# We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

29

# Efficient Outlier Detection in RFID Trails

Elio Masciari and Giuseppe M. Mazzeo
*ICAR-CNR*
*Italy*

## 1. Introduction

Radio Frequency Identification (RFID) applications are emerging as key components in object tracking and supply chain management systems. In the next future almost every major retailer will use RFID systems to track the shipment of products from suppliers to warehouses. In addition to providing insight into shipment and other supply chain process efficiencies, such data can be also valuable for determining product seasonality and other trends resulting in key information for the companies plans. Moreover, companies are already exploring more advanced uses for RFID. For example, tire manufacturers plan to embed RFID chips in tires to determine the tire deterioration. Many pharmaceutical companies are embedding RFID chips in drug containers to better track and avert the theft of highly controlled drugs. Airlines are considering RFID-enabling key onboard parts and supplies to optimize aircraft maintenance and airport gate preparation turnaround time.

Due to the streaming nature of RFID readings, large amounts of data are generated by these devices. In particular, RFID applications will generate a lot of so-called “thin” data, i.e. data pertaining to time and location. This phenomenon will be even more relevant when RFIDs are so cheap that every individual item can be tagged thus leaving a “trail” of data as it moves across different locations.

This scenario raises new challenges in effectively and efficiently exploiting such large amounts of data. In this paper we define a technique for detecting anomalous data in order to prevent problems related to inefficient shipments or fraudulent actions. To this end, we introduce a framework enabling users to control correct shipment of items. As an anomalous situation is detected, the system signals it in order to quickly recover the possible error. Moreover, while the trajectory is monitored we provide a validation step in order to take into account the natural “concept drift”, i.e. deviation from the shipping plan due to modification of the service. In such a case, the signalled trajectory becomes new knowledge about the overall system. In order to check whether a trajectory has to be considered anomalous with respect to past ones, we adopt a distance-based approach. Specifically, we measure similarity between item routes by comparing their Discrete Fourier Transforms. We point out that the proposed approach is flexible and can be applied in different scenarios in order to monitor very different systems independently of their spatial extension. Preliminary experiments show the effectiveness of our approach.

This chapter is organized as follows. In Section 2 the problem and its application context are described. Section 3 introduces a measure of similarity between RFID series based on the Discrete Fourier Transform, which will be used to detect anomalies in object shipping as

Source: Development and Implementation of RFID Technology, Book edited by: Cristina TURCU,
ISBN 978-3-902613-54-7, pp. 554, February 2009, I-Tech, Vienna, Austria

described in Section 4. In Section 5 a technique for managing the concept drift related to the shipping plans is introduced. Section 6 presents some experimental results, showing the effectiveness of the adopted measure of similarity. In Section 7, some related work is presented. Finally, in Section 8 conclusion are drawn and some possible improvements to be developed in future works are proposed.

## 2. Problem statement

An RFID system consists of a set of *n* sources (i.e., tag readers), located at different positions, $R=\{r_1,\ldots, r_n\}$, producing *n* independent streams of data, representing tag readings. Each data stream can be viewed as a sequence of triplets ‹*rid, epc, ts*›, where: (i) $rid \in \{1,\ldots,n\}$ is the tag reader identifier (observe that it implicitly carries information about the spatial location of the reader), (ii) *epc* is the electronic product code, and (iii) *ts* is a *timestamp*. Basically, a reading ‹*rid, epc, ts*› denotes the fact that the item *epc* was at the location of the reader *tid* at time *ts*. The data streams produced by the sources are caught by a *RFID Data Stream Management System* (RfidDSMS), which combines the RFID readings into a unique data stream in order to support data analysis. We assume that *epcs* are gouped by classes and each class of *epcs* is assigned a shipping plan, which can be time-varying. As objects move across the distribution network, they are traced using the spatio-temporal information generated by RFID readers. The system architecture is shown in figure 1.

