

Detecting counterfeit products using supply chain event mining

Ho Sung Lee*, Hyo Chan Bang*

* IoT Platform Research Team, Electronics and Telecommunications Research Institute, Korea

hosung.lee@etri.re.kr, bangs@etri.re.kr

Abstract— Counterfeiting is a growing problem all over the world, threatening the health of consumers and lead to financial losses for legally run business. By detecting counterfeit products before they are distributed to the end-users, the problem can be prevented. In this study, we propose an alternative frequent pattern mining algorithm to discover licit supply chain patterns from trace records and a classification algorithm to distinguish counterfeit products with these licit supply chain patterns. The presented algorithms are studied with computer simulations that model the flow of genuine and counterfeit products in a comprehensive supply chain. The results suggest that these algorithms could be used to automatically detect suspicious products.

Keywords— Counterfeiting, EPCglobal, EPCIS, Frequent pattern mining, Radiofrequency identification, Sequential pattern mining, Supply Chain

I. INTRODUCTION

Counterfeit products are fake replica of the genuine product and threaten the health and safety of end-users and consumers. According to the OECD survey, the percentage of counterfeit products in world trade is 5~7% and the WHO estimates that approximately 10% of all drugs are counterfeit in pharmaceutical industry. To detect and prevent counterfeiting, track-and-trace-based anti-counterfeiting solution is proposed. The general idea behind this solution is to analyse the trace records of items for counterfeit indications [1].

In an EPCglobal world, the trace record is the sum of EPCIS event (EPC reading event) of a single product and provides an overview of the route that a product has taken from to the manufacturer to the retailer [2].

Referring to the above surveys, most of the EPCIS events are generated by genuine products in supply chain and the proportion of EPCIS events generated by counterfeit products is very less than. Therefore, it is possible to discover licit supply chain pattern which are frequent event sequence of genuine products from trace records and then to detect counterfeit products by comparing a trace record of a product with licit supply chain patterns.

In this paper, we propose an alternative frequent pattern mining algorithm to discover licit supply chain patterns from trace records and a classification algorithm to detect counterfeit products. In supply chain massive products are moved through same distribution channel and so trace records which have same distribution route are stored in the database

redundantly. So we construct the Sequence Tree to remove the redundancy of trace records which have same distribution channel and with this tree discover a set of frequent sequence patterns efficiently. Most previous studies require the specification of a threshold (called minimum support) for mining a complete set of sequential patterns satisfying the threshold. However, in practice, it is difficult for users to provide an appropriate threshold. To solve this problem, we propose a novel threshold (called detail degree). Finally we present a simple classification algorithm with licit supply chain patterns.

This paper is organized as follows. Section II reviews related work about anti-counterfeiting and frequent pattern mining. In Section III we present an algorithm to discover licit supply chain patterns from trace records. In Section IV we present an algorithm to detect counterfeit products with licit supply chain patterns. In Section V we evaluate the proposed algorithms with a simulation study of virtual supply chain. Section VI summarizes our study and points out some future works.

II. RELATED WORK

A. Anti-counterfeiting

The idea of use case that RFID event data can be used for anti-counterfeiting has been first suggested in [3]. The research community addresses authentication of RFID tagged products primarily by trying to make tag cloning hard using cryptographic tag authentication protocols [4].

The track-and-trace based anti-counterfeiting solutions are proposed [5, 6]. With rule-based anti-counterfeiting solution, companies can design conditions, which give indication for counterfeiting activities. These conditions are implemented as rules. When a condition is broken, a rule triggers an alert [5]. With location based product authentication solution, it is a series of probabilistic techniques to enable reliable detection of cloned tags in cases where the visibility is incomplete. This solution is based on hypothesis that cloned tags create abnormal event that can be detected in the trace records. This solution builds an automatic location-based authentication system by evaluating transition probabilities between the RFID events [6].

B. Frequent pattern mining

Frequent pattern mining has been a focused theme in data mining research for over a decade. Abundant literature has been dedicated to this research and tremendous progress has been made, ranging from efficient and scalable algorithms for frequent itemset mining in transaction databases to numerous research frontiers. Frequent patterns are itemsets, subsequences, or substructures that appear in a data set with frequency no less than a user-specified threshold (called minimum support). For example, a set of items, such as milk and bread that appear frequently together in a transaction data set, is a frequent itemset [7].

