Proceedings of the 2015 International Conference on
Advanced Mechatronic Systems, Beijing, China, August, 22-24, 2015

Complex Event Processing on Uncertain Data Streams in Product Manufacturing Process

Na Mao, Jie Tan
Integrate Information System Research Center
Institute of automation, Chinese Academy of Sciences
Beijing, China
{na.mao, jie.tan}@ia.ac.cn

***Abstract*—With the development of automatic production, manufacturing factories record tremendous amounts of data with sensor devices deployed in a factory. Because of the inherent inaccuracy of sensor readings, these data is of high level of uncertainty. How to use Complex Event Processing (CEP) to get useful information for quality monitoring of products from a lot of uncertain raw data continually generated from the production lines is becoming a challenging research. Therefore, in this paper, we propose a model of uncertain complex event processing system for real-time monitoring in product manufacturing process. And then we define the probabilistic event model and propose a probabilistic event detection algorithm based on rNFA and its optimization plan by event filtering. At the same time, we introduce Conditional Probability Matrix (CPM) and describe the calculation of probability of complex events with the multiplication theorem of probability. The experimental results show that our proposed method is efficient to detect complex events over probabilistic event streams with better event throughput capabilities and lower time consumption.**

Keywords*—Complex Event Processing; Uncertain Event Streams; rNFA; Event Filtering and Pruning; Quality Monitoring; Manufacturing Process.

I. Introduction

Owing to the globalization of product manufacturing, manufacturing partners have high responsibility to manufacture products effectively and make themselves more competitive. In order to improve the capacity of product lines, it needs to monitor the real-time status of the manufacturing process for high quality. Today, manufacturing factories record tremendous amounts of data with sensor devices deployed in manufacturing lines. However, the production data from manufacturing shop-floors is too simple to be directly used for business and execution level. So how to get useful information to monitor and control the quality of products from a lot of raw data continually generated from the production lines is becoming an urgent problem.

In order to solve the problem, Complex Event Processing (CEP) [1] is introduced to detect the complex and meaningful information within a factory. Using Complex Event Processing, we analyze multiple events based on event patterns or rules for identifying meaningful complex events within both large collections of events or event streams which transformed from the raw data streams. Indeed, a complex event is a composite of primitive events.

Besides, the raw data from the physical world is of high level of uncertainty because of the inherent inaccuracy of sensor readings. We classify these data into two groups: unreliability and uncertainty [2]. Here, unreliability refers to the erroneous or incomplete readings that can be corrected by deterministic cleansing rules. Instead uncertainty refers to the readings which cannot be determinately eliminated by cleansing rules due to their essence, such as the possibility in producing the same product up to standard in the same operation by different workers. In this paper, we focus only on the data streams containing uncertain readings. Since uncertain event streams widely exist in reality, dealing with the uncertainty in complex event processing raises an interesting study in the academic and industry.

However, some existing complex event processing systems, such as SASE [3], Cayuga [4] and Esper [5], assume that data is precise and certain, or it has been cleansed before processing. They don't consider the uncertainty and can't detect probabilistic complex events from uncertain event streams well. At present, Cascadia [6] and Lahar [7] systems can process RFID data with uncertainty, but they mainly focus on the probabilistic data model and related queries, instead of uncertain complex event stream and optimization for probabilistic complex event detection. There are two challenges for complex event detection from probabilistic event streams [8]. One is how to detect complex events from real-time updating probabilistic event streams. The other is how to calculate the probability of the complex event generated from relevant uncertain events.

As a result, we propose a model of uncertain complex event processing system for real-time monitoring in product manufacturing and present a probabilistic event detection algorithm based on rNFA structure in this paper. The outline of the paper is organized as follows. Section II analyzes the related work. In Section III, we present a complex event processing model for product manufacturing process. A probabilistic event detection algorithm is proposed in Section IV. The performance evaluation of our proposed method for a quality monitoring application in product lines is described in Section V. Finally, the conclusion is highlighted in Section VI.