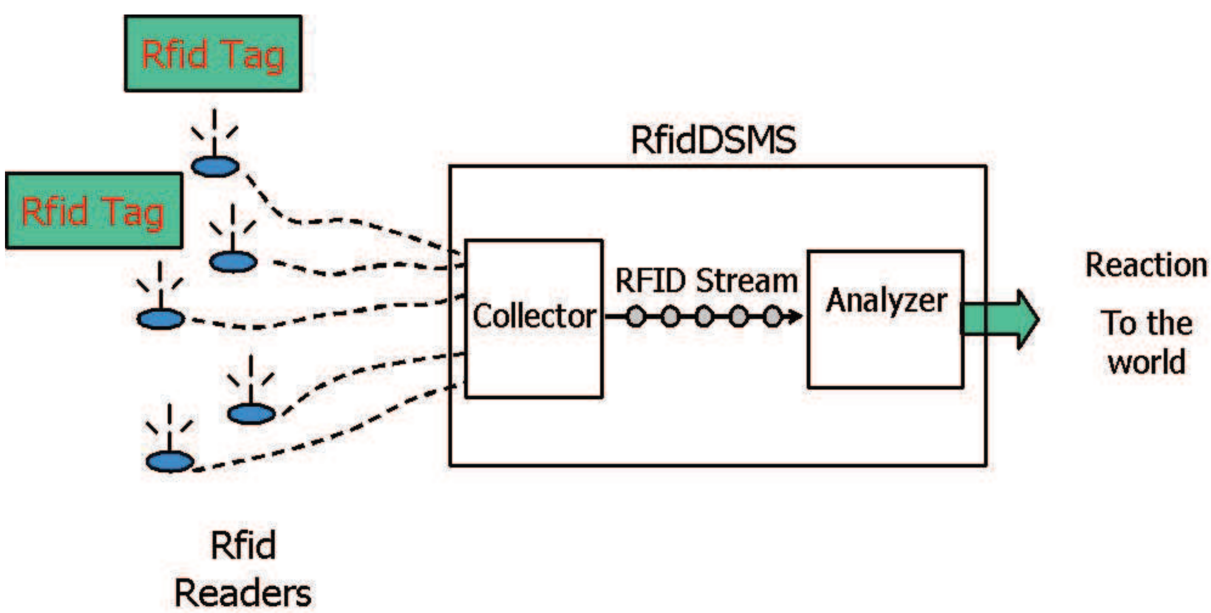



Fig. 1. RfidDSMS architecture

Our goal is to identify anomalies in the flow of the data. A possible anomaly can be an object that is planned to pass through a series of checkpoints but some check is missing, so it could be the case that someone modified the path with fraudulent intentions. Another relevant anomaly that would be interesting to be revealed is the continuous stay of an object at the same place, since it could be the case that the object is damaged or some shipment problem occurred. Thus, we can state the problem of detecting outliers in RFID data as follows.

*Given an RFID system as described above, how is it possible to efficiently detect whether a new reading ‹rid, epc, ts› is representative of an anomaly in the shipping of epc?*

In order to show how we tackle this problem, we introduce a running example to which we will refer through the rest of the paper. Consider the following sequence of readings being collected by the *RfidDSMS* assuming, for the sake of simplicity, that the system consists of *5* readers, $r_1, r_2, r_3, r_4, r_5$, *3* tagged items, $o_1, o_2, o_3$ ($o_i$ represents the *epc* of the *i*-th item) and the initial time is 0. A possible sequence of readings could be:

$Seq = \{\langle r_1,o_1,0\rangle, \langle r_2,o_1,1\rangle, \langle r_1,o_2,1\rangle, \langle r_3,o_1,2\rangle, \langle r_2,o_2,2\rangle, \langle r_1,o_3,2\rangle, \langle r_4,o_1,3\rangle, \langle r_4,o_2,3\rangle, \langle r_2,o_3,3\rangle, \langle r_5,o_1,4\rangle, \langle r_5,o_2,4\rangle,\langle r_2,o_3,4\rangle, \langle r_2,o_3,5\rangle\}$.

