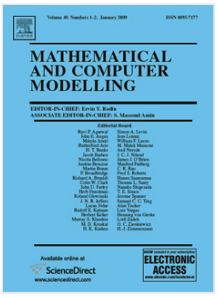
Accepted Manuscript



Deadline-aware complex event processing models over distributed monitoring streams

Yu Gu, Ge Yu, Chuanwen Li

PII: S0895-7177(11)00565-6
DOI: 10.1016/j.mcm.2011.09.017
Reference: MCM 4802

To appear in: *Mathematical and Computer Modelling*

Received date: 3 December 2010
Revised date: 8 September 2011
Accepted date: 8 September 2011

Please cite this article as: Y. Gu, G. Yu, C. Li, Deadline-aware complex event processing models over distributed monitoring streams, *Mathematical and Computer Modelling* (2011), doi:10.1016/j.mcm.2011.09.017

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

*Manuscript
Click here to view linked References
ACCEPTED MANUSCRIPT

Deadline-Aware Complex Event Processing Models over Distributed Monitoring Streams

Yu Gu * , Ge Yu , Chuanwen Li

School of Information Science and Engineering, Northeastern University, Shenyang 11004, PR China

Abstract

Complex event processing over data streams has raised a lot of attention in the database community, especially due to the development of ubiquitous data sensing equipments. Specially, some mission-critical applications have deterministic QoS requirements (e.g., event delay upper bound), in order for the results to be useful. However, most existing complex event processing methods only provide best-effort services with the statistics QoS, such as average response delay, which can not guarantee more detected composite events are responded under the deterministic QoS requirements. Therefore, it is quite imperative to provide a kind of deterministic QoS based processing model for these event-based applications. In this paper, we dedicate to the most important QoS metric in complex event processing monitoring applications, namely deadline. First, each distributed monitoring stream is modeled separately as a stochastic process according to monitoring objects' arrival situations. Then, the processing is modeled according to a generalized automata model. Moreover, a sophisticated time-cycle based framework is proposed based on the arrival and processing models to output more composite events under the deadline constraint. Finally, the experiment evaluation is made to validate the correctness and efficiency of our model for the deadline-aware complex event processing in the monitoring scenarios.

Key words: Deadline-Aware, Complex Event processing, Data Stream, Queuing Theory, Stochastic Process

* Corresponding author at:School of Information Science and Engineering, Northeastern University, Shenyang 11004, PR China
Email address: guyu@ise.neu.edu.cn (Yu Gu).

ACCEPTED MANUSCRIPT

1 Introduction

Currently, in a growing number of applications, data are naturally generated in the form of streams. Furthermore, CEP(complex event processing) over data streams [1] is considered to be valuable to support some monitoring applications such as the web access monitoring , the vehicle traffic tracking and the human behavior pattern detection. Different from the general queries based on the filter, project, join and aggregation over data streams, CEP pays more attention to correlate sequential data and detect complex pattern from the viewpoint of events. When a composite event following the desired pattern occurs, the result is desired to be output immediately for users to take further actions. Especially, with the development of monitoring equipments such as RFID and other sensors, distributed monitoring streams will be produced to reflect the spatio-temporal tracks of different monitored objects. Such as in an RFID-enabled scenario, if an object is detected in a monitored area, a primitive event containing the correspondent tag id will be defined. All these primitive events will shape into an event stream for each monitoring area. In order to correlate the information among distributed event streams, the events will be converged into a central server for the composite event detection.

QoS for data stream processing can be defined in a variety of ways and includes a diverse set of service requirements such as performance, availability, reliability, security, *etc.* All these service requirements are important aspects of a comprehensive data stream processing QoS service offering. Taking the unbounded, large volume and real-time properties of event stream into consideration, we take a more performance-centric view of data stream processing QoS and focus primarily on the issues in providing time-related delay constrains. At present, existing CEP systems only provide best-effort services. Due to the ignorance of deterministic QoS definition, they can only provide potentially statistics QoS, such as average response delay. In essence, all the event streams are mixed together and processed in a FIFO manner according to the event arrival time stamp. This kind of "QoS-unware" and "best-effort" services probably return too many outdated answers or task failures, which may lead to the low performance or even catastrophic results. To solve this problem, we will focus on the complex event processing model under deterministic QoS constrained. For the specific problem when the response delay upper bound is defined, our goal is to output most detected composite events before the calculated deadline instead of just improving the average response delay. For example, for two event type E_1 and E_2 from two corresponding event streams, the query is to detect the pattern that some E_2 type event arrives after some E_1 type event and the query is demanded to be finished under some user-defined upper-bound delay. In the case of limited resource, it is intuitional if E_2 type event stream is given the priority to be processed, more composite events satisfying the deadline constraint will be output. However,

ACCEPTED MANUSCRIPT

in this way, some E_2 events which arrive after E_1 events will be processed in advance, leading to the false query results. Therefore, some trade-off exists in this kind of resource allocation. The situation will become more complex when more event streams are involved. In addition, in order to propose the optimization allocation model, the underlying arrival model, processing model and cost model should be comprehensively and reasonably considered. Based on this fundamental models, we aim at building a logically distributed and asymmetric time-slice based framework to replace the centralized FIFO method. In general, the main contributions we make in this paper are as follows:

- The event arrival model based on the stochastic process, the processing operation based on various semantics and consumption modes and the corresponding cost model are proposed.
- The termination event deadline-satisfying model and processing disorder model for multiple event streams are inferred.
- An optimization time-sliced round-robin framework is built to asymmetrically allocate resource to different event streams based on the mathematical analysis.
- The proposed models are evaluated with practical values and other scheduling strategies in extensive experiments.

The rest of this paper is organized as follows. We begin Section 2 by the related work. In section 3, we present our event arrival and processing cost models for the monitoring streams. A basic time-sliced framework for binary operation is proposed in Section 4. A framework involving multiple-event-stream operation are further discussed in section 5. Thereafter, a set of intensive experimental results are presented in Section 6. Finally, the paper is ended with our conclusions in Section 7.

2 Related work

There has been considerable research activity pertaining to stream systems and data stream algorithms. Some systems that incorporate stream processing include Niagaracq [2], STREAM [3], TelegraphCQ [4], Aurora [5] and many others. In data stream research community, QoS is first proposed in [6]. The QoS requirements in Aurora are two-dimensional QoS graphs specified by the application administrator telling the utility of the output in terms of several performance-related attributes. The system monitors the execution performance based on these QoS metrics. If the QoS drops below an acceptable level, the system will shed load until the performance increases. Obviously, this is a kind of posteriori QoS, it may be unacceptable for many mission critical applications, especially when the quality of data contents is critical. And it can not provide the deterministic performance guarantee. Sutherland et al. propose an adaptive scheduling framework [7] that has the ability to select a

ACCEPTED MANUSCRIPT

single algorithm from a pool of algorithms available to the system. However, the requirements specify the desired behavior in relative terms. That is, the QoS is not specified for an absolute performance goal (i.e., achieve an output rate of X tuples per sec or have no more than Y tuples in the query plan at once), but rather specifies that the system should aim to maximize the output rate or minimize the queue size. Qstream [8,9] suggests the idea of priori QoS for data stream processing, however the time-related performance requirements are ensured by the resource reservation, which incurs too much resource idleness, therefore less flexible. Moreover, since Qstream is implemented on an RTOS, its deterministic behavior regarding the processing times relies on the real-time operating system environment. It can not provide a general QoS guaranteeing for data stream processing. Moreover, CEP is not considered in Qstream. Wu et al.[10] propose a new QoS-oriented query scheduling strategy. By correlating with the classical job scheduling problem, an efficient operator scheduling method is well studied. However, the framework is not designed for the fixed real-time scenarios and also complex event processing operators are not considered.

Event streams can be considered as a special type of data streams. Different from the processing manner based on the window relation algebra, event streams mainly focus on the pattern correlation of the incoming events. In recent years, event streams with the relative complex event processing techniques have attracted a lot of attention. Typically, SASE [1,11] and Cayuga [12,13] separately offer the extended event language, event query processing and optimization strategies to facilitate the event collection, cleaning, generation and correlation. Especially, some researches [14,15] focus on the complex event processing and optimization in the RFID monitoring scenarios. In addition, [16] extends the complex event processing techniques into distributed scenarios and a query plan partition framework is proposed. Also, EStream [17] proposes the efficient incremental event detection mechanisms based on the data stream query engine. Moreover, as a process query system, PQS [18] takes the event steams as the input parameters and the process of the events is detected according to the model solutions. Furthermore, Cascadia [19] and Laha [20] take account of the potential uncertain of the events and propose the efficient event inference and analysis models, especially for the RFID monitoring applications. Also, Liu et al.[21] study the problem of multi-dimensional event sequence processing. Specifically, a novel E-Cube model is demonstrated that combines CEP and OLAP techniques for multi-dimensional event pattern analysis at different abstraction levels. Recently, in order to handle the opportunities and risks detected by pattern queries, Wang et al.[22] propose to embed the active rule support within the CEP engine, a novel stream-oriented transactional model and a family of stream transaction scheduling algorithms are introduced to ensure the correctness of concurrent stream execution. However, to out best knowledge, the state-of-the-art complex event processing frameworks only focus on the best-effort service in the soft-time

ACCEPTED MANUSCRIPT

applications.

3 Arrival and Processing Models for Event Streams

In this section, we will first model the primitive event arrival and CEP processing without considering delay constraints, which is the foundation of analyzing our asymmetric time-sliced processing framework.