The Apriori [8] is the first algorithm of frequent pattern mining research for market basket analysis in the form of association rule mining. It analyses customer buying habits by finding associations between the different items that customers place in their “shopping baskets”. For instance, if customers are buying milk, how likely are they going to also buy cereal (and what kind of cereal) on the same trip to the supermarket? Such information can lead to increased sales by helping retailers do selective marketing and arrange their shelf space [7].

The FP-growth mines the complete set of frequent pattern without candidate generation to avoid the costly generation of a large number of candidate set and is about an order of magnitude faster than the Apriori algorithms [9]. However, we could not use this algorithm to our case because the FP-growth does not consider the order of items either within a transaction or across transactions.

The PrefixSpan is a pattern-growth approach to sequential pattern mining and it works in a divide-and-conquer way [10]. A performance comparison shows that the PrefixSpan has the best overall performance [7]. However, we could not use this algorithm to our case because the PrefixSpan mines the complete set of frequent sequential patterns which have subsequence redundancy.

The CloSpan is an alternative but equally powerful solution: instead of mining the complete set of frequent subsequences and it mines frequent closed subsequences only, i.e., those containing no super-sequence with the same support [11]. This method is based on a property of sequence databases, called equivalence of projected databases, stated as follows: Two projected sequence databases, $S|_{\alpha} = S|_{\beta}$, $\alpha \subseteq \beta$, are equivalent if and only if the total number of items in $S|_{\alpha}$ is equal to the total number of items in $S|_{\beta}$, where $S|_{\alpha}$ is the projected database with respect to the prefix α . Based on this property, the CloSpan can prune the non-closed sequences from further consideration during the mining process [7]. The TSP is an alternative algorithm to the CloSpan and mines top-k closed sequential patterns with the minimum length instead of the minimum support [12]. However it is difficult for users who discover frequent sequential patterns with the both algorithm [11, 12] to decide an appropriate threshold. To overcome this difficulty, we propose an alternative frequent pattern mining algorithms.

III. DISCOVERING LICIT SUPPLY CHAIN PATTERNS

In this section, we first define some notation and terminology, and then illustrate our proposed method using an example.

Let $E = \{e_1, e_2, \dots, e_n\}$ be a set of all events. A **sequence** is an ordered list of events, which is a single trace record. A **sequence database** D_s is a set of tuples $\langle pid, s \rangle$, where pid is a product identifier and s is a sequence. The **support of the event**, sup_{ev} is a frequency of occurrence of an event in D_s . The **support of the sequence**, sup_{seq} is a frequency of occurrence of a sequence in D_s . The **detail degree** T_d is a user-specified threshold, which is a kind of percentile by the support of event. The **minimum support**, min_sup is a threshold of frequency. The **sequence representatives** S_r is a set of sequence patterns, which identify and characterize sequence families.

For the sequence database D_s in Table 1 with $T_d = 0.90$, sequence representatives in D_s can be minded in the following steps.

TABLE 1. SEQUENCE DATABASE

Product Identifier (EPC)	Event Sequence
urn:epc:id:sgtin:8806435.005851.8801	a, b, c, d, e, f
urn:epc:id:sgtin:8806435.005851.8802	a, b, c, d, e, f
urn:epc:id:sgtin:8806435.005851.8803	a, b, c, d, e, f
urn:epc:id:sgtin:8806435.005851.8804	a, b, c, d, e, f
urn:epc:id:sgtin:8806435.005851.8805	a, b, c, d, e, f
urn:epc:id:sgtin:8806435.005851.8806	a, b, h, e, f
urn:epc:id:sgtin:8806435.005851.8807	g, b, c, d, e, f
urn:epc:id:sgtin:8806435.005851.8808	g, b, c, d, e, f
urn:epc:id:sgtin:8806435.005851.8809	g, b, c, d, e, f
urn:epc:id:sgtin:8806435.005851.8810	g, b, c, d, e, f

Step 1: Construct Event Support Table and Sequence Tree.

Scan D_s . If a new event appears, add an entry to Event Support Table and set the support to 1. If not, increase the support. And then update Sequence Tree; each tree node has the count. When read all event sequence in D_s , calculate the support index of each event. The support index is the proportion of support in the sum of support.