In order to quickly identify the kind of anomalies described above, we need to define a technique for continuously comparing the streams generated by the readers with the original shipment plans. In order to accomplish this task we need to define a measure of similarity between two sequences. Intuitively, two sequences are said to have a similar structure if they correspond in the type of the readings they contain and in the order the readings appear. Observe that, even if it is easy to detect whether the structure of two streams is almost the same (i.e., the item has been scanned by the same set of readers), this information is rarely useful for our aims. Indeed, we would like to quantify the similarity between the structures of two streams, also emphasizing the differences that are most relevant. For instance, we consider similar two streams that exhibit the same features with different regularities, since this could be simply due to a shipment delay.
Much attention has been devoted to the problem of detecting similarity between time series using approaches such as time warping (Kruskall & Liberman, 1999). In this paper we propose a different approach, which is essentially based on the idea of associating each stream of readings related to an item with a time series representing its structure, and checking the structural similarity on the basis the Discrete Fourier Transform of its time series. As we will see, this approach is both efficient and effective.

## 3. A DFT-based similarity measure for RFID data

In this section we introduce a technique for encoding the input data stream into a time series and define a suitable similarity function that will be used to identify outlier data. First, we will describe how to encode RFID reading into time series representing the route of each item being monitored, and then we will introduce the metric for comparing the similarity of time series which will be adopted to identify outliers.
Given a sequence *Seq* of RFID readings, it is possible to obtain the route of a item $x$, that will be denoted as *Route(Seq, x)*, by extracting from *Seq* all the triplets with the *epc* value equal to that of $x$, and ordering these triplets by their ascending timestamp.
This operation yields an ordered set of triplets, from which the *epc* value can be disregarded (as it is equal to $x$ for all the triplets), thus obtaining a sequence of pairs representing the association time-location of the item $x$ that has been monitored.
These pairs can be represented as an array, where the cell indexes represent the timestamps, and the cell values represent the corresponding locations.
Observe that, for some timestamps the readings related to some items could not exist. In this case, we assume that the corresponding location for these timestamps is a *null* value.
From the sequence introduced in the previous section, we obtain the following routes

$Route(Seq, o_1)=[r_1, r_2, r_3, r_4, r_5, null]$
$Route(Seq, o_2)=[null, r_1, r_2, r_4, r_5, null]$
$Route(Seq, o_3)=[null, null, r_1, r_2, r_2, r_2]$

These arrays can be transformed into arrays of integers by assigning a positive integer identifier to each reader and assuming the *null* value to correspond to *0*.
For instance, by associating $r_i$ with *i*, the routes of our running examples are those reported below:

*Route(Seq,* $o_1$*)=[1, 2, 3, 4, 5, 0]*
*Route(Seq,* $o_2$*)=[0, 1, 2, 4, 5, 0]*
*Route(Seq,* $o_3$*)=[0, 0, 1, 2, 2, 2]*

We call this encoding scheme of the readers *simple encoding*. We also introduce a more sophisticated encoding scheme, called *pairwise encoding*. This encoding scheme works as follows. Instead of assigning a unique identifier to each reader, as in the simple encoding, we assign a positive integer location to each different pair of readers. Thus, a function $E{:}R \cup \{null\} \times R \cup \{null\} \rightarrow N$ will represent the encoding function. For instance, in our running example, the encoding function *E* could be defined as in Fig.1.

| | *null* | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|---|---|
| *null* | *0* | *1* | *2* | *3* | *4* | *5* |
| $r_1$ | *6* | *7* | *8* | *9* | *10* | *11* |
| $r_2$ | *12* | *13* | *14* | *15* | *16* | *17* |
| $r_3$ | *18* | *19* | *20* | *21* | *22* | *23* |
| $r_4$ | *24* | *25* | *26* | *27* | *28* | *29* |
| $r_5$ | *30* | *31* | *32* | *33* | *34* | *35* |

Fig. 1. Example of function adopted for the pairwise encoding scheme.

On the basis of the function *E* a vector which encodes the route $[p_1, p_2, \ldots, p_k]$, where $p_i \in R \cup \{null\}$, of an object *o*, is obtained by assigning $E(null, p_1)$ to the first element of *Route(Seq, o)*, and $E(p_{i-1}, p_i)$ to the *i-th* element ($1<i<k$) of *Route(Seq, o)*.
Thus, for our running example, the encoded routes are the following.