3.1 Arrival Model for Primitive Events

Our focus is on the complex event processing in monitoring applications. In this sort of applications, a logic area can be defined as some specified region where monitoring equipments are deployed. A logic area can be defined as an intersection, a room even a web page. Especially with the development of the virtual world and CPS, such monitoring applications and logic areas can be widespread. When some information is detected by the logic area, a primitive event will be produced including the time, location and object information. For example, when a tagged car comes into an RFID reader monitored logic area, a corresponding primitive event will be produced. In this way, for each distributed logic area, all the primitive events will pool into a monitoring event stream which will be modeled with the same event type. In essence, the rate of each monitoring data stream is related to the regularity for an "object" to enter a logic area. We can find some statistical characteristics for this kind of access. Especially, the mathematical model based on Stochastic Process is very suitable to describe the arrival regulation in each logic area and accordingly the event arrival discipline. For most monitoring applications, the specific stochastic process model can be chosen according to application features and historically statistical information. Also, the hypothesis testing and advanced statistical methods can be adopted to define the parameters more accurately.

Specially, *Poisson Stochastic Process* namely *Poisson Stream* is widely used in real-world service system to express the long-term and independent customer arrival features, which is also suitable for many monitoring scenarios such as the vehicle traffic and web access monitoring. In this paper, because we do not focus how to efficiently estimate the arrival model, we just take *Poisson Stream* as the typical example of our stochastic arrival model and employ it to make the further analysis. Equation 1 defines the expression of *Poisson Stochastic Process*, where $M(t)$ represents the volume of the data arrival during the arbitrary interval t . In the same way, our primitive event arrival rate in this paper can be modeled according to equation 1.

ACCEPTED MANUSCRIPT

$$\pi(n,t) = P\{M(t) = n\} = e^{-\lambda t}\frac{(\lambda t)^n}{n!} \tag{1}$$

3.2 Complex Event Processing Semantics and Cost Model

In order to allocate the resource efficiently, the reasonable model for the CEP and corresponding cost analysis are necessary. In this subsection, we first transform various operations into the sequence operations and propose complex consumption modes to reflect different application scenarios. Reasonable cost models are discussed according to the feature of various consumption modes.

3.2.1 Event Operation Model

Supposing the consecutive repeated events of some type which occur between other event types are considered as an event, we first define the following event processing operators.Note that E_i represents some event type and e_i represents some event instance of this type.

- **Sequence**: denoted as $SEQ(E_1, E_2)$ for binary events, satisfying $t_a^1 < t_a^2$, where t_a^i denotes the arrival time stamp of e_i. Sequence is the fundamental and most common operation in event monitoring applications. By further correlations, $SEQ(E_1, E_2, \cdots, E_n)$ reflect the sequence operations of multiple events.
- **Non-sequence**: including conjunction denoted as $CON(E_1, E_2, \cdots, E_n)$, disjunction denoted as $DIS(E_1, E_2, \cdots, E_n)$ and negation denoted as $NEG(E_i)$. Negation operation usually needs to be used together with sequence operation.
- **Constraint**: including partition constraint Partition(E_i, id), which will partition the events according to the id before the correlation; temporal constraint TCons($E_i, TExp()$), such as Within(E_i, t_1, t_2) satisfying $t_1 < E_i.t_{end} - E_i.t_{begin} < t_2$ to limit efficient interval of the detected composite event; value constraint VCons($E_i, VExp()$) to offer the time-irrelevant attribute restrictions; continuity constraint Continuity($E_i, \mathcal{S}_c$) to represent whether irrelative events are allowed to occur during the correlation, such as Continuity(E_i, ALL) to reject all the irrelative events and Continuity(E_i, E_i') to reject the irrelative events of type E_i. We utilize # to denote irrelevant event instances for Continuity operator.

Irrespective of the consumption modes, sequence operation can be represented by the typical automata model $\langle Q, \Sigma, \delta, q0, F\rangle$. Based on the automata model, other operations can be conducted following some kind of order which is partially illustrated in figure 1. Especially, we call the events in the termination

ACCEPTED MANUSCRIPT

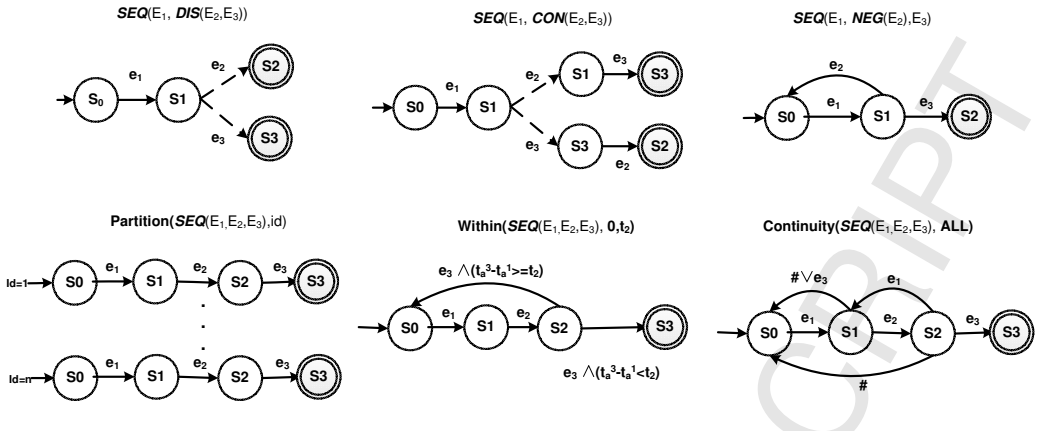


Fig. 1. Automata representation for different event operations

state as the termination events while other events are called non-termination events.

3.2.2 Cost Model for Unrestricted Consumption Mode

In the event detection, another key problem which can not be described by the above event operators is how to explain the semantics of the continuous event occurrence of the same type. Also, whether the events which have been matched will continue to match other events will affect the query results. This issue is just the consumption mode problem, namely event context problem. Typically, unrestricted mode requires all the matched events to continue the further correlation during some window interval.

For the non-termination event of the unrestricted mode, because only the insertion and pointer modification are involved, the cost can be considered stable. But for the termination event, the cost is proportional to the number of the matched events. We utilize e_i^j to represent the jth event instance of the type i. Supposing the termination event e_m^n will match $\chi(e_m^n)$ arrival events, we can infer $\chi(e_m^n)$ in equation 2 where the loop condition reflect the arrival event number between e_m^{n-1} and e_m^n.

$$\chi(e_m^n) = \chi(e_m^{n-1}) + \sum_{e_{m-1}^i.t_a \in [e_m^{n-1}.t_a, e_m^n.t_a]} \chi(e_{m-1}^i) = \sum_{e_{m-1}^i.t_a \in [e_m^1.t_a, e_m^n.t_a]} \chi(e_{m-1}^i) \tag{2}$$

Although equation 2 gives the original cost expression of the unrestricted mode, the relationship between the cost and event stream rates is not clearly illustrated. We will further expand equation 2 in the following theorems.

ACCEPTED MANUSCRIPT

Theorem 1 *For $SEQ(E_1, E_2, \cdots, E_n)$, supposing $E_1, E_2, \cdots, E_n$ have the same arrival rate and the events arrive following the order of $E_1, E_2, \cdots, E_n$. when m events arrive for each type, we can get $|E_1, E_2, \cdots, E_n|_m = C(m + n - 1, n)$, where $|E_1, E_2, \cdots, E_n|_m$ represents the composite event number and $C(m + n - 1, n)$ denotes the combination expression C^n_{m+n-1}.*

Proof. Based on the mathematical induction method, when $n = 1$, obviously $|E_1|_m = m$. And because $C(m + 1 - 1, 1) = C(m, 1) = m$, the theorem holds. Assume the theorem holds when $n = k$, namely $|E_1, E_2, \cdots, E_k|_m = C(m + k - 1, k)$, then when $n = k + 1$, we can get

$$|E_1, E_2, \cdots, E_{k+1}|_m = \sum_{i=1}^{m} |E_1, E_2, \cdots, E_{k+1}|_i = \sum_{i=1}^{m} C(i + k - 1, k) =$$

$$\sum_{i=1}^{m} \frac{(i + k - 1)!}{k!(i - 1)!} = \frac{k!}{k!0!} + \frac{(k + 1)!}{k!1!} + \frac{(k + 2)!}{k!2!} + \cdots + \frac{(k + m - 1)!}{k!(m - 1)!}$$

$$= \frac{(k + 1)!}{(k + 1)!1!} + \frac{(k + 1)!(k + 1)}{(k + 1)!1!} + \frac{(k + 2)!}{k!2!} + \cdots + \frac{(k + m - 1)!}{k!(m - 1)!}$$

$$= \cdots = \frac{(m + k)!}{(k + 1)!(m - 1)!} = C(m + k, k + 1)$$

, i.e.

$$|E_1, E_2, \cdots, E_{k+1}|_m = \sum_{i=1}^{m} C(i + k - 1, k) = C(m + k, k + 1)$$

, the theorem is proven.

According to theorem 2, assuming the arrival rate for all the event streams is v, the number of the composite events is $|E_1, E_2, \cdots, E_n|_m = C(\lfloor vt \rfloor + n - 1, n)$ at the time stamp t. If the arrival rates of distributed event streams are different, we can infer the following theorem based on theorem 1.