The Event Support Table constructed from Table 1 is shown in Table 2 and The Sequence Tree is shown in Figure 1.

TABLE 2. EVENT SUPPORT TABLE

Event Identifier	Support	Support Index
a	6	0.102
b	10	0.169
c	9	0.153
d	9	0.153
e	10	0.169
f	10	0.169
g	4	0.068
h	1	0.017

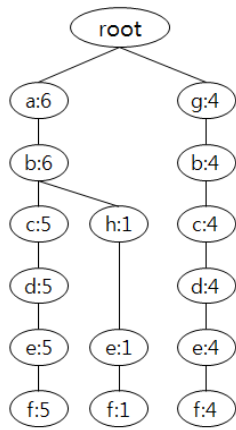


Figure 1. Sequence Tree

Step 2: Decide Minimum Support.

Sort the Event Support Table by the support descending order. And then find the entry point of the cumulative sum of support index is equal or more than T_d .

For Table 2 with $T_d = 0.90$, the min_sup is 6.

Step 3: Construct Sequence Support Table.

Traverse the Sequence Tree to construct the Sequence Support Table. The sequence identifier is the original sequence. The frequent sequence is the sequence which is consisted of frequent event, infrequent event is replaced with wildcard event (string “*”). The Length is the count of event except for wildcard event. When Sequence Tree traversal is finished, calculate the support index of each sequence in Sequence Support Table. The support index is the proportion of support in the sum of support.

The Sequence Support Table constructed from the Sequence Tree in Figure 1 is shown in Table 3.

TABLE 3. SEQUENCE SUPPORT TABLE

Sequence Identifier	Frequent Sequence	Length	Support	Support Index
a, b, c, d, e, f	a, b, c, d, e, f	6	5	0.5
a, b, h, e, f	a, b, *, e, f	4	1	0.4
g, b, c, d, e, f	*, b, c, d, e, f	5	4	0.1

Step 4: Construct Sequence Pattern Tree.

Read frequent sequence in Sequence Support Table. If a frequent sequence has wildcard event and the subsequence after wildcard event is included in another sequence, merge the frequent sequence.

The Sequence Pattern Tree constructed from Table 3 is shown in Figure 2.

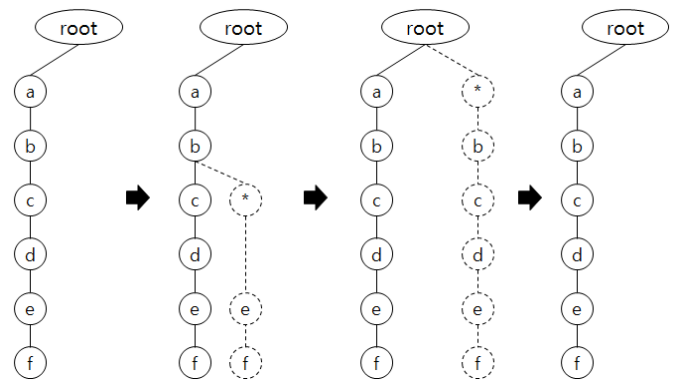


Figure 2. Sequence Pattern Tree

Step 5: Extract Sequence Representatives.

Find candidate patterns in Sequence Pattern Tree. The candidate patterns are a set of frequent sequences which satisfy the same support in a pattern cluster. The frequency index of candidate pattern is calculated by the followed (P_i is a candidate pattern, S_j is an original sequence in a pattern cluster, n is the count of original sequences),

$$\text{frequency_index}(P_i) = \sum_{j=1}^n \text{support_ratio}(P_i, S_j) * \text{support_index}(S_j) \quad (1)$$

$$\text{support_ratio}(P_i, S_j) = \begin{cases} \frac{|P_i|}{|S_j|}, & \text{if } P_i \subset S_j \\ 0, & \text{others} \end{cases} \quad (2)$$

And then select the candidate patterns which has maximal frequency index in each pattern cluster as sequence representatives.

The Candidate Pattern Table constructed from the Sequence Pattern Tree in Figure 2 is shown in Table 4. This table has a single pattern cluster and a set of sequence representatives is $\{<b, c, d, e, f>\}$.