*Route(Seq,* $o_1$*)= [1, 8, 15, 22, 29, 30]*
*Route(Seq,* $o_2$*)= [0, 1, 8, 16, 29, 30]*
*Route(Seq,* $o_3$*)= [0, 0, 1, 8, 14, 14]*

In the following, we will omit *Seq* from *Route(Seq, x)*, thus writing *Route(x)*, when *Seq* can be implicitly assumed. Unfortunately, comparing two time series obtained by this encoding scheme can be very difficult, since time series could have different lengths and thus costly resizing and alignment operations, and stretching (or narrowing) would be necessary.
These drawbacks can be avoided if the signals are compared by examining their *Discrete Fourier Transforms* (Oppenheim & Shafer, 1999), which reveals much about the distribution and relevance of signal frequencies and can be computed incrementally as new readings arrive avoiding the problem of recomputing at each step the signal for each item.
Given a sequence of readings *Seq* and an item *x*, we denote as *DFT(Seq, x)* the Discrete Fourier Transform of the time series *Route(Seq, x)*. More formally, being $r_x$*=Route(Seq, x)* the array of size *N* representing the time series of item *x* extracted from the reading sequence *Seq*, *DFT(Seq, x)* is the array $R_x$ of size *N* such that

$$R_x[k] = \sum_{t=0}^{N-1} r_x[t] \cdot e^{-\frac{2\pi i}{N} k \cdot t} \quad (k \in \{0, \ldots, N-1\})$$

In order to compare two routes, we consider the difference in the magnitude of the corresponding frequency components in their DFTs. This allows (i) to abstract from the length of the streams, and (ii) to know whether a given subsequence (representing for example a set of repeated readings) exhibits a certain regularity, no matter where it is located along the signal. Specifically, being *x, y* two items whose readings are in the stream *Seq*, $r_x$, $r_y$ their routes and $R_x$, $R_y$ the corresponding DFTs, we define the *DFT distance* ($dist_{DFT}(Seq, x, y)$) of the items as the sum of the squared difference of the magnitudes of the two signals, that is

$$dist_{DFT}(Seq, x, y) = \sum_{k=0}^{N-1} (|R_x[k]| - |R_y[k]|)^2$$

In the following, we will omit *Seq* from *DFT(Seq, x)* and $dist_{DFT}(Seq, x, y)$, thus writing *DFT(x)* and $dist_{DFT}(x, y)$, when *Seq* can be implicitly assumed. We will also adopt the notation $\mathrm{dist}_{\mathrm{DFT}}(\mathrm{p, q})$ and $dist_{DFD}(P, Q)$ to represent the distance between two time series *p* and *q* whose DFTs are *P* and *Q*, respectively.

## 4. Outlier identification

In this section we define a strategy for identifying anomalies in the data flow, on the basis of the similarity measure for RFID time series defined in the previous section. In order to better understand this problem we describe two possible scenarios in our running example.
Consider three containers, whose epcs are $o_1$, $o_2$ and $o_3$, containing dangerous material that must be delivered through check points $r_1$, $r_2$, $r_3$, $r_4$, $r_5$ (in the given order). Assume that the RFID system reveals their routes to be *Route(Seq,* $o_1$*)*, *Route(Seq,* $o_2$*)*, and *Route(Seq,* $o_3$*)* reported in the previous section.
It easy to see that $o_1$ followed the correct route whereas $o_2$ and $o_3$ routes are affected by two different irregularities. Specifically, $o_2$ did not pass through $r_3$ and $o_3$ stayed too long at $r_2$. As regards $o_2$, two main explanations could be provided: (i) the original routing has been changed for shipment improvement, or (ii) someone changed the route with fraudulent intentions (e.g., in order to robber the content or to modify it). In our encoding strategy this case will produce two different signals exhibiting low similarity since the structure is different. As regards $o_3$, its sequence may occur either because (i) the container (or the content) is damaged so it cannon be shipped form $r_2$ until some recovery operation is performed or (ii) the shipment has been delayed. Depending on the anomaly detected, different recovery actions need to be performed. In our encoding strategy this situation will be reported in the frequency spectrum as a few components with amplitude much more larger than the others.
The two situations described above have an intuitive explanation but we point out that our encoding strategy enables detecting all the anomalies that cause the time series representing the routes to be different. Based on the above examples, we can now define our notion of outlier. Given the route $r_x$=*Route(x)* of an object *x* being traced whose planned sequence is represented by a time series $r^*$, and a threshold value *T*, we say that $r_x$ is an outlier route if $dist_{DFT}(r_x,r^*)>T$. The threshold value can be chosen on the basis of the stream being monitored. Once defined our notion of outlier we can design an effective method for tracking objects. In particular, as objects enter our RFID environment, the readings are