II. Related Work

We introduce the related work of uncertain complex event processing for product manufacturing in this section. Complex event processing is used to find interesting or unusual events to the user in many types of applications from different fields. Also, CEP is applied to monitor the status of manufacturing process for effective production. In the paper of [9], the

978-1-4673-7995-3/15/$31.00

author discussed the design of RFID middleware system based on CEP with manufacturing scenarios. The authors in [10] applied CEP technology to enterprise information systems, and demonstrated how the event processing engine works and interacts with other existing information systems. Especially, the authors extract complex event pattern from workflow model in their proposed system, which makes CEP more cooperative. The paper [11] provided a methodology to model CEP using Timed Net Condition Event System to analyze and describe discrete-event dynamic systems in a manufacturing line.

Recently some works have been carried out to study the uncertainty for CEP. Segev et al. [12] suggested uncertain complex event grammar rule expression using probability theory and presented a method to calculate the probability of complex events based on Bayesian network and sampling method. The author in [13] used statistical models and Markov sequences to represent the uncertain RFID data stream and proposed to use the Markov sequence converter to achieve sequence event matching. In the work of [8], the author used CIQ and CPI-Tree structures to detect complex events with single scanning probabilistic stream. Wang et al. [14] proposed a high performance complex event processing method with Nondeterministic Finite Automaton and Active Instance Stacks over distributed probabilistic event streams. In this paper [15], the authors proposed an optimized method with DFA automaton and matching tree to detect complex events against uncertain raw input data stream.

However, recent studies mainly discuss the framework and idea about applying probabilistic CEP in manufacturing, but the details in algorithm realization and optimization.

III. Complex Event Processing Model for Product Manufacturing Process

To solve the real-time monitoring problem in product manufacturing process and react to the unusual situation quickly, we propose a model of uncertain complex event processing system for product manufacturing in this section. It includes the following three parts.

A. Framework for Real-time Monitoring Based on CEP

In order to satisfy the need of monitoring in production lines, we propose a framework for real-time monitoring based on complex event processing. Fig. 1 shows the architecture of the model [16]. We use RFID readers, sensors and other devices to get raw data from the physical world in manufacturing workshops. In CEMS, all the primitive events transformed from raw data by event processors are integrated into high-level complex events, which are used to monitor the manufacturing status, such as quality control. The notifications from CEMS are sent to enterprise applications for the support of decision making.

We illustrate our ideas through a simple scenario in the product manufacturing process. A kind of product is designed to take three key operations named as A, B and C on a workstation sequentially in a period of time. And then the product will transform to another workstation for further operations. In order to ensure the product quality, we monitor the quality of the WIP (Working in Process) when every key operation is finished. We use CEP for fast and effective recognition.

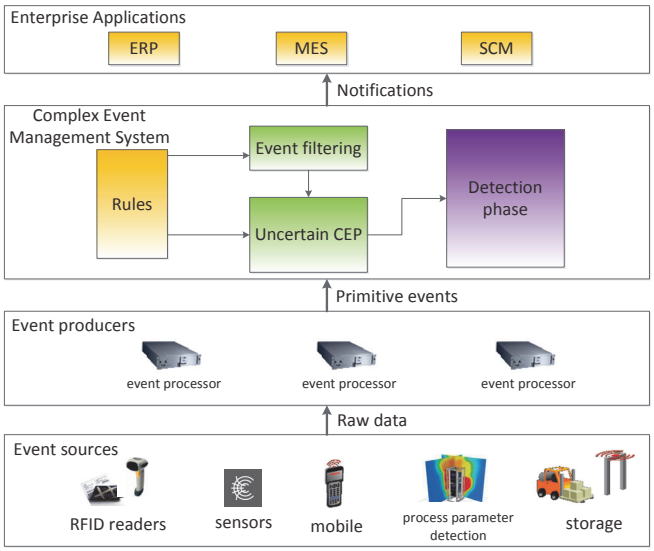

Fig. 1. CEP based model for monitoring in product manufacturing system.