TABLE 4. CANDIDATE PATTERN TABLE

Pattern Cluster	Candidate Pattern	Frequency Index
C1 = { <a, b, c, d, e, f>, <a, b, h, e, f> <g, b, c, d, e, f> }	a, b, c, d, e, f	0.500
	b, c, d, e, f	0.750
	b, *, e, f	0.510

These sequence representatives are licit supply chain patterns. These are used to detecting counterfeit products.

IV. DETECTING COUNTERFEIT PRODUCTS

The classification algorithm for detecting counterfeiting products is very simple. It uses regular expression technique. The sequence representatives are converted to regular expression, for example $<b, c, d, e, f>$ is converted to “ b, c, d, e, f ” and $<s1, s2, *, s34, s56>$ is converted to “ $s1, s2, ([0-9a-z,]*), s34, s56$ ”.

With regular expression library, confirm whether an event sequence of product match at least one regular expression in a set of sequence representatives. If it is matched, the product is genuine. If not, the product may be counterfeit.

V. EXPERIMENT

A. Simulation study

We evaluate the proposed algorithms with a comprehensive supply chain which is shown in Figure 3. An ellipse is a reading point of RFID event and has a unique identifier. We generate various dataset based this supply chain.

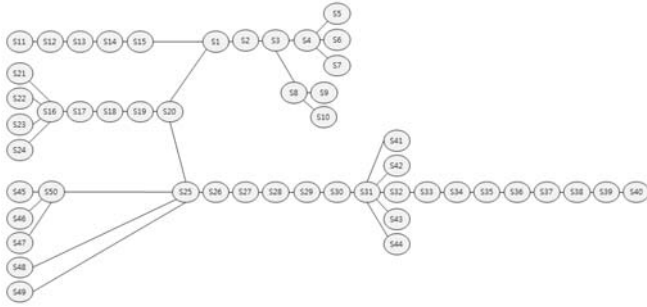


Figure 3. Virtual supply chain for simulation study

B. Performance of discovering licit supply chain patterns

The two kinds of performance experiment are performed on 2.80GHz Intel Core i7 PC machine with 4GB main memory.

The first is a comparison of runtime per the size of dataset and the results show that runtime is increased linearly by the size of sequence database (Figure 4).

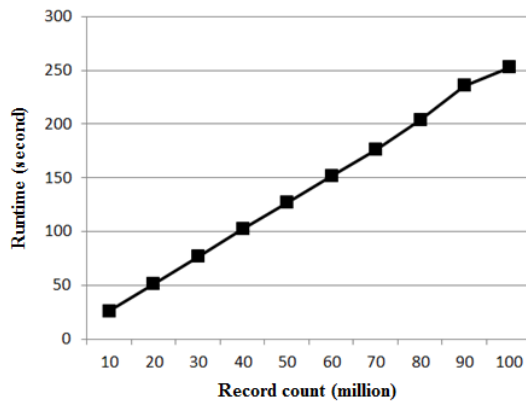


Figure 4. Runtime per the size of dataset

The second is a comparison of runtime per the process count in dataset and the results show that the runtime is not affected much by the process count in supply chain (Figure 5 and Table 5).

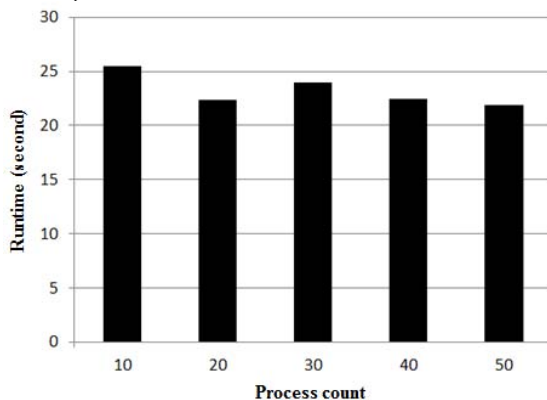


Figure 5. Runtime per the process count

TABLE 5. THE SIZE OF SEQUENCE DATABASE PER RPCESS COUNT

Process Count	Dataset Size (KB)	Runtime (second)
10	536135	26
20	448735	22
30	487962	24
40	506967	22
50	500304	22

The results from above experiments show that the cost of reading sequence database is higher than other processing cost.

C. Accuracy of detecting counterfeit product

We evaluate the classification algorithm for 5 kinds of training dataset and 5 kinds of detail degree, T_d (Table 6). The dataset for training is generated with **counterfeit ratio**, R_c which is the proportion of trace record of counterfeit products in training dataset. The test dataset is generated from training dataset and has 100 records.