Theorem 2 *For n event streams, if the rate of ith event stream E_i is v_i, we can get the following estimation,*

$$|E_1, E_2, \cdots, E_n|_t \approx C(n + \lfloor v_n t \rfloor - 1, n) \prod_{i=1}^{n-1} \frac{v_i}{v_n}$$

Proof. Based on the mathematical induction method, when $n = 1$, $|E_1, E_2, \cdots, E_n|_t = |A_1|_t = \lfloor v_1 t \rfloor = C(\lfloor v_1 t \rfloor, 1)$, the theorem holds. Assume the theorem holds when $n = k$, i.e.

$$|E_1, E_2, \cdots, E_k|_t \approx C(k + \lfloor v_k t \rfloor - 1, k) \prod_{i=1}^{k-1} \frac{v_i}{v_k}$$

ACCEPTED MANUSCRIPT

, then when $n = k + 1$, we can get

$$|E_1, E_2, \cdots, E_{k+1}|_t = \sum_{t' < t} |E_1, E_2, \cdots, E_k|'_t$$

$$\approx \frac{v_k}{v_{k+1}} \sum_{t' < t} C(k + \lfloor v_{k+1} t' \rfloor - 1, k) \prod_{i=1}^{k-1} \frac{v_i}{v_{k+1}} = \prod_{i=1}^{k} \frac{v_i}{v_{k+1}} \sum_{t' < t} C(k + \lfloor v_{k+1} t' \rfloor - 1, k)$$

. According to theorem 2,

$$\prod_{i=1}^{k} \frac{v_i}{v_{k+1}} \sum_{t' < t} C(k + \lfloor v_{k+1} t \rfloor - 1, k) = C(k + \lfloor v_{k+1} t \rfloor, k + 1) \prod_{i=1}^{k} \frac{v_i}{v_{k+1}}$$

. Therefore, the theorem is proven.

In general, the unrestricted mode requires introducing the sliding window to control the processing and storage cost and avoid the resource exhaustion due to the combination explosion. If the window size is w, we can infer the number during the interval w as follows,

$$C(n + \lfloor v_n w \rfloor - 1, n) \prod_{i=1}^{n-1} \frac{v_i}{v_n}$$

Also, if the event streams follow the poisson process or other stochastic models, the cost can be estimated based on the expectation of event rates. In conclusion, if the cost coefficient is k, the cost C_χ for the unrestricted mode with the window size w can be inferred as

$$\chi(e_m^n) = \chi(e_m^{n-1}) + \sum_{e_{m-1}^i . t_a \in [e_m^{n-1} . t_a, e_m^n . t_a]} \chi(e_{m-1}^i) = \sum_{e_{m-1}^i . t_a \in [e_m^1 . t_a, e_m^n . t_a]} \chi(e_{m-1}^i) \tag{3}$$

3.2.3 Cost Model for Other Consumption Modes

In fact, most real-life monitoring applications will not utilize the unrestricted mode. To solve specific application requirements and build the correct cost model, it is very necessary to define other consumption modes based on the features of different monitoring applications. For the same query pattern, different consumption will produce different results. The context concept proposed in the active database system Snoop [23] is similar to the consumption modes except that the in-depth modeling and cost analysis are not given.

In essence, each event is corresponding to a specified state. Suppose $\mathcal{E}_j$ represents some event space, $\mathcal{S}_j = \{s_1, \cdots, s_n\}$ represents some state space and s_i

ACCEPTED MANUSCRIPT

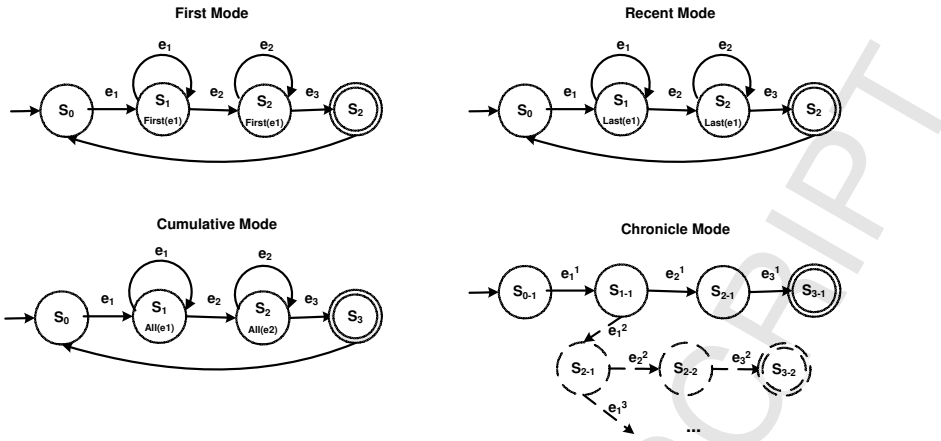


Fig. 2. Automata representation for different consumption modes

represents some specified state instance. According the the automata model, the state transition $s_i \rightarrow s_j$ can be finished in the time complexity O(1). For each state instance, the corresponding event instance may change, e.g. for s_i, $e_i^k \rightsquigarrow e_i^l$ or $\sum e_i^k \rightsquigarrow \sum e_i^l$, which can also be finished in the time complexity O(1). Also, the state derivation denoted as $\mathcal{S}_i \mapsto \mathcal{S}_j$ may occur, i.e., $\mathcal{S}_i = \{s_1, \cdots, s_n\}$ derives the new state space $\mathcal{S}_j = \{s_1, \cdots, s_n\}$. Without the implementation optimization, the cost of state derivation may be unstable. We will further introduce various consumption modes based on the model above. In addition, the automata representation of the consumption modes with stable cost is illustrated in figure 2.

- **Recent Mode**. For the repeated events, only the latest arrival event will participate the correlation and the matched event will not continue to participate the further correlation. For example, if some important exhibit is stolen, the last information sensed by the monitor reader should be reserved for the correlation. Recent mode requires the state transition and corresponding event instance transition, rather than the state derivation.
- **First Mode**. For the repeated events, only the earliest arrival event will participate the correlation and the matched event will not continue to participate the further correlation. For example, after the shop is closed, the first information sensed by the reader deployed at the doors will be valuable to detect the illegal invasion. Also, first mode does not need any state derivation.
- **Chronicle Mode**. Each repeated event will separately participate the correlation following the FIFO manner and the matched event will not continue to participate the further correlation. This mode is widely used in the automatic detection of a batch of commodities during the transfer chain. This mode requires not only the state transition but also the state derivation, so the implementation optimization must be designed to keep the cost stable. As illustrated in the chronicle mode of figure 2, we propose to utilize a two-dimensional linked list structure to locate the insertion position of all

ACCEPTED MANUSCRIPT

Table 1
Query results and cost models for different consumption modes

	query results of $\langle e_1^1, e_1^2, e_2^1, e_2^2, e_3^1, e_3^2\rangle$ under the query $SEQ(E_1, E_2, E_3)$	costs of the non-termination events	additional costs of the termination events
Unrestricted Mode	$(e_1^1, e_2^1, e_3^1), (e_1^1, e_2^2, e_3^1), (e_1^2, e_2^1, e_3^1), (e_1^2, e_2^2, e_3^1), (e_1^1, e_2^1, e_3^2), (e_1^1, e_2^2, e_3^2), (e_1^2, e_2^1, e_3^2), (e_1^2, e_2^2, e_3^2)$	$C_{insert} + C_{pointer} + \{C_{hash}\}$	$kC(n + \lfloor v_n w \rfloor - 1, n) \prod_{i=1}^{n-1} \frac{E(v_i)}{E(v_n)}$
First Mode	(e_1^1, e_2^1, e_3^1)	$C_{trans} + \{C_{hash}\}$	$\{C_{comp}\}$
Recent Mode	(e_1^2, e_2^2, e_3^1)	$C_{trans} + \{C_{hash}\}$	$\{C_{comp}\}$
Chronicle Mode	$(e_1^1, e_2^1, e_3^1), (e_1^2, e_2^2, e_3^2)$	$C_{insert} + \{C_{hash}\}$	$C_{delete} + \{C_{comp}\}$
Cumulative Mode	$(\sum\{e_1^1, e_1^2\}, \sum\{e_2^1, e_2^2\}, e_3^1)$	$C_{trans} + C_{aggr} + \{C_{hash}\}$	$\{C_{comp}\}$
Continuous Mode	$(e_1^1, e_2^1, e_3^1), (e_1^1, e_2^2, e_3^1), (e_1^2, e_2^1, e_3^1), (e_1^2, e_2^2, e_3^1)$	$C_{insert} + \{C_{hash}\}$	$k \prod_{i=1}^{n-1} \frac{E(v_i)}{E(v_n)}$

the events. For the termination event, only a linked-list deletion operation is needed.

- **Cumulative Mode**. All the repeated events will separately participate the correlation as a whole and the matched event will not continue to participate the further correlation. For example, the patient's continuous heartbeat stop monitoring is needed to be judged as a whole according to some repetition number threshold. Also, because no state derivation will be incurred and the cumulation can be calculated in the incremental manner, each event processing cost is stable.
- **Continuous Mode**. Each repeated event will separately participate the correlation following the FIFO manner. For the non-termination event, the correlation process will be continued until some termination event is detected. For example, the mode can be useful for the suspect analysis of the non-payment supermarket commodity. Continuous mode can be considered as a special case of unrestricted mode with the window size $1/v_n$. According to equation 3, the termination event cost can be estimated as $k \sum_{i=1}^{n-1} \frac{E(v_i)}{E(v_n)}$.

In table1, we conclude the query results and cost estimations for different consumption modes under the query $SEQ(E_1, E_2, E_3)$. $\{\}$ represents the optional cost according to the applications. Based on this stable cost model, we can further propose our deadline-aware resource allocation strategies in the next section.

4 Deadline-aware Processing Model for Binary Streams

4.1 Deadline Satisfying Ratio Model

According to the statistical arrival model, suppose event streams follow *Poisson Stochastic Process* and is independent with each other. According to our processing cost model based on the extended automata, when the *continuous* mode and *unrestricted* mode employ approximate average cost model, we can

ACCEPTED MANUSCRIPT

infer the processing cost for each primitive event is stable and tends to be a const. First, we suppose there are n processors, each of which is responsible for the service of one event stream. Thus we may not consider the correlation of event streams and only aim at the analysis on one kind of event stream. In the real case, a corresponding simulated time-sliced round-robin processing model is adopted in the central server processor. Suppose λ is the parameter of poisson process, which represents the average arrival rate and constant μ is used to express the number of primitive events processed per time unit with the allocated resource, i.e., $t_p = 1/\mu$ represents the processing time for each primitive event. On the basis of the description of the model above, we can find that it accords with M/D/1 model of the queueing theory [24], where M represents the poisson process and D represents the stable service cost. Suppose $\rho = \lambda/\mu$, we will make the important conclusions as follows.