Because every worker has a certain probability to produce a qualified product, the event generated from this scenario is uncertain. If the probability of complex event is less than the threshold, an alarm is made by the system.

B. Event Model

1) Events: The basic of complex event processing is events, whose definition is varied by different application situations. So we define the events we used in this paper as follows.

Definition 1 *Event.* An event is an atomic occurrence of an activity in a system [1]. We also call it primitive event. An event is represented as E = $\langle$ *id, eventType, timestamp,* {*attributes*} $\rangle$, where *id* denotes the unique identifier of the event, *eventType* is the type of the event, *timestamp* denotes the occurrence time of the event, {*attributes*} represents a set of attributes of the event, composing name and value for an attribute.

Definition 2 *Probabilistic Event.* A probabilistic event is an event with the probability value that means the occurrence probability of the event. It has the same formal representation as the certain event, but there is an attribute *Probability* with the value between 0 and 1, which is the sign of a probabilistic event. So we extend the event to E_P = $\langle$ *id, eventType, timestamp,* {*attributes*}, {*Probability, (0, 1)*} $\rangle$.

For example, an event in the example scenario mentioned above e_1 = $\langle$ "10101" , "OperationA", "2015-05-27 11:03:20", {Workstation, 1}, {WorkerID, 4}, {ProductID, 221}, {probability, 90%} $\rangle$ is a probabilistic event that indicates a worker with Number 4 takes Operation A in the first workstation for the product with Number 221 at the time 2015-05-27 11:03:20, and the probability of qualified operation is 90%. If an event is a certain event, there is no attribute of probability or the value of probability is 1.

Definition 3 *Complex Event.* A complex event is a combination of primitive or complex events by some rules. Events are connected by a set of operators, such as SEQ, ANY, Negative and so on [17]. A complex event is represented as CE = $\langle$

E, R, T_S $\rangle$, where *E* represents the events that compose the complex event, *R* represents the rule of the combination, T_S represents the time span of the complex event.

Definition 4 *Probabilistic Complex Event.* A probabilistic complex event is composed by events from the uncertain event streams, which can be represented as $CE_P = \langle E, R, T_S, Pr \rangle$. We define *Pr* the probability that the complex event occurs. It can be considered as the reliability of the pattern query during the corresponding the time interval T_S and calculated in the complex event matching process. In the example mentioned above, the probability of complex events gotten from event streams is defined as the qualified rate of products in the manufacturing process.

2) Rules: Complex events are processed according to event rules. We present a complex event query language for specifying complex event rules over streams in this section.

Definition 5 *Complex Event Query Language.* We use a declarative pattern matching language proposed in SASE [17] in this paper. A complex event query is defined using the syntax as follows:

[FROM ⟨ input stream ⟩]
PATTERN ⟨ pattern structure ⟩
[WHERE ⟨ pattern matching condition ⟩]
[WITHIN ⟨ sliding window ⟩]
[HAVING] ⟨ pattern filtering condition ⟩
[RETURN ⟨ output specification ⟩]

The FROM clause specifies the input stream we used, if this clause is omitted, the default stream is used as input stream. We next explain the other constructs using the examples drawn from the scenarios of quality control in production lines before.

Query 1:
PATTERN SEQ (OperationA a, OperationB b, OperationC c)
WHERE skip-till-next-match (a, b, c)
{[Workstation] ∧ [ProductID]}
WITHIN 1 hour
HAVING Conf (*) $>$ 0.7
RETURN *

Query 1 retrieves the qualified operations whose probability is greater than 0.7 in the same workstation for the same product. The PATTERN clause declares the structure of a pattern to be matched against the input stream. We use the SEQ construct to specify a sequence pattern of three components, which means Operation A, B and C for the product should be operated sequentially.