TABLE 6. ACCURACY MATIX

$R_c \backslash T_d$	0.99	0.97	0.95	0.93	0.90
0.01	0.86	0.82	0.82	0.82	0.82
0.05	0.81	0.79	0.79	0.79	0.79
0.03	0.82	0.77	0.77	0.77	0.77
0.07	0.83	0.76	0.76	0.76	0.76
0.10	0.84	0.78	0.78	0.78	0.78

In case of $T_d = 0.99$, the accuracy is the highest among them, but it is not a good result than we expected. So we check the confusion matrix and the results show the average of False Positive is about 0.16.

VI. CONCLUSIONS

In this paper, we present a frequent pattern mining algorithm to discover licit supply chain patterns from trace records and a classification algorithm to detect counterfeit products. We can get good result for performance. However we need to improve the accuracy of detecting counterfeit products to apply to the pharmaceutical industry.

We propose a novel threshold called “detail degree”. By distribution of event in sequence database, the minimum support is changed adaptively.

Our future work will apply this solution to the pharmaceutical industry and modify these algorithms to be able to run on Hadoop-MapReduce framework.

ACKNOWLEDGMENT

This work was supported by the Ministry of Knowledge Economy (MKE) of the Republic of Korea. [Project No. 65-2010-43, On-site Applicable Intelligent Software and Common Standard Platform Development]

REFERENCES

- [1] ETH Zurich, SAP Research, “Problem-Analysis Report on Counterfeiting and Illicit Trade”, Deliverable 5.1 of EU-BRIDGE Project, [Online]. Available: <http://bridge-project.eu/data/File/BRIDGE%20WP05%20Anti-Counterfeiting%20Problem%20Analysis.pdf>, 2007.

- [2] EPCglobal Inc. "EPC Information Services (EPCIS) Version 1.0.1 Specification", <http://www.gs1.org/epcglobal>, [Online]. Available: http://www.gs1.org/gsm/kc/epcglobal/epcis/epcis_1_0_1-standard-20070921.pdf, 2007.
- [3] T. Staake, F. Thiesse, E. Fleisch, "Extending the EPC Network – The Potential of RFID in Anti-Counterfeiting", *Proceeding of ACM Symposium on Applied Computing*, pp. 1607-1612, 2005.
- [4] A. Juels, "RFID security and privacy: a research survey", *Selected Areas in Communications, IEEE Journal*, pp. 381-394, 2006.
- [5] J. Al-Kassab, M. Lehtonen, F. Michahelles, "Anti-Counterfeiting Prototype Report" Deliverable 5.4 of EU-BRIDGE Project, [Online]. Available: http://www.bridge-project.eu/data/File/BRIDGE_WP05_Anti_Counterfeiting_Prototype_Report.pdf, 2008.
- [6] M. Lehtonen, F. Michahelles, E. Fleisch, "How to detect cloned tags in a reliable way from incomplete RFID traces", *Proceeding of the IEEE International Conference on RFID*, pp. 257-264, 2009.
- [7] J. Han, H. Cheng, D. Xin, X. Yan, "Frequent pattern mining: current status and future directions", *Data Mining and Knowledge Discovery archive Volume 15 Issue 1*, pp. 55-86, 2007.
- [8] R. Agrawal, T. Imielinski, A. Swami, "Mining association rules between sets of items in large databases", *Proceeding of the ACM SIGMOD International Conference on Management of Data*, pp. 207-216, 1993.
- [9] J. Han, J. Pei, Y. Yin, "Mining frequent patterns without candidate generation", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 1-12, 2000.
- [10] J. Pei, J. Han, B. Mortaravi-Ad, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu, "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth", *Proceedings of the 17th International Conference on Data Engineering*, pp. 215-224, 2001.
- [11] X. Yan, J. Han, and R. Afshar, "CloSpan: Mining closed sequential patterns in large datasets", *Proceeding of the 2003 SIAM international conference on data mining*, pp. 166-177, 2003.
- [12] P. Tzvetkov, X. Yan, J. Han, "TSP: Mining top-k closed sequential patterns", *Knowledge and Information Systems, Volume 7, Issue 4*, pp. 438-457, 2005.