If $\rho < 1$, the limit distribution of the virtual waiting time $\tilde{t_w}$ and waiting time t_w both exist, and the two values are equal independent of the initialization conditions, namely equation 4 holds,

$$\lim_{x \to +\infty} P\{\tilde{t_w} \leq x\} = \lim_{x \to +\infty} P\{t_w \leq x\} = W(x) \tag{4}$$

and the L-S transformation of $w(x)$ can be inferred as equation 5.

$$W^*(x) = \frac{(1-\rho)x}{x - \lambda + \lambda e^{x/\mu}} (Re(x) > 0) \tag{5}$$

If $\rho \geq 1$, then we can conclude equation 6. In this situation, t_w will not obey any probabilistic distribution.

$$\lim_{x \to +\infty} P\{\tilde{t_w} \leq x\} = \lim_{x \to +\infty} P\{t_w \leq x\} = 0 \tag{6}$$

Furthermore, for M/D/1 model, the average waiting time can be inferred exactly as equation 7.

$$E(t_w) = \frac{\rho}{2\mu(1-\rho)} + \frac{1}{\mu} \tag{7}$$

From the analysis above, we can see that the event waiting time depends on the event arrival rate and event processing time. The faster event arrival rate and the larger event processing time will lead to the larger event waiting time. Under extreme conditions that the average processing time is larger than the average arrival time interval, the system will be overloading all along. Thus

ACCEPTED MANUSCRIPT

whether the system is able to run consecutively can be judged by ρ, and in general systems, $\rho < 1$ should be assured.

The sum of the event waiting time t_w and the allocated event processing time t_p is called the event response delay t_l, namely $t_l = t_w + t_p$. Generally the response delay is used to assure that the system is real-time, so the upper limit of the response delay t_d may be restricted in the monitoring systems. In this situation, each event e_i should have its own deadline $e_i.t_D = e_i.t_a + t_d$. Note that the event arrival rate is objective and can not be changed by the system, so in order to reduce the response delay, we can only reduce the event processing cost by increasing the allocated time processing for this event stream. Suppose that QoS which is required by users is defined as follows: $\langle t_d, \sigma \rangle$, where σ denotes the satisfying ratio of all the events for the response delay upper bound t_d. Thus the relationship between σ and t_d can be inferred as equation 8.

$$P\{t_l \leq t_d\} = P\{t_w + t_p \leq t_d\} = P\{t_w \leq t_d - 1/\mu\} \geq \sigma \tag{8}$$

The average waiting time based on equation 8 is not enough for the further evaluation while L-S transformation based on equation 5 is complicated and can not be calculated easily. Based on the consideration that we only concern the upper limit of the response delay, the system may be thought to be close to saturation service state at the moment that the response delay reaches the upper limit. Under the state of the saturation service, according to the mathematically approaching theory[24], we can prove the approximate probabilistic density function of the event response delay t_l given μ as follows:

$$t_l \sim f(x) = \begin{cases} 2(\lambda - \lambda^2/\mu)e^{-2(\lambda-\lambda^2/\mu)(x-1/\mu)} & x \geq t_p \\ 0 & x < t_p \end{cases} \tag{9}$$

Furthermore, the probabilistic density function of the event waiting time t_w given μ can be inferred as follows:

$$t_w \sim f^{'}(x) = \begin{cases} 2(\lambda - \lambda^2/\mu)e^{-2(\lambda-\lambda^2/\mu)(x)} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{10}$$

Therefore, the deadline satisfying ratio σ can be calculated as equation 11.

$$\sigma = P\{t_w < t_d - t_p\} = \int_{-\infty}^{t_d - t_p} f(x)dx = 1 - e^{-2(\lambda-\lambda^2 t_p)(t_d - t_p))} \tag{11}$$

ACCEPTED MANUSCRIPT

The satisfying ratio σ given μ, denoted as $\theta_\sigma(\mu)$ is illustrated in equation 12.

$$\theta_\sigma(\mu) = 1 - e^{-2(\lambda-\lambda^2/\mu)(t_d-1/\mu))} \tag{12}$$

In the same way, given σ and t_d, we can infer the minimum allocated μ as follows,

$$\mu \geq \frac{2\lambda}{1 + t_d\lambda - \sqrt{(1+t_d\lambda)^2 - 2(\lambda t_d \ln(1-\sigma) + 2\lambda t_d)}} \tag{13}$$

Note that until now, we only consider the processing of the single primitive event. But our task is to detect composite events, so the t_d constraint should be imposed on one composite event rather than each of its constituent primitive events. Because one composite event is considered to be detected once its termination event e_n is detected, the t_d constraint should be essentially imposed on e_n. For $SEQ(E_1, E_2, \cdots, E_n)$, we can get $t_d^{seq} = t_d^n$. Also, it seems that σ^{seq} is only determined by σ_n and σ^{seq} should increase with the increase of μ_n. But in fact, due to processing disorder occurrence may also increase with the increase of μ_n leading to more incorrect detected results, only allocating more μ to e_n is not reasonable. Thus, processing disordering problem should be further considered.

4.2 Binary Processing Disordering Model

In this section, we will analyze the problem of the processing disordering when only two event streams are involved. Obviously, in a time-cycle manner, if one primitive event in one event stream arrives earlier than another one in another event stream, it does not mean the earlier arrival primitive event can be processed first, i.e., $t_a^1 < t_a^2 \nRightarrow t_s^1 < t_s^2$ and $t_a^1 < t_a^2 \nRightarrow t_e^1 < t_s^2$, where t_s and t_e respectively represent the starting time and ending time of the processing. For $SEQ(E_1, E_2)$, following the strict order, $t_l^1 \leq t_w^2 + t_i(e_1, e_2)$ needs to be satisfied, representing e_2 should begin to be processed after e_1 has been processed, where $t_i(e_1, e_2)$ represents the interval of event arrivals namely $t_a^1 - t_a^2$. In the same way, we can infer the processing disordering condition is $t_l^1 - t_w^2 > t_i(e_1, e_2)$.

According to the probabilistic density function of t_l and t_w based on equation 9 and 10, we can infer the probabilistic density function of $t_l^1 - t_w^2$ as follows:

$$t_l^1 - t_w^2 \sim g(x) = \int_{-\infty}^{+\infty} t_w^2(-t) t_l^1(x-t)dt = \int_{\max(t_p^1,x)}^{+\infty} t_w^2(-t) t_l^1(x-t)dt \quad (14)$$

Assuming $h_i = 2\lambda_i - 2\lambda_i^2 t_p^i$, $g(x)$ can be inferred in equation 15,

$$g(x) = \frac{h_1 h_2 e^{h_1(-\max(t_p^1,x)+t_p^1)+h_2(x-\max(t_p^1,x))}}{h_1+h_2} = \begin{cases} \frac{h_1 h_2 e^{h_1(-x+t_p^1)}}{h_1+h_2} & x \geq t_p^1 \\ \frac{h_1 h_2 e^{h_2(x-t_p^1)}}{h_1+h_2} & x < t_p^1 \end{cases} \quad (15)$$

Furthermore, we can obtain the corresponding distribution function in equation 16,

$$G(y) = P\{t_l^1 - t_w^2 > y\} = \int_y^{+\infty} g(x)dx = \begin{cases} \frac{h_2 e^{h_1(-y+t_p^1)}}{h_1+h_2} & y \geq t_p^1 \\ 1 - \frac{h_1 e^{h_2(y-t_p^1)}}{h_1+h_2} & y < t_p^1 \end{cases} \quad (16)$$

Based on our proposed automata model, the main step to process e_1 is usually the state transition. As long as e_1 begins to be processed, even if e_2 begins to be processed before e_1 finishes the processing, the error will not occur in general situations. We can infer the relaxed processing disordering condition as $t_w^1 > t_w^2 + t_i(e_1, e_2)$. Therefore, we can simplify equation to infer the probabilistic density function of $t_w^1 - t_w^2$ as follows,

$$t_w^1 - t_w^2 \sim g(x) = \int_{-\infty}^{+\infty} t_w^2(-t) t_l^1(x-t)dt = \int_{\max(0,x)}^{+\infty} t_w^1(-t) t_l^w(x-t)dt \quad (17)$$

Furthermore, we can calculate the probabilistic density function of $t_w^1 - t_w^2$ as equation 18,

$$g(x) = \frac{h_1 h_2 e^{h_1(-\max(0,x))+h_2(x-\max(0,x))}}{h_1+h_2} = \begin{cases} \frac{h_1 h_2 e^{h_1(-x)}}{h_1+h_2} & x \geq 0 \\ \frac{h_1 h_2 e^{h_2(x)}}{h_1+h_2} & x < 0 \end{cases} \quad (18)$$

And the distribution function of $t_w^1 - t_w^2$ can be inferred as 19,

ACCEPTED MANUSCRIPT

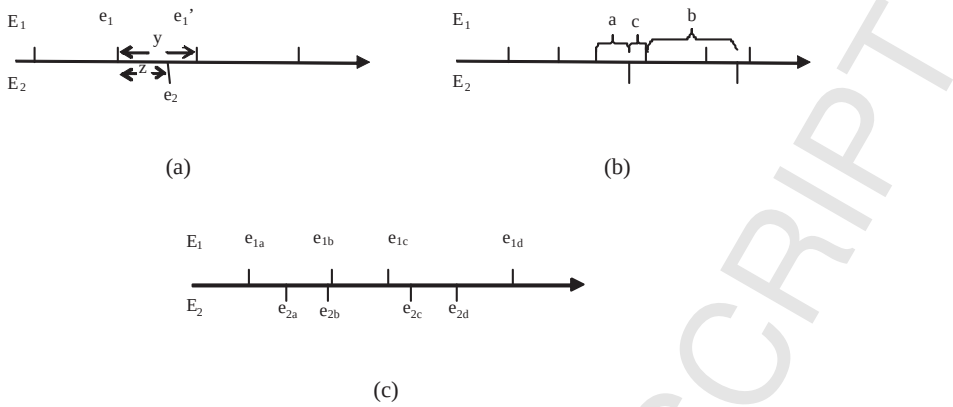


Fig. 3. $f(x)$ analysis illustration

$$G(t_i) = P\{t_w^1 - t_w^2 > y\} = \int_y^{+\infty} g(x)dx = \begin{cases} \frac{h_2 e^{h_1(-y)}}{h_1+h_2} & y \geq 0 \\ 1 - \frac{h_1 e^{h_2 y}}{h_1+h_2} & y < 0 \end{cases} \tag{19}$$

Obviously, $t_i(e_1, e_2) \geq 0$, so given t_i, the processing disordering ratio of e_1, e_2 denoted as $\omega(e_1, e_2)$ can be expressed as follows,

$$\omega(e_1, e_2) = G(t_i) = \frac{h_2 e^{h_1(-t_i)}}{h_1 + h_2} \tag{20}$$

Assuming the arrival time interval of any two matched events follows the PDF $f(x)$, which is related to whether the partition operation is involved.