The WHERE clause contains attribute-based predicates to restrain the relevant events defined in the pattern. In Query 1, the predicate, [Workstation] and [ProductID], require the three operations to be done at the same workstation for the same product. It is worth noting that skip-till-next match is an event selection strategy which is introduced in [18]. The WITHIN clause specifies a sliding window over the pattern, defining the constraints that a complex event should occur within 1 hour. And then the HAVING clause further filters each pattern match by applying predicates on the constituent events. In Query 1, the predicate in HAVING requires the probability of the pattern matches to be no less than 0.7. Finally, the RETURN clause transforms each pattern match which satisfies the HAVING predicate into a result event.

C. Event Processing Model Based on NFA

SEQ event detection is the main part of the CEP engine. In this paper, we focus on the SEQ event processing for an event stream. Nondeterministic Finite Automata (NFA) is used to represent the structure of the event sequence [19]. And a basic method for SEQ event processing based on NFA and AIS is proposed in [17]. We explain the detection process using the example illustrated in Fig. 2. The NFA and AIS are created for SEQ (A, B, C). The event detection is started at the "0" state of the NFA and the SEQ event is detected after entering the "3" state. The events that trigger the change of states are logged into AIS and an event link between neighboring states is created. So the matched SEQ events that end with c_{10} are (a_1, b_3, c_{10}), (a_1, b_6, c_{10}), (a_4, b_6, c_{10}).

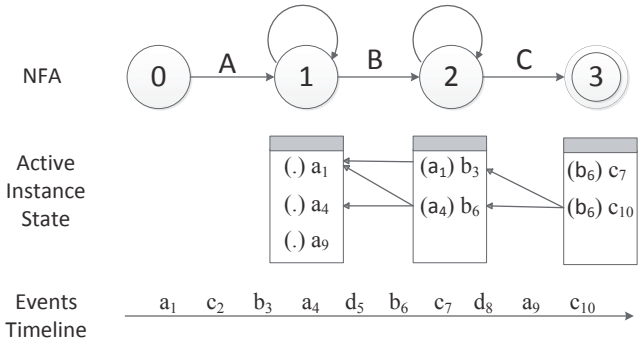

Fig. 2. Example of basic method for SEQ event processing.

However, in the product manufacturing lines, the WIP is assembled following the principle of first-in first-out (FIFO). So the event selection is satisfied with the skip-till-next-match and matches of the example in Fig. 2 are not suitable in the practical application. Besides, the constraints on events are much more complex than those for the example in Fig. 2, the basic method with NFA and AIS has a lower efficiency if the input stream is large. As a result, a new method is needed to detect SEQ event in product manufacturing process.

NFA with run buffers (rNFA). We propose the rNFA structure which is NFA combined with run buffers. Instead of AIS, a run of rNFA is used to record the events that have partial matching result for the target event pattern in the run buffer. Moreover, an accepting run is a run that has reached the final state. The rNFA for Query 1 is illustrated in Fig. 3.

IV. Complex Event Processing Under Uncertainty

In this section, we introduce the method for complex event processing under uncertainty. We describe the probability calculation model first and introduce Conditional Probability Matrix in the application for product manufacturing. And then we talk about our probabilistic event detection algorithm based on rNFA model. Besides, we propose the optimization plan considering the efficiency of pattern matching.

A. Probability Calculation Model for Uncertain CEP

Because of the uncertainty for the input stream, the complex event detected from the stream is also uncertain. In order