collected by the *RfidDSMS*. At the *RfidDSMS* the signal corresponding to the correct shipping plan is stored ($r^*$) along with its *DFT* ($R^*$). As readings are generated, the *DFT* of the signal corresponding to the actual plan of the item $x$ is computed ($R_x$) by using the previously described encoding strategy. If $dist_{DFT}(R^*,R_x)$ is larger than a predefined threshold $T$ (suitable for the context being analyzed) a *signalling service* will notify the detected outlier in order to allow a proper recovery action.
In more details, as new readings arrives their contributions to the *DFT* of the proper *epc* is computed. This task can be accomplished very efficiently since the computation of the DFT can be performed incrementally without recomputing it from scratch.
If the computed distance between the original signal and the actual one is larger than $T$ a *signalling service* will notify the detected anomaly, the location and the *epc* to be checked.

## 5. Managing changes in shipping plans

In the previous section it was assumed that the correct shipping plan for a given object is defined. Even though it is a realistic assumption, it is quite common that shipping plans change throughout time, for instance because the shipping has been optimized or because new requirements impose objects to pass through different check points. In order to manage this situation, it is necessary to enable the system to follow these changes. To this end, we propose a semi-automatic technique, in which the (i) system signals potential anomalies which can depend either on a shipping problem or on a planned shipping change and (ii) a human user check which of the two cases occurred taking the proper recovery action.
We assume that our system stores the DFT of the routes of the object currently in shipping. Furthermore, for each class of *epcs*, representing objects are shipped together, and for each location $r_i$, the *average routing* $R^*(epc\_class, r_i)$ is stored. Basically, $R^*(epc\_class, r_i)$ represents a weighted average of the partial DFTs of all the correct past routes followed by the objects belonging to the class *epc_class* arriving at $r_i$. These values are adopted for checking the validity of new routes, and they are continuously updated. Fig. 2 shows an algorithm which signals possible anomalies and maintains the average shipping plans up-to-date. The algorithm is invoked when a new triple ‹*l, o, t*› arrives (we recall that ‹*l, o, t*› denotes the fact that object $o$ is at location $l$ at time $t$) and works as follows. First, the DFT $R_o$ of the route of object $o$ is updated. The update requires time $O(N)$, where $N$ is the number of locations traversed by the object. Then, the class $e$ of epcs to which $o$ belongs is retrieved. If a hash table is adopted to map the classes of epcs, this operation can be performed in constant time. Then, the DFT-based distance between $R_o$ and the average DFT $R^*(e, l)$ of past correct routes