For the query involves the partition operation, we can usually get some statistical information about $f(x)$ according to the distance between any two logic areas. For example, $d(i, j)$ represents the distance between the logic area R_i and R_j, the average speed for a monitored object to move from R_i to R_j is $v(i, j)$. We can consider $f(x) = d(i, j)/v(i, j)$.

For the query without any partition operation, $f(x)$ has the relationship with the arrival situation and different modes may lead to different results. We will analyze some specific consumption modes as follows. Other arrival distributions and processing models can be inferred in the similar way.

For the *poisson stream*, the arrival time interval of E_1 and E_2 follows the negative exponential distribution with PDF $f_{i1}(x)$ and $f_{i2}(x)$. For the recent mode, the E_1 to match will be the closest one to E_2. As illustrated in figure 3(a), e_1 is matched by e_2 and $f(x)$ is just the pdf of $e_2.t_a - e_i.t_a$.

ACCEPTED MANUSCRIPT

Supposing y represents the time interval between e_1 and the next arrival event of the type E_1, and z represents the time interval between e_1 and the next arrival event of the type E_2. We can infer y follows the distribution of $f_{i1}(y)$. Due to the memoryless property of the negative exponential distribution, we can infer z follows the distribution of $f_{i2}(z)$. As illustrated in figure 3(a), when $z < t$, e_1 will participate the matching and $z = x$. Therefore, we can infer the distributions of x as follows,

$$f(x) = \int_0^x f_{i1}(y) f_{i2}(x) dy = \lambda_1 \lambda_2 e^{-\lambda_2 x} \int_0^x e^{-\lambda_1} y dy = \lambda_2 e^{-\lambda_2 x}(1 - e^{-\lambda_1 x}) \quad (21)$$

Specially, when the rates of two arrival events differ in a relative large scale, e.g., $\lambda_1 > \lambda_2$ illustrated in figure3(b), the time interval to be computed is equal to a. Because the average arrival interval of E_1 events is λ_1 and E_2 will appear between two consecutive E_1 events randomly, the average interval between some E_2 event and the foregoing E_1 event is $1/2\lambda_1$. Therefore, we can estimate $f(x)$ as the negative exponential distribution with the expectation $1/2\lambda_1$.

For the accumulative mode, the current E_1 events will match E_2 event and the matched E_1 event will continue to match other E_2 events. Because all the E_1 events come from the same event stream which will be processed in the FIFO manner, in this situation, as long as the processing disorder does not occur between the last E_1 event and the target E_2 event, the result correctness will be guaranteed. Therefore, in essence, the analysis is the same with the situation of the recent mode.

For the first mode, the first arrival E_1 event will match the E_2 event, so the time interval between these two events is equal to b illustrated in figure 3(b). Note that the average size is $1/2\lambda_1$ and the average size of $b + c$ is $1/\lambda_2$, so b can be considered to follow the negative exponential distribution with the expectation $1/\lambda_2 - 1/2\lambda_1$.

For the chronicle mode, if the average rate of E_2 is slower than E_1, the interval of the matched events will be larger and larger, x will not follow any distribution and the processing disorder will not occur. Otherwise, as illustrated in figure 3(c), the number of E_1 events which arrive between two consecutive E_2 events, affects the interval of matched E_1 event and E_2 event. We can infer the probability that k events arrive between two consecutive E_2 events as follows,

$$P_k = \iiint_{\delta} fi_1(x_1) fi_1(x_2) ... fi_1(x_k) fi_2(y) dv = \iiint_{\delta} fi_2(y) \sum_{i=1}^{k} fi_1(x_i) dv \quad (22)$$

ACCEPTED MANUSCRIPT

where δ is the hypercube composed of $x_1, x_2, ..., x_k, y$ under the condition $\sum_{i=1}^{k} x_i < y$. Furthermore, we can infer the mathematics expectation Ep_k as,

$$Ep_k = \iiint_{\delta} \sum_{i=1}^{k} fi_1(x_i) \sum_{i=1}^{k} fi_2(y_i) \sum_{i=1}^{k} d_i dv \tag{23}$$

where d_i is the interval between the corresponding E_{1k} event and E_{2k} event.

Therefore, for the chronicle mode, the average interval can be expressed as $Eti = \sum_{i=1}^{\inf} Ep_k P_k$. However, $Eti = \sum_{i=1}^{\inf} Ep_k P_k$ is difficult to be computed. Based on another important observation, because E_2 events arrive faster than E_1 events, k has a quite low probability to be a large number. If we estimate $k = 1$, the situation can be concluded into the recent mode when E_2 events arrive faster.

Another important issue is the effect of the sliding window. If the window only deletes some unmatched events with no impact on the final results, the conclusion discussed above still holds. Otherwise, given a sliding window size w, we can infer the disorder ratio ω_w as follows,

$$\omega_w = \int_{w}^{\infty} f(x)dx$$

Especially, for the accumulative mode, $f(x)$ is computed according to the recent mode. But with the sliding window, when the first event is expired, there will be the processing disorder, so $f(x)$ should be changed to be computed following the first mode in this case.

Other constraint operations can be reflected by the integral condition. Given $f(x)$, we can infer the average processing disordering ratio denoted as ω_{avg} as follows,

$$\omega_{avg}(E_1, E_2) = \int_{0}^{\infty} G(x)f(x)dx \tag{24}$$

4.3 Feedback Modifying Model

Because $\omega_{avg}(e_1, e_2)$ is related to μ_1 and μ_2, we can denote $\omega_{avg}(\mu_1, \mu_2)$ to represent $\omega_{avg}(e_1, e_2)$. If the event is processed in disorder, we will not get correct results. Hence, $1 - \omega_{avg}(\mu_1, \mu_2)$ implicates the correctness ratio of the detected composite events. Accordingly, $\theta_\sigma(\mu_2)(1-\omega_{avg}(\mu_1, \mu_2))$ represents the correct results detected under the deadline constraint. Assuming the number

ACCEPTED MANUSCRIPT

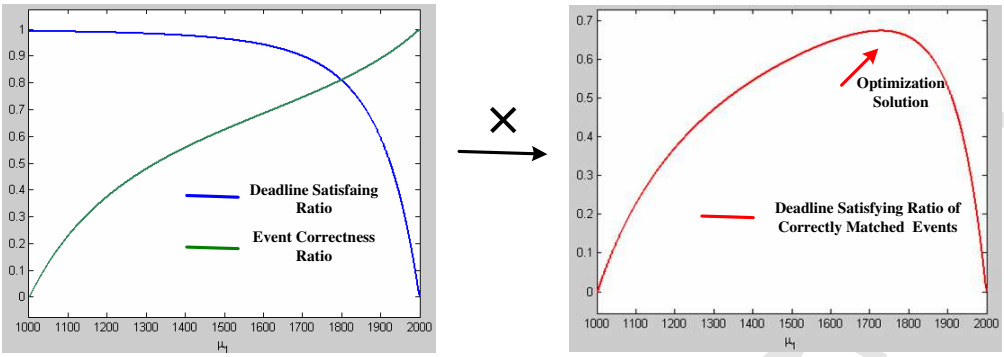


Fig. 4. The function curves for satisfying ratio and correctness ratio ($\lambda_1 = \lambda_2 = 1000, \mu = 3000, t_d = 1ms$)

of E_1 to be processed in the unit time using the total processing resource available is μ, and $C_2/C_1 = k$, the feedback correction model can be expressed as the following target function in equation . The calculated μ_1 and μ_2 can be used to define the time-slice allocated to E_1 and E_2 in each unit time.

$$\begin{aligned} & \max \quad [\theta_\sigma(\mu_2)(1-\omega_{avg}(\mu_1,\mu_2))] \\ s.t. & \begin{cases} \mu_1 > \lambda_1, \mu_2 > \lambda_2 \\ \mu_1 + k\mu_2 \leq \mu \end{cases} \end{aligned} \tag{25}$$

As illustrated in figure 4, as μ_2 increases, the deadline satisfying ratio $\theta_\sigma(\mu_2)$ increases. But because μ_1 will be decreased correspondingly, leading to the increases of t_w^1 and $\omega_{avg}(\mu_1, \mu_2)$. Therefore, $\theta_\sigma(\mu_2)$ and $1 - \omega_{avg}(\mu_1, \mu_2)$ have the trade-off relationship. The product will reach peak when μ_2 is allocated some value between λ_2 and μ, which implicates generating most results under the deadline constraint with the allocated resource.