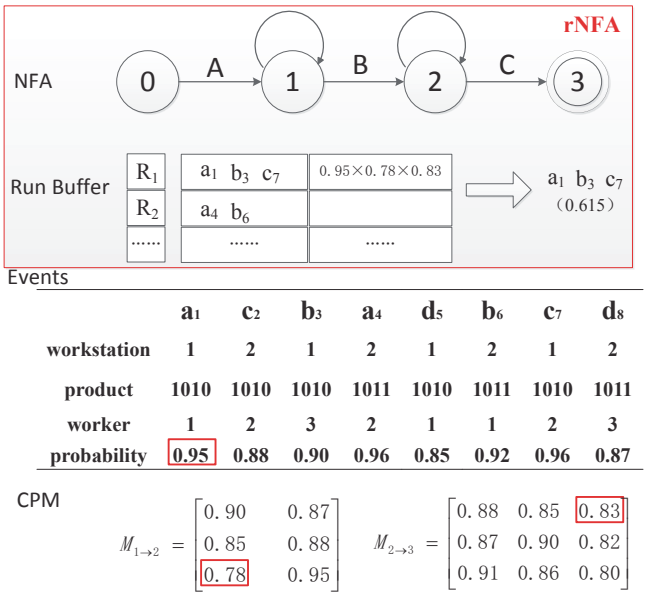

Fig. 3. Example of probability event detection based of rNFA.

to express the probability of the complex event, we propose a probability calculation model for uncertain CEP.

According to the principle of event processing based on NFA, when calculating SEQ events, some of the primitive events have Markov property. That is to say, the probability of the next event in the sequence is only related to the current event but has nothing to do with the previous events. The probability of the sequence with Markovian events can be calculated by the multiplication theorem of probability. Besides these Markovian events, we assume that there might be some primitive events that are independent of others [20]. Therefore, we can calculate the probability of an SEQ event as follows.

For an SEQ event E = SEQ (e_1, e_2, , e_n), the input stream is partitioned into set S and set T, where set T contains the independent primitive events and set S contains one or more Markov chain. The probability of event E is represented as follows:

$$Pr(E) = \prod_{e_j \in T} Pr(e_j) \bullet \prod_{e_i \in S} (Pr(e_{i1}) \prod_{m=1}^{|S_i|-1} Pr(e_{m+1}|e_m)) \quad (1)$$

where e_{i1} means the first event in the Markov chain S_i that is in set S, and $Pr(e_{m+1}|e_m)$ denotes the conditional probability of the events in the Markov chain.

Consider the example scenario in manufacturing lines. Note that qualified rate of the product differs by workers and operations. So we introduce Conditional Probability Matrix (CPM) to describe the dependence between the current event and the previous. Assuming that m workers are assigned to take Operation k and n workers for Operation k+1. The matrix $M = (p_{ij})_{n \times m}$ saves the conditional probability from Operation k to the next operation and p_{ij} means the conditional probability when the jth worker finished Operation k before the ith worker finished the next operation. However, the probability distribution and the CPM are obtained through the statistics or machine learning from historical event streams. Thus, it should be updated in a period of time.

B. Probabilistic Event Detection Algorithm

In order to detect complex events from the probabilistic event streams, we extend the evaluation model NFA and propose a novel probabilistic event detection algorithm with rNFA on the basis of the existing stream processing engine SASE. The basic idea of our algorithm is that we use rNFA model, which detects the matched complex event based on NFA and stores the intermediate results in a run of rNFA. When any event is selected to add the run, we calculate the probability of the current run. If we get an accepting run, a complex event is detected with the calculated probability. This algorithm is shown in Algorithm 1.

Algorithm 1 Probabilistic Event Detection Algorithm

Input: Uncertain Event Stream s, Complex Event Query q.
Output: Complex Event Set CE.

```
1: initialization and build NFA through Query q;
2: initialize the set of runs: activeRuns;
3: for all instance e in uncertain event stream s do
4:     if e triggers the start state of NFA then
5:         create a new run r and add event e to the run r
6:         initialize the probability of r with e.probability
7:         add the run r to activeRuns
8:     else if e triggers the state transition of NFA then
9:         for all run r in the run set activeRuns do
10:            check the predicate for transition
11:            if checkResult is true then
12:                add event e to the current run r
13:                calculate probability for the current run r
14:            else discard event e
15:            if run r is an accepting run then
16:                check the timespan of r
17:                if r.timespan ≤ timewindow then
18:                    add run r to CE and output r.probability
19:            delete run r from the run set activeRuns
20:    else discard event e
21: end
```