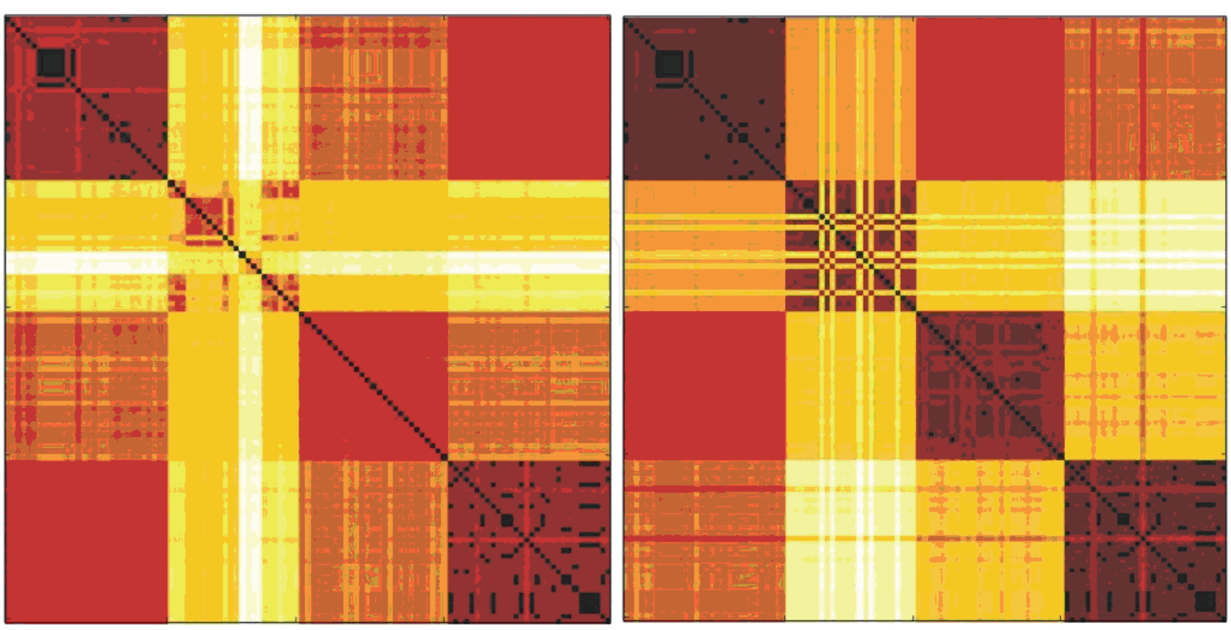
```
INPUT: a triplet ‹l, o, t›;
begin
       update_R(o, l, t);
       e ← epc_class(o);
       if dist(Rₒ, R*(e,l)) > T
              signal(‹l, o, t›);
       else
              R*(e, l) ← α * Rₒ + (1 - α) * R*(e, l);
end
```

Fig. 2. Algorithm for detecting wrong routes

is computed. If this distance is less than a given threshold $T$, $R^*(e, l)$ is updated, in order to take into account the last correct route. The update is performed by adopting a parameter $\alpha$ for weighting new valid routes. The computation of the distance, and the update of the average route can be performed in time O($N$). If the distance is larger than the threshold T, the triplet is signalled as a possible anomaly. In this case, a human action is required. If the user considers valid the signalled route, then she will update $R^*(e, l)$ taking into account last route. The algorithm works in time $O(N)$.

## 6. Experimental evaluation

In this section, we present some experiments we performed to assess the effectiveness of the proposed approach in detecting outliers in RFID data streams. The direct result of each test is a similarity matrix representing the degree of similarity for each pair of streams. We analyzed about *104* streams, belonging to *4* classes of epcs: 1) `Tuna Fish`, whose readings are generated by *26* tagged containers storing *500* cans each, 2) `Tomato`, whose readings are generated by *23* tagged containers storing *400* cans each, 3) `Syrupy Peach`, whose readings are generated by *20* tagged containers storing *350* cans each, and 4) `Meat`, whose readings are generated by *35* tagged containers storing *600* cans each. For each pair of objects, the DFT-distance $d$ between their routes is computed, and the similarity between the routes of the pair is assumed to be equal to $1/(d+1)$. We adopted both encoding schemes, simple and pairwise, thus obtaining two different *confusion matrices* depicted in Fig. 3. The confusion matrices are represented by images, where the colour of the pixel $(i,j)$ represents the similarity between the routes of the $i$-th and $j$-th objects. More specifically, the higher the similarity, the darker the pixel colour. Note that the blocks on the diagonal of the matrix correspond to intra-class similarities, whereas the blocks outside the diagonal represent the inter-class similarities. A quantitative analysis of the results, averaged by classes of epcs, is reported in Tables 1 and 2.



