[7] M. Buettner, R. Prasad, M. Philipose, and D. Wetherall. Recognizing Daily Activities with RFID-based Sensors. In *ACM UbiComp*, 2009.

[8] Q. Dong, A. Shukla, V. Shrivastava, D. Agrawal, S. Banerjee, and K. Kar. Load Balancing in Large-Scale RFID Systems. In *IEEE INFOCOM*, 2007.

[9] M. Ester, H. peter Kriegel, J. S, and X. Xu. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. AAAI Press, 1996.

[10] K. Fyhn, R. Jacobsen, P. Popovski, A. Scaglione, and T. Larsen. Multipacket Reception of Passive UHF RFID Tags: A Communication Theoretic Approach. *Signal Processing, IEEE Transactions on*, 59(9), 2011.

[11] A. Gudipati and S. Katti. Strider: Automatic Rate Adaptation and Collision Handling. In *ACM SIGCOMM*, 2011.

[12] P. Hu, P. Zhang, and D. Ganesan. Leveraging Interleaved Signal Edges for Concurrent Backscatter. In *ACM HotWireless*, 2014.

[13] S. Jana and S. K. Kasera. On Fast and Accurate Detection of Unauthorized Wireless Access Points Using Clock Skews. In *ACM MobiCom*, 2008.

[14] L. Kang, K. Wu, J. Zhang, H. Tan, and L. Ni. DDC: A Novel Scheme to Directly Decode the Collisions in UHF RFID Systems. *Parallel and Distributed Systems, IEEE Transactions on*, 23(2), 2012.

[15] B. Kellogg, A. Parks, S. Gollakota, J. R. Smith, and D. Wetherall. Wi-fi Backscatter: Internet Connectivity for RF-powered Devices. In *ACM SIGCOMM*, 2014.

[16] R. Khasgiwale, R. Adyanthaya, and D. Engels. Extracting Information From Tag Collisions. In *IEEE RFID*, 2009.

[17] S. H. Kim and P. Park. An Efficient Tree-Based Tag Anti-Collision Protocol for RFID Systems. *Communications Letters, IEEE*, 11(5), 2007.

[18] L. Kong, L. He, Y. Gu, M.-Y. Wu, and T. He. A Parallel Identification Protocol for RFID systems. In *IEEE INFOCOM*, 2014.

[19] T. Li, M. K. Han, A. Bhartia, L. Qiu, E. Rozner, Y. Zhang, and B. Zarikoff. CRMA: Collision-resistant Multiple Access. In *ACM MobiCom*, 2011.

[20] H.-C. Liu and J.-P. Ciou. Performance Analysis of Multi-Carrier RFID Systems. In *SPECTS*, 2009.

[21] V. Liu, A. Parks, V. Talla, S. Gollakota, D. Wetherall, and J. R. Smith. Ambient Backscatter: Wireless Communication out of Thin Air. In *ACM SIGCOMM*, 2013.

[22] V. Liu, V. Talla, and S. Gollakota. Enabling Instantaneous Feedback with Full-duplex Backscatter. In *ACM MobiCom*, 2014.

[23] C. Mutti and C. Floerkemeier. CDMA-based RFID Systems in Dense Scenarios: Concepts and Challenges. In *IEEE RFID*, 2008.

[24] A. N. Parks, A. Liu, S. Gollakota, and J. R. Smith. Turbocharging Ambient Backscatter Communication. In *ACM SIGCOMM*, 2014.

[25] S. Rallapalli, L. Qiu, Y. Zhang, and Y.-C. Chen. Exploiting Temporal Stability and Low-rank Structure for Localization in Mobile Networks. In *ACM MobiCom*, 2010.

[26] L. Shangguan, Z. Li, Z. Yang, M. Li, and Y. Liu. OTrack: Order Tracking for Luggage in Mobile RFID Systems. In *IEEE INFOCOM*, 2013.

[27] L. Shangguan, Z. Yang, A. X. Liu, Z. Zhou, and Y. Liu. Relative Localization of RFID Tags Using Spatial-temporal Phase Profiling. In *USENIX NSDI*, 2015.

[28] D. Shen, G. Woo, D. Reed, A. Lippman, and J. Wang. Separation of Multiple Passive RFID Signals Using Software Defined Radio. In *IEEE RFID*, 2009.

[29] D.-H. Shih, P.-L. Sun, D. C. Yen, and S.-M. Huang. Taxonomy and Survey of RFID Anti-collision Protocols. *Computer Communications*, 29(11), 2006.

[30] S. Thomas and M. Reynolds. A 96 Mbit/sec, 15.5 pJ/bit 16-QAM Modulator for UHF Backscatter Communication. In *IEEE RFID*, 2012.

[31] J. Wang, H. Hassanieh, D. Katabi, and P. Indyk. Efficient and Reliable Low-power Backscatter Networks. In *ACM SIGCOMM*, 2012.

[32] J. Wang and D. Katabi. Dude, Where's My Card?: RFID Positioning That Works with Multipath and Non-line of Sight. In *ACM SIGCOMM*, 2013.

[33] J. Wang, D. Vasisht, and D. Katabi. RF-IDraw: Virtual Touch Screen in the Air Using RF Signals. In *ACM SIGCOMM*, 2014.

[34] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu. E-eyes: Device-free Location-oriented Activity Identification Using Fine-grained WiFi Signatures. In *ACM MobiCom*, 2014.

[35] L. Yang, Y. Chen, X.-Y. Li, C. Xiao, M. Li, and Y. Liu. Tagoram: Real-time Tracking of Mobile RFID Tags to High Precision Using COTS Devices. In *ACM MobiCom*, 2014.

[36] J. Yu, K. Liu, and G. Yan. A Novel RFID Anti-Collision Algorithm Based on SDMA. In *IEEE WiCOM*, 2008.

[37] D. Zanetti, B. Danev, and S. Capkun. Physical-layer Identification of UHF RFID Tags. In *ACM MobiCom*, 2010.

[38] P. Zhang, P. Hu, V. Pasikanti, and D. Ganesan. EkhoNet: High Speed Ultra Low-power Backscatter for Next Generation Sensors. In *ACM MobiCom*, 2014.

[39] X. Zhang and K. G. Shin. E-MiLi: Energy-minimizing Idle Listening in Wireless Networks. In *ACM MobiCom*, 2011.

[40] Y. Zheng and M. Li. PET: Probabilistic Estimating Tree for Large-Scale RFID Estimation. *IEEE Transactions on Mobile Computing*, 11(11), 2012.

[41] Y. Zheng and M. Li. Fast Tag Searching Protocol for Large-scale RFID Systems. *IEEE/ACM Trans. Netw.*, 21(3), 2013.

[42] X. Zhou, Z. Zhang, Y. Zhu, Y. Li, S. Kumar, A. Vahdat, B. Y. Zhao, and H. Zheng. Mirror Mirror on the Ceiling: Flexible Wireless Links for Data Centers. In *ACM SIGCOMM*, 2012.