To explain our algorithm well, Fig.3 is used to show the example of our detection algorithm for Query 1. When event a_1 arrives, it triggers the start state of the NFA structure generated by Query 1. We add a_1 to the new run R_1 and set the probability of R_1 to be 0.95. Because event c_2 doesnt satisfy the condition of state transition of the NFA, it will be discarded. And event b_3 triggers the NFA to transfer from "1" state to "2" state. Then we check the predicate, that is, compare b_3 and a_1 with the value of Workstation and ProductID, and we find they are the same. So b_3 is added to R_1 and the probability of R_1 is updated by 0.95×0.78. Similarly, a_4 is added to a new run R_2 and b_6 is added to R_2, while d_5 is discarded. When event c_7 arrives, it triggers the NFA to transfer from "2" state to "3" state. But it just satisfies the predicate for R_1. We add c_7 to R_1 rather than R_2. The probability of R_1 is updated to $0.95 \times 0.78 \times 0.83$. R_1 is an accepting run and we output R_1 in a matched complex event (a_1, b_3, c_7) with the probability of 0.615.

C. Optimization

In order to reduce the matching cost and improve the efficiency of the detection, we filter the coming event streams before match detection to discard the irrelevant events. That is to say, we filter the events whose event type is not in the complex event query, such as event d_5 and d_8 in Fig. 3.

At the same time, we check the probability of the partial matching result in a run. If it is less than the threshold, delete the run from the run buffer. Because the value of probability is between 0 and 1, so even if the run reaches the accepting state, the probability of the run is lower than the immediate result. Therefore, we discard the run ahead of time. The optimization algorithm with event filtering and pruning plan is described in Algorithm 2.

Algorithm 2 Event Filtering and Pruning Plan

Input: Uncertain Event Stream s, Complex Event Query q, Threshold k.
Output: Complex Event Set CE.

```
1: initialization and build NFA through Query q;
2: initialize the set of runs: activeRuns;
3: for all instance e in uncertain event stream s do
4:     if e.eventType is not in the NFA model then
5:         discard event e
6:     else if update run with Algorithm 1 then
7:         if r.probability ≤ k∨r.timespan > timewindow then
8:             delete run r from activeRuns
9:         else if run r is an accepted run in timewindow then
10:            add run r to CE and output r.probability
11:        delete run r from the run set activeRuns
12: end
```

V. Experimental Results

In this section, we implement the proposed approach in a prototype uncertain CEP engine which is designed based on SASE. We add some processing functions for uncertain events and modify the original core method for optimization to evaluate our proposed method. We perform a series of experiments based on the application scenario mentioned above and present a detailed performance analysis of our algorithm.

A. Experimental Setup

The experiment is run on Microsoft Windows 7 operating systems, Intel i7-4510U 2.0GHz and 2.6GHz CPU processor, 8GB memory, and all algorithms are implemented with Java. The JVM maximum allocation pool size is 1GB. To test the system and the proposed approach, we implement an event generator to create a stream of probabilistic events through the parameters shown in Table I. In real applications, data is collected from different sensor devices and a single stream can be got by the fusion of all the data. So in our experiment, we simulate the fused event stream from the manufacturing process. Noting that the range of probability in producing qualified product is set to be [80%, 99%] for the quality monitoring in the real application.

TABLE I. Parameters for Event Generation

Parameter	Description
S	Number of the uncertain events
WS	Number of workstations
P	Number of products in processing
W	Number of workers per operation
T	Number of event types (operations)
Pro	Range of probability

B. Experimental Results

In this section, we use two common metrics, throughput and time consumption to evaluate the performance of the proposed approach. The throughput defines the number of events processed for a second by the CEP system. And the time consumption means the time of processing the incoming events.

1) Probabilistic Event Detection Algorithm: The experiment is taken to compare the performance of the proposed probabilistic event detection system (we call it PCEP) with SASE. We evaluate the throughput of the prototype with different sizes of input event streams and time window. To do so, we fixed the number of products in processing and the number of workstations at 2, the number of event types at 4 and number of workers per operation at 5. At the same time, we varied the number of events from 1000 to 500000 and the size of time window from 5 to 40, respectively.