(a) (b)

Fig. 3. Confusion matrix obtained using simple encoding (a) and pairwise encoding (b)

| | Tuna Fish | Tomato | Syrupy Peach | Meat |
|---|---|---|---|---|
| Tuna Fish | 0.9608 | 0.6039 | 0.7568 | 0.8094 |
| Tomato | 0.6039 | 0.8553 | 0.6545 | 0.6039 |
| Syrupy Peach | 0.7568 | 0.6545 | 0.8095 | 0.7265 |
| Meat | 0.8094 | 0.6039 | 0.7265 | 0.9436 |

Table 1. Similarity between routes of 4 different classes of items using simple encoding

| | Tuna Fish | Tomato | Syrupy Peach | Meat |
|---|---|---|---|---|
| Tuna Fish | 0.9824 | 0.7542 | 0.8043 | 0.7584 |
| Tomato | 0.7542 | 0.9225 | 0.6784 | 0.5264 |
| Syrupy Peach | 0.8043 | 0.6784 | 0.9545 | 0.6834 |
| Meat | 0.7584 | 0.5264 | 0.6834 | 0.9665 |

Table 2. Similarity between routes of 4 different classes of items using pairwise encoding

The quantitative results obtained reveal that our measure of similarity is effective. In fact, for all classes, the intra-class similarity values are sufficiently higher than the inter-class ones, allowing for separating all classes from one another and thus stating that the containers follows the predicted plan. Furthermore, the pairwise encoding provides better results than the simple one, as in the majority of cases, intra-similarity is higher and inter-similarity is lower. This is rather expected, as a new reading updates the DFT of current routes by taking into account not only the location where the reading was taken, but also the provenance of the object.

## 7. Related work

Sensor networks has emerged in recent year as a fruitful research field since it provide interesting challenges both for designing more efficient system and for data management. We disregard in our discussion the physical aspect and we shall focus on data management issues. Bonnet et al. (2001) describe the design guidelines for sensor database systems and recognize numerous advantages of the distributed approach over the warehousing approach, under the condition of having sensors with computational capability. In particular, they study the *in-network aggregation* and distributed query processing. The Cougar project (Yao & Gehrke, 2002) is focused on techniques for processing queries over sensor data. In this project there is a distributed sensor network environment and a central administration for sensors management. The *TinyDB*, proposed by Madden et al. (2002), is a distributed query processor that runs on each of the sensor nodes; this system is focused on optimizing data acquisition for long-running queries, being no data stored locally at the nodes. Yao and Gehrke (2003) extend the general guidelines for sensor databases proposed by Bonnet et al. (2001) focusing on the query processing issue, and propose the definition of a query layer that (i) improves the capabilities of a generic sensor network, and (ii) defines a declarative language for efficiently in-network query processing. Schlesinger and Lehner (2004) combine the Grid framework with sensor database systems. In their framework, each Grid node holds cached data of other Grid nodes using a *data replication scheme*. Furthermore, they propose a model describing inconsistencies in Grid organized sensor database systems, and a technique for finding optimal distributed query plans by deciding on which node of the grid a query should be evaluated. Concerning the specific case of RFID data management, it has been studied in (Gonzales et al., 2006). In particular the problem of defining an efficient warehousing model along with techniques for summarizing and indexing data are discussed. They introduce a model based on hierarchy of summary called *RFID-Cuboids* of the RFID data aggregated at different abstraction levels in order to allow different kinds of data analysis. Regarding time series comparison, a traditional approach known in literature is based on time warping (Yi et al., 1998), which mainly consists in considering every possible stretching and narrowing of the signals being compared, and choosing the best matching. However, time-warping-based approaches are quite expensive (quadratic in complexity) when dealing with long series. Furthermore, time warping cannot be incrementally computed as objects move throughout locations, differently from DFT, which can be efficiently updated.

## 8. Conclusions

In this chapter we addressed the problem of detecting outliers in RFID readings stream. The technique we have proposed is mainly based on the idea of representing an stream of readings as a time series. Thereby, the structural similarity between two series can be computed by exploiting the Discrete Fourier Transform (DFT) of the associated signals. Experimental results showed the effectiveness of our approach, with particular regard to some of the encoding schemes defined in the paper.

The current work is subject to further significant extensions. As a matter of fact, the structural similarity between routes can be refined exploiting additional information on the RFID stream such as the actual distance among the reader. In our current implementation, the reader encoding function does not take into account semantic similarities between tags being scanned (i.e., objects belonging to the same category). However, precision could be improved by exploiting tagged object similarity techniques, such as, e.g., the exploitation of suitable ontologies. FFT-based distance measures, different from the one introduced in the paper could be used. Indeed, the FFT transformation contains lots of information about the contents of the original stream, and different distance measures could be more appropriate to exhibit such information. A further possibility to improve the proposed encoding schemes is that of defining a different strategy for dealing with the stream of readings in particular we plan to use more robust methods (e.g., non parametric-ones) for determining outliers, like those introduced in (Subramaniam et al., 2006). However, eventually new implementation has to be carefully studied in order to avoid inefficiency that are common in a system that has to deal with huge amounts of data.