5 Deadline-aware Processing Model for Multiple Streams

5.1 Processing Disordering Model for Multiple Streams

In general situations, multiple event streams will be involved. Supposing the event stream number is n, the unified probabilistic density function of all the events' waiting time is $\prod_{i=1}^{n} t_w^i(x)$. Hence, we can prove the disordering ratio in this case as follows,

$$\omega(E_1, E_2, \cdots, E_n) = 1 - \iiint_{\Omega} \prod_{i=1}^{n} w_t^i(x_i) dv \tag{26}$$

ACCEPTED MANUSCRIPT

And the integral region is in a n-dimensional space where $\Omega = \bigwedge_i^{n-1}(x_i < x_{i+1} + t_{i,i+1})$. Furthermore, we can transform equation 26 as follows,

$$\omega(E_1, E_2, \cdots, E_n) = 1 - \int_0^{\infty} t_w^n n(x_n) dx_n \int_0^{x_n + t_{n-1,n}} t_w^{n-1}(x_{n-1}) dx_{n-1} \cdots \int_0^{x_2 + t_{1,2}} t_w^1(x_1) dx_1 \tag{27}$$

Also, we need to obtain the average disordering ratio for our feedback modifying. Assuming $f_{i,i+1}(x)$ is the probabilistic density function of the arrival time interval between e_i and e_{i+1}, the average disordering ratio can be inferred by equation 28,

$$\omega_{avg}(E_1, E_2, \cdots, E_n) = 1 - \iiint_{\Omega} \omega(E_1, E_2, \cdots, E_n) \prod_{i=1}^{n-1} f_{i,i+1}(x) dv \tag{28}$$

Similar to the $f(x)$ analysis of the binary event stream processing, $f_{i,i+1}(x)$ can be estimated. To simplify the equation expressions, without the loss of generality, we suppose the processing cost of different event streams is the same, i.e., $\sum_{1 \le i \le n} \mu_i = \mu$. The general situation and the corresponding normalization process will be further illustrated in the next subsection.

In order to obtain the smallest disordering ratio, we propose the following theorem.

Theorem 3 *Assuming the total available resource is μ, the allocation schema μ_i to make the average disordering ratio $\omega_{avg}(\mu_1, \mu_2, ..., \mu_n)$ minimum is the solution of the following equation.*

$$\begin{cases} \frac{\partial}{\partial \mu_1} \omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n) + t = 0 \\ \frac{\partial}{\partial \mu_2} \omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n) + t = 0 \\ ... \\ \frac{\partial}{\partial \mu_n} \omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n) + t = 0 \\ \mu_1 + \mu_2 + ... + \mu_n = \mu \end{cases}$$

where $\frac{\partial}{\partial \mu_i} \omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n)$ represents the partial differential coefficient and t is an assistant unknown number.

ACCEPTED MANUSCRIPT

Proof. In essence, the problem can be transformed as follows,

$$s.t.\begin{cases} \min & [\omega_{avg}(\mu_1, \mu_2, ..., \mu_n)] \\ \forall i \mu_i \geq \mu_{mini} \\ \mu_1 + \mu_2 + ... + \mu_n \leq \mu \end{cases}$$

Furthermore, the assistant function can be constructed as follows,

$$F(\mu_1, \mu_2, ..., \mu_n) = \omega_{avg}(\mu_1, \mu_2, ..., \mu_n) + t(\mu_1 + \mu_2 + ... + \mu_n - \mu)$$

The *standing points* of $F(\mu_1, \mu_2, ..., \mu_n)$ can be defined as follows,

$$\begin{cases} \frac{\partial}{\partial \mu_1}\omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n) + t\frac{\partial}{\partial \mu_1}(\mu_1 + \mu_2 + ... + \mu_n - \mu) = 0 \\ \frac{\partial}{\partial \mu_2}\omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n) + t\frac{\partial}{\partial \mu_2}(\mu_1 + \mu_2 + ... + \mu_n - \mu) = 0 \\ ... \\ \frac{\partial}{\partial \mu_n}\omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n) + t\frac{\partial}{\partial \mu_n}(\mu_1 + \mu_2 + ... + \mu_n - \mu) = 0 \end{cases}$$

When $\mu_n = \mu_{minn}$, we can get,

$$\begin{cases} \frac{\partial}{\partial \mu_1}\omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n) + t = 0 \\ \frac{\partial}{\partial \mu_2}\omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n) + t = 0 \\ ... \\ \frac{\partial}{\partial \mu_n}\omega_{avg}(\mu_1, \mu_2, \cdots, \mu_n) + t = 0 \\ \mu_1 + \mu_2 + ... + \mu_n = \mu \\ \mu_n = \mu_{minn} \end{cases}$$

The solution of theorem 3 can be computed using the MATLAB optimization tool package.

Theorem 4 *When the processing disorder ratio is low, and the consecutive disorder ratio can be taken as independent, we can give the following estimation equation for the average disorder ratio.*

$$w_{avg}(E_1, E_2, \cdots, E_n) \approx 1 - \prod_{i=1}^{n-1}(1 - w_{avg}(E_i, E_{i+1}))$$

ACCEPTED MANUSCRIPT

Table 2
Resource allocation case for minimizing the disorder ratio

	μ_1	μ_2	μ_3	μ_4	ω_{avg}	$\bar{\omega}_{avg}$
$SEQ(E_1, E_2)$	4800	1200			0.097	0.097
$SEQ(E_1, E_2, E_3)$	2997	1803	1200		0.414	0.383
$SEQ(E_1, E_2, E_3, E_4)$	1852	1612	1336	1220	0.745	0.665

Proof

$$w_{avg}(E_1, E_2, \cdots, E_n) = P\{Disorder(E_1, E_2, \cdots, E_n)\}$$
$$= 1 - P\{!Disorder(E_1, E_2, \cdots, E_n)\} = 1 - P\{!Disorder(E_1, E_2)$$
$$\wedge !Disorder(E_2, E_3) \wedge ... \wedge !Disorder(E_{n-1}, E_n)\}$$

If there are no identical events in $Disorder(E_i, E_{i+1})$ and $Disorder(E_j, E_{j+1})$, they are independent with each other. Furthermore, we can infer $!Disorder(E_i, E_{i+1})$ and $!Disorder(E_j, E_{j+1})$ are independent. Therefore, we can get,

$$w_{avg}(E_1, E_2, \cdots, E_n) \approx 1 - P\{!Disorder(E_1, E_2)\}P\{!Disorder(E_2, E_3)\}$$

$$\cdots P\{!Disorder(E_{n-1}, E_n)\} = 1 - \prod_{i=1}^{n-1}(1 - w_{avg}(E_i, E_{i+1})$$

. Hence, the theorem is proved.

Note that as the number of the event type becomes larger, the error ratio of the estimation based on the theorem 4 will become larger. In this case, we have to utilize theorem 3 for the model solution. In table 2, we give an example to analyze the computed disorder ratio under the proposed minimum disorder model, where ω_{avg} denotes the mathematical solution based on the theorem 3 and $\bar{\omega}_{avg}$ denotes the estimation solution based on the theorem 4. The comparison between ω_{avg} and $\bar{\omega}_{avg}$ can be observed in this example.

5.2 Time-slice Allocation Algorithm

Obviously, the solution of the following optimization model is just the core idea of our asymmetric time-slice allocation framework.

$$\max \quad [\theta_\sigma(\mu_n)(1 - w_{avg}(\mu_1, \cdots, \mu_n))] \tag{29}$$
$$s.t. \begin{cases} \forall i \quad \mu_i > \lambda_i \\ \sum_i \mu_i \leq \mu \end{cases}$$

In essence, our proposed model is preemptive time-sliced scheduling. Especially, due to the stable processing cost for each event stream, the fixed loop control can be utilized instead of the thread switch to avoid extra scheduling costs. Because our target is to maximize the deadline-satisfying ratio, the

ACCEPTED MANUSCRIPT

Algorithm: The time-sliced allocation scheduling algorithm
Input: the number of CPU instructions per second κ
the number of instructions to process each event over some event streams $Ins(e_i)$
average rate of some event stream λ_i
time allocated for other tasks per unit time $t^{'}_{cycle}$
delay upperbound t_d
Output: allocated processing time for each event stream per unit time t^i_{cycle}

1. $t^i_p \leftarrow 1/\lambda_i, t^n_{cycle} \leftarrow 1 - t^{'}_{cycle} - \sum_{i=1}^{n-1} Ins(e_i)/(t^i_p\kappa)$
2. $t^n_p \leftarrow Ins(e_n)/(t^n_{cycle}\kappa)$
3. **if** $t^n_p < 1/\lambda_n$ **return** error
4. $\gamma \leftarrow \frac{(t^n_p - 1/\lambda_n)}{\Gamma}$
5. **while** $(t^n_p > 1/\lambda_n)$
6. $\quad t^n_p \leftarrow t^n_p - \gamma$
7. $\quad$ Compute $\theta_\sigma(\mu_n)$ according to equation 12
8. $\quad$ Compute $\min(\omega_{avg})$ and $\mu_1, \cdots, \mu_{n-1}$ according to theorem 3 or theorem 4;
9. $\quad$ **if** $\max(\theta_\sigma(1 - \min(w_{avg}))$=true **then**
10. $\quad\quad$ **return** $t^n_{cycle} \leftarrow Ins(e_n)/(t^n_p\kappa)$, $t^i_{cycle} \leftarrow Ins(e_i)\mu_i/(\kappa)(1 \le i < n)$;
11. $\quad$ **end if**
12. **end while**

Fig. 5. Algorithm for the time-slice allocation scheduling

proposed model obviously outperforms FIFO scheduling strategies. Also, the features of the complex event processing makes the EDF scheduling inefficient. Because only the deadline of the composite event which is determined by the termination event can be given, if all the primitive events are bounded by this deadline, EDF turns out into FIFO with extra costs. In addition, our time-sliced model can be efficient for the distributed and parallel processing environment.