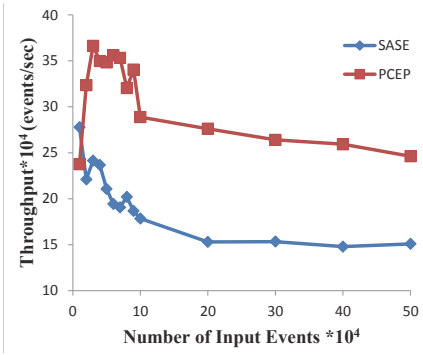

Fig. 4. Throughput over different sizes of input event streams.

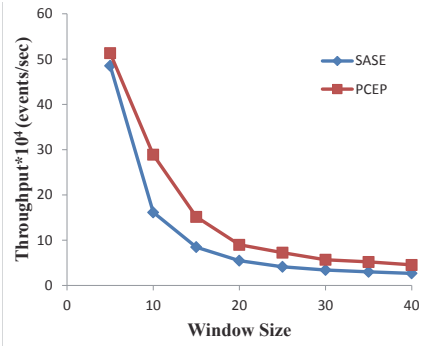

Fig. 5. Throughput over different sizes of time window.

Fig. 4 and Fig. 5 show the experimental comparison results. We can clearly observe that compared with SASE, our PCEP

shows better event throughput capabilities during the detection process for the same probabilistic event stream. As shown in Fig. 4, with the increasing number of input events, the throughput decreases for the both generally. But the throughput of our PCEP increases when the size of input streams is less than 30000, which is because our PCEP doesn't reach the the maximum capacity of the system. Fig.5 plots the throughput against the time window for the query. We can see that, as the number growing of the time window, the throughput decreases rapidly with the window size less than 20 and then in a relatively flat way. The reason for this phenomenon is that the window size is much larger than the length of event pattern which leads to longer time to get the needed complex event from the input events.

2) Event Filtering and Pruning Plan: We take the experiment to compare the performance of the proposed event filtering and pruning plan with PCEP without optimization. We evaluate the time consumption during the probabilistic event detection with different sizes of event types. In this experiment, we fixed S at 200000, WS at 2, P at 2 and W at 5, and varied T from 5 up to 30.

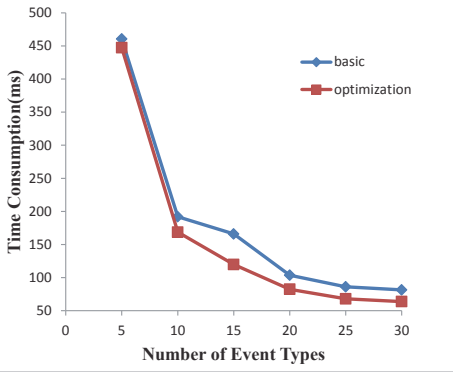

Fig. 6. Time consumption over different sizes of event types.

The experimental comparison result is shown in Fig. 6. We can see that the time consumption of the two methods differs little when the number of event types is 5. That is because the experiment on Query 1 has three valid event types and few events should be filtered.As the number to event types rising, there are much more irrelevant events for event detection. So our method with event filtering and pruning plan can filters out these events and cut down the time consumption. That is to say, the proposed optimization for PCEP has an improvement in reducing time consumption during the detection process for the same probabilistic event streams.

VI. Conclusion

In this paper, we propose a model of uncertain complex event processing system for real-time monitoring in product manufacturing. We define the probabilistic event model and propose a probabilistic event detection algorithm based on rNFA structure and its optimization plan by event filtering. At the same time, we introduce Conditional Probability Matrix (CPM) and describe the calculation of probability of complex events. The experimental results show that our proposed method is efficient to detect complex events over probabilistic event streams with better event throughput capabilities and lower time consumption.