## 9. References

Bonnet, P.; Gehrke, J. & Seshadri, P. (2001). Towards Sensor Database Systems, *Proceedings of Second International Conference on Mobile Data Management (MDM 2001)*, ISBN: 978-3-540-41454-4, pp. 3-14, Hong Kong, China, January 2001, Springer-Verlag, London, UK

Gonzales, H.; Han, J.; Li, X. & Klabjan, D. (2006). Warehousing and Analyzing Massive RFID Data Sets, *Proceedings of the 22nd International Conference on Data Engineering (ICDE 2006)*, ISBN: 0-7695-2570-9, pp. 83, Atlanta, GA, USA, April 2006, IEEE Computer Society Washington, DC, USA

Kruskall, J., B. & Liberman, M. (1999). The Symmetric Time Warping Problem: From Continuous to Discrete, In: *Time Warps, String Edits and Macromolecules: The Theory* *and Practice of Sequence Comparison*, Kruskal, J.,B. & Sankoff, D. editors; pp. 125–161, Stanford: CSLI Publications, ISBN: 1575862174, Stanford

Madden, S., R.; Franklin, M. J.; Hellerstein, J., M. & Hong, W. (2003). TinyDB: An Acquisitional Query Processing System for Sensor Networks. *ACM Transactions on Database Systems (TODS)*, vol. 30, no. 1 (March 2005), pp. 122-173, ISSN: 0362-5915

Oppenheim, A., V. & Shafer R., W. (1999). *Discrete-Time Signal Processing,* Prentice Hall, ISBN: 978-0137549207

Schlesinger, L. & Lehner, W. (2004). Querying Asynchronously Updated Sensor Data Sets under Quantified Constraints, In *GeoSensor Networks*, vol. 1, part 2, Stefanidis, A. & Nittel, S. editors., pp. 13-30, CRC, ISBN: 978-0415324045

Subramaniam, S.; Palpanas, T.; Papadopoulos, D.; Kalogeraki, V. & Gunopulos, D. (2006). Online Outlier Detection in Sensor Data Using Non-parametric Models, *Proceedings of the 32nd international conference on Very Large Data Bases (VLDB 2006)*, pp. 187-198, Seoul, Korea, September 2006, VLDB Endowment

Yao, Y. & Gehrke, J. (2002). The Cougar Approach to In-Network Query Processing in Sensor Networks. *ACM SIGMOD Record*, vol. 31, no. 3, pp. 9-18, ISSN: 0163-5808

Yao, Y. & Gehrke, J. (2003). Query Processing in Sensor Networks, *Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR 2003)*, Asilomar (CA), USA, January 2003

Yi, B.; Jagadish, H., V. & Faloutsos, C. (1998). Efficient Retrieval of Similar Time Sequences Under Time Warping, *Proceedings of the 14th International Conference on Data Engineering (ICDE 1998)*, pp. 201-208, ISBN: 0-8186-8289-2, Orlando (FL), USA, February 1998, IEEE Computer Society Washington, DC, USA


**Development and Implementation of RFID Technology**
Edited by Cristina Turcu

ISBN 978-3-902613-54-7
Hard cover, 450 pages
**Publisher** I-Tech Education and Publishing
**Published online** 01, January, 2009
**Published in print edition** January, 2009

The book generously covers a wide range of aspects and issues related to RFID systems, namely the design of RFID antennas, RFID readers and the variety of tags (e.g. UHF tags for sensing applications, surface acoustic wave RFID tags, smart RFID tags), complex RFID systems, security and privacy issues in RFID applications, as well as the selection of encryption algorithms. The book offers new insights, solutions and ideas for the design of efficient RFID architectures and applications. While not pretending to be comprehensive, its wide coverage may be appropriate not only for RFID novices but also for experienced technical professionals and RFID aficionados.