Because different event streams may incur different processing costs, the normalization should be conducted before the resource allocation. If the processed number per unit time over each event streams, denoted as $\mu(1), \cdots, \mu(n)$ can be tested, the normalization can be conducted according to the proportion. For the scenarios where on line test can not be finished, given the instruction number $Ins(e_i)$ for some event type e_i and supposing κ instructions can be executed per unit time, the processing time for e_i can be calculated as $T_i = Ins(e_i)/\kappa$. Therefore, if t^i_{cycle} represents the time-sliced cycle for event stream S_i, and some other system tasks will occupy $t^{'}_{cycle}$ CPU resource, i.e. $\sum_{i=1}^{n} t^i_{cycle} + t^{'}_{cycle} = 1$, we can infer $t^i_p = T_i/t^i_{cycle}$, $t^i_{cycle} = T_i/t^i_p$, $t^i_{cycle} = Ins(e_i)/(t^i_p\kappa)$ where $t^i_p = 1/\mu_i$. In conclusion, we give the time-sliced allocation scheduling algorithm as follows.

ACCEPTED MANUSCRIPT

6 Experimental Evaluation

Because RFID is the most commonly used and meaningful scenarios for the complex event processing [1]13, our evaluation is first based on an RFID monitored large-scale scenarios. However, a large deployment of RFID device is quite expensive due to the limitation of the space and device. More important, we usually need even thousands of tagged objects in the monitoring scenarios to test the algorithms' efficiency. So, available papers usually use a simulated RFID data set according to different goals based on their own generators. In our paper, we design a reasonable RFID monitoring scenarios in a smart museum. The RFID readers are deployed at different exhibition areas and different readers will produce different event streams when tagged visitors come into the sensing range. In this way, the primitive event can be modeled as ¡objectID, readerID, timestamp¿. For a reader with ID=i, the events satisfying E_i will be produced. In the long term, the arrival of the visitors can be considered to follow the poisson process. In the paper [1], some query semantics of similar scenarios are proposed based on the SEQ operator. We generate the data based on our simulation platform RFIDSim which is designed for effectively simulating real-life RFID applications. In fact, only two factors need to be considered by our tests. One is related to the velocity(v or λ), which can be flexibly controlled and adjusted by the simulator. The other is related to the resource allocation(μ) which can be efficiently adjusted by controlling loop times which is illustrated in our algorithm of section 5.2. Note that all the processing process is conducted by our independent program which just takes the simulated streaming data as the input. Also, in order to test the feasibility and adaptability of our framework, real-life data sets about web server access records are further utilized in section 6.5. We will separately test the major models and algorithms proposed in the paper. All the experiments were run on a 2.4 GHz Pentium 4 CPU and 1 GB RAM, and implemented in C. Also, Matlab is employed for some model solution computation.

6.1 Processing Cost Model Tests

In this test, we evaluate the event processing cost under different consumption modes. For the unrestricted mode, as illustrated in figure 6 which separately displays composite event number with the variations of the arrival rate and the lasting time, the estimated results based on the theorem 2 can obtain very high correctness. This test proves the correctness of our inference about the cost of the unrestricted mode. According to the analysis of section 3.2.3, the tests also implicate the same conclusion for the continuous mode given $t = 1/E(v_3)$ because it is a variant of the unrestricted mode. Because other tasks may be processed at the same time, we may prefer to occupy part of the processing

ACCEPTED MANUSCRIPT

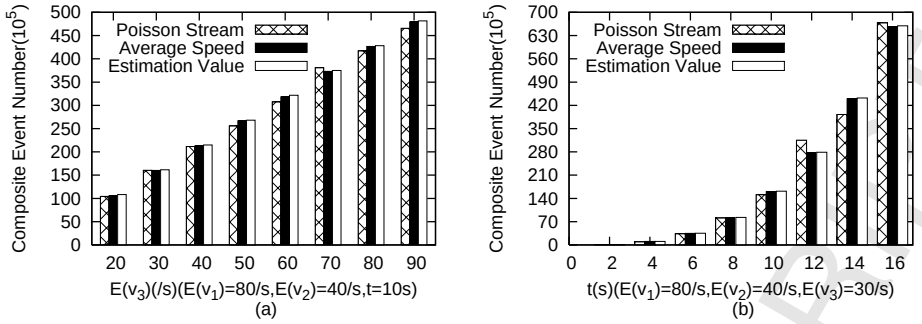


Fig. 6. The test for event estimation under unrestricted mode

resource. For the recent mode under the query $t_p^1 \approx t_p^2 \approx t_p^3$, when the CPU allocation reaches six percent, the test results show $t_p^1 \approx t_p^2 \approx t_p^3$ and $\mu \approx 3000$. It illustrates the processing cost of the recent mode is stable and only limited processing capability is needed to gain high efficiency. Similarly, in the test, we can observe the stable cost can be obtained for the first, cumulative and chronicle mode, which is accordant with the analysis in table 1. Because the cost model is the foundation of our satisfying model, this test offers a good preliminary for the further evaluation. In the next tests, we only display the results based on the first mode which is widely used in the RFID monitoring. μ can be adjusted by changing the resource allocated for the CEP task. Without the loss of generality, the test results can illustrate other consumption modes and more complex queries.

6.2 Termination Event Deadline Satisfying Ratio Tests

In this test, we verify the deadline satisfying model concluded in equation 12, which dose not consider the potential processing disorder. Because the composite event deadline satisfying ratio is only determined by the termination event satisfying ratio, the number of the involved event types has no impact on the model. As an example, we set the query as $SEQ(E_1, E_2)$. By adjusting the value of $t_p^2 = 1/\mu_2$, t_d and $1/\lambda_2$, the tested deadline satisfying ratio and the theoretical value are compared. As illustrated in figure 7, the deviation between the theoretical value and the practical value is minor which is acceptable to serve the resource allocation model. In figure7(a),as t_p^2 increases, μ_2 will decrease and thus the deadline satisfying ratio can be observed to decrease. But a decreased μ_2 may cause a higher processing disorder, which leads to the optimal allocation problem of this paper. In figure7(b), the deadline satisfying ratio increases with the increase of the delay upper bound. That is because more events can be detected before a looser deadline. The effects of the arrival rate on the deadline satisfying ratio is further given in figure7(c). A slower arrival rate(λ) will ensure more results output under the deadline using given resource. Also, as a metric to evaluate the soft real-time feature,

ACCEPTED MANUSCRIPT

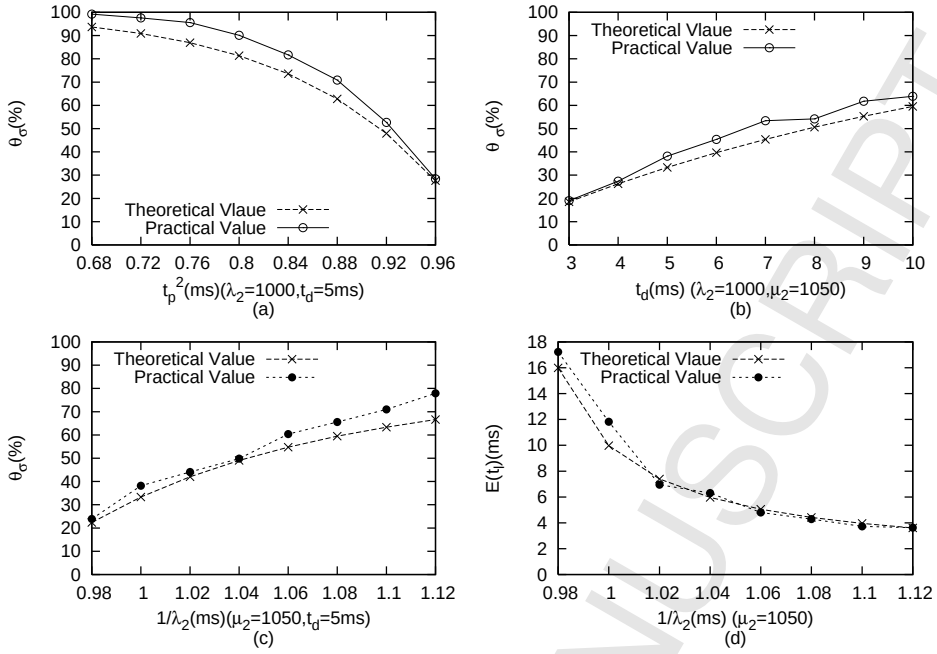


Fig. 7. The test for the deadline satisfying ratio

we test the average delay estimation illustrated in figure7(d). We can observe the relatively accurate results can be gained.

6.3 Processing Disorder Tests

In this test, the processing disorder model is verified. As illustrated in figure 8(a), for the query $SEQ(E_1, E_2)$. The deviation between the theoretical value and the practical value is minor. As μ_2 increases, μ_1 will decrease and the processing disorder ratio will increase. Considering the deadline satisfying ratio evaluated in section 6.2, the trade-off of the resource allocation is important to output more correct and real-time results. To further evaluate the impact of more event types, we use $SEQ(E_1, E_2, E_3)$ in figure8(b). In such multiple event streams case, the theoretical mathematical value based on the theorem 3 and the estimation value based on the theorem 4 both gain satisfactory accuracy. Especially, the theoretical mathematical value outperforms the estimation value in accuracy which is analyzed in section5.2. But the estimation value can be more efficiently to apply in some scenarios. In the test, μ_3 is randomly set as 1100, with the increase of μ_2, μ_1 will decrease. The theoretical minimum processing disorder will occur when $\mu_2 = 1259$ which is basically accordant with the test results in figure8(b). Therefore, this set of evaluation can prove the effectiveness of our processing disorder model for binary stream sources and multiple stream sources.