Acknowledgment

The research is supported by the National Natural Science Foundation of China under Grant U1201251 and National Science & Technology Support Plan of China under Grant 2015BAF09B01.

References

[1] Luckham D.C., *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Boston: USA. 2001.

[2] Yanming Nie, Zhanhuai Li, and Qun Chen. *Complex Event Processing over Unreliable RFID Data Streams*. 13th Asia-Pacific Web Conference (APWeb 2011), pp 278-289, April 2011.

[3] Gyllstrom D, Wu E, Chae H J. *SASE: Complex Event Proeessing over Streams*. Rro of CIDR, pp.407-411, Dec 2007.

[4] Brerma L, Demers A, Gehrke J.*Cayuga: a high-performance event Processing engine*. Proeeedings of the 2007 ACM SIGMOD international conference on Management of data. pp.1100-1102, June 2007.

[5] Esper Team. http://www.espertech.com/. Visited Sep. 2014.

[6] E. Welbourne, N. Khoussainova. *Cascadia: a system for specifying, detecting, and managing RFID events*. pp.281C294, MobiSys, June.2008.

[7] R-e C, Letehner J, Balazinska M. *Event Querieson Correlated Probabilistie Streams*. Proc of the SIGMOD Conf. pp.715-72, June. 2008.

[8] C. Xu, S. Lin, W. Lei, et al. *Complex Event Detection in Probabilistic Stream*. Proceedings of the 12th International Asia-Pacific Web Conference (APWeb 2010). pp. 361-363, April.2010.

[9] L. Dong, D. Wang and H. Sheng. *Design of RFID middleware based on complex event processing*. IEEE conference on Cybernetics and Intelligent Systems, Bangkok (2006), pp.1-6.

[10] C. Zang and Y. Fan. *Complex event processing in enterprise information systems based on RFID*. In: Enterprise Information System, Vol. 1, No. 1 (2007), pp.3-23.

[11] W. Ahmad, A. Lobov, J. Lastra. *Formal Modelling of Complex Event Processing: A Generic Algorithm and its Application to a Manufacturing Line*. In: 10th IEEE International Conference on Industrial Informatics (INDIN) (2012), pp. 380-385.

[12] Segev W, Avigdor G, Opher E. *Handling uncertain rules in composite event systems*. 2005 LAIRS Conferenee. pp.860-861, May.2005.

[13] Kimelfeld B, Re C. *Transducing markov sequences*. Proceedings of the twenty-ninth ACM SIGMOD-SIGACTSIGART symposium on Principles of database systems of data. pp.15-26, June.2010.

[14] Yongheng Wang, Xiaoming Zhang. *Complex Event Processing over Distributed Probabilistic Event Streams*. 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery,pp.1489-1493,May. 2012.

[15] H. Kawashima, H. Kitagawa and X. Li. *Complex Event Processing over Uncertain Data Streams*. Proceedings of the fifth international conference on P2P, parallel, grid, cloud and internet computing.pp.521-526, Nov,2010.

[16] Na Mao, Jie Tan. *A Framework of RFID-based Complex Event Processing System for Assembly Manufacturing*. Applied Mechanics and Materials Vol. 751, pp 287-292, 2015.

[17] Wu E., Diao Y., Rizvi S. *High-performance complex event processing over streams*. Proceedings of the 2006 ACM SIGMOD international conference on Management of data: 407-418. New York: USA. 2006.

[18] Yanlei Diao, Neil Immerman, Daniel Gyllstorm. *SASE+: An Agile Language for Kleene Closure over Event Streams*, UMass Technical Report, 2007.

[19] Diao Y., Altinel M., Zhang H., Franklin M.J., and Fischer P.M. *Path sharing and predicate evaluation for high-performance XML filtering*. TODS, 28(4), 467-516, Dec. 2003.

[20] Y.H. Wang, K. Cao, X.M. Zhang. *Complex event processing over distributed probabilistic event streams*. Computer and Mathematics with Applications 66(2013), pp 1808-1821.