ACCEPTED MANUSCRIPT

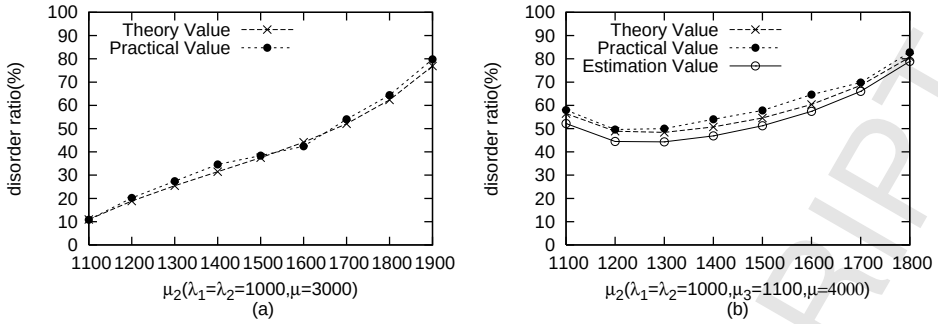


Fig. 8. The test for the processing disorder

6.4 Optimization Resource Allocation Model Tests

In this test, we verify the proposed optimization resource allocation model based on the disorder feedback. Theory-RF and Theory-NRF separately represent the satisfying ratios with and without the processing disorder feedback. As illustrated in figure 9(a), for $SEQ(A, B)$, with the increase of μ_2, Theory-RF is the reasonable estimation for the practical value, and the calculated resource allocation $\mu_2 = 1271$ is accordant with the test result. However, for Theory-NRF, the estimation is quite inaccurate because a large amount of processing disorders are ignored. The tests for $SEQ(A, B, C)$ are illustrated in figure9(b). By setting $\mu_3 = 1100$, the computed optimization solution is $\mu_2 = 1284$ which is also accordant with the test results. Note that Theory-NRF is not changed because it is only determined by the resource allocation of the termination event, namely μ_3 in this case. In figure9(c), the optimization solution following our resource allocation model is illustrated. Also, the deviation between the theoretical value and the practical value is minor. With the increase of μ, the satisfying ratio will increase until reaching one, because if enough processing resource is available, all the events can be processed before the deadline and in the correct sequence. Finally, we compare our proposed resource allocation model based on the round-robin with other scheduling models in figure9(d). The results prove the round-robin outperforms FIFO and EDF in CEP deadline satisfying scenarios. Because EDF will incur extra computation costs compared to FIFO, it dose not display satisfactory performance in our test. We have analyzed the detailed reason in section 5.2

6.5 Tests of Real Data Sets

In this test, we randomly choose two groups of web server access data sets available from http://ita.ee.lbl.gov. One data set named NASA-HTTP contains the HTTP requests to the NASA Kennedy Space Center WWW server in Florida and the other named Saskatchewan-HTTP contains the HTTP requests to the

ACCEPTED MANUSCRIPT

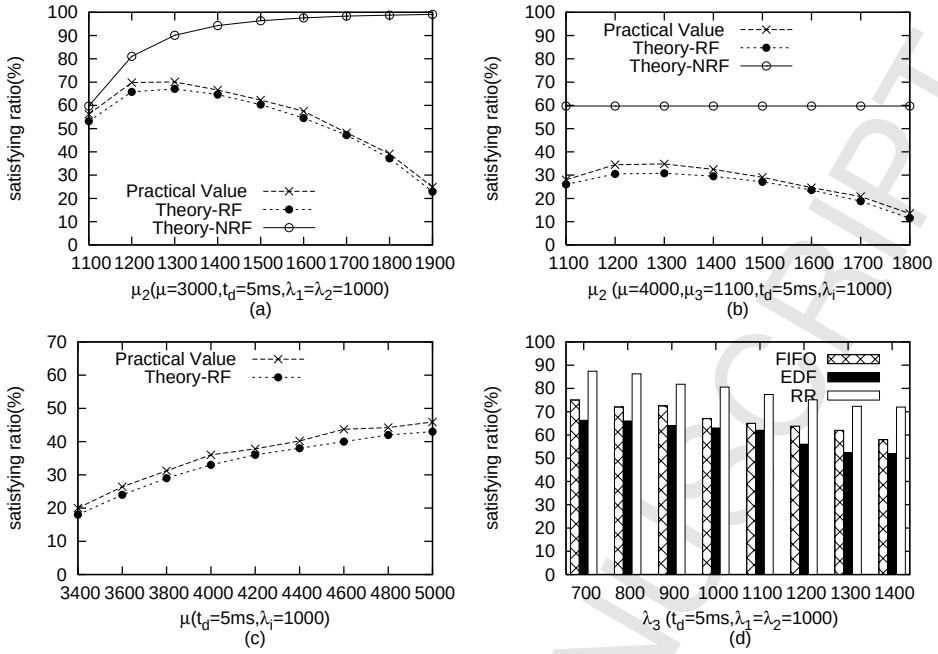


Fig. 9. The test for the optimization resource allocation model

University of Saskatchewan's WWW server in Saskatoon, Canada. The data format contains the information of host, timestamp, request, HTTP reply code and bytes in the reply. Each data set can be considered as a distributed event source and allocated an event type. SEQ query can detect the sequential correlation pattern of the two servers and offer the support for further analysis and mining. The data are loaded in the streaming fashion and feed our programs as the input. The test results of SEQ query are illustrated in figure 10. In order to reflect the essence of our resource allocation model, we choose to display the effect of the allocated resource on the deadline satisfying model, processing disorder model,feedback modifying model and various scheduling strategies.

In figure 10(a), we illustrate the effect of μ_2 on the deadline satisfying ratio. With the more resource allocated to the termination event, the deadline satisfying ratio is increased and the practical results are satisfactory compared to the theory values. In the same way, by adjusting μ_2 in figure 10(b), we can observe a higher μ_2 which means a lower μ_1 will lead to a higher processing disorder ratio. Also, the tested value can verify the theoretical model. By combining the above tests, we test the feedback modifying model which outputs the correct results under the deadline constraint in figure 10(c). Obviously, a proper allocation between μ_1 and μ_2 will gain the optimal performance for the final results, which reflects the trade-off between finishing more composite events before the deadline and decreasing the processing disorder. We can observe that the result is generally accordant with our model illustrated in equation(25). Finally, we compare various scheduling strategies by varying μ_2

ACCEPTED MANUSCRIPT

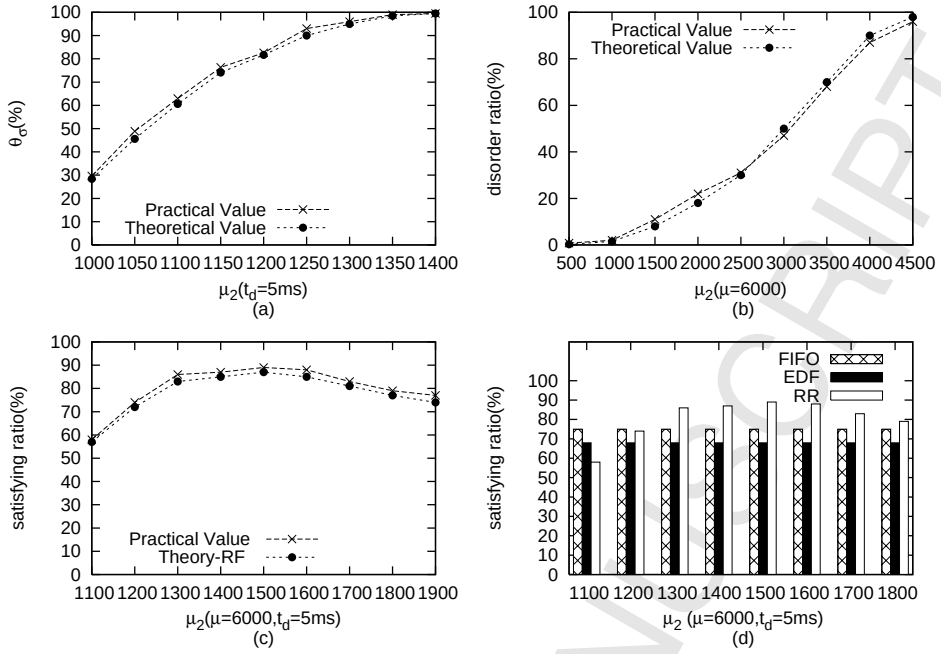


Fig. 10. The test for the server access data

in figure10(d). When an optimal μ_2 is determined which is adopted by our algorithm, RR can outperform FIFO and EDF obviously. Note that FIFO and EDF scheduling have no internal relationship with μ_2 because no resource allocation decision is needed for them.

7 Conclusion

The emergence of various sensing and monitoring systems has enabled applications to execute complex business logic and the functionality to detect and respond to complex events in real-time is desirable. Faced with the event streams containing the rich semantics and the upper bound requirements, for the complex event processing scenarios, how to obtain more composite events before the deadline is a very key problem. In this paper, we build an integrated framework including the arrival model, processing model, cost model, termination-event deadline satisfying model, processing disorder model and the proposed composite event deadline satisfying optimization model. Compared to other scheduling methods, our proposed model can maximize the correct query results under the real-time constraints.

Acknowledgement

Supported by the National Science Foundation of China under Grant (Nos. 61003058,60933001).

ACCEPTED MANUSCRIPT

References

[1] E. Wu, Y. Diao, S. Rizvi, High-performance complex event processing over streams, Proc. of SIGMOD, Chicago, USA, 2006.

[2] F. T. J. Chen, D. DeWitt and Y. Wang, Niagaracq: A scalable continuous query system for internet databases, Proc. of SIGMOD, Dallas, Texas, USA, 2000.

[3] R. Motwani, J. Widom, A. Arasu, et al, Query processing, resource management, and approximation in a data stream management system, Proc of CIDR, Asilomar, CA, USA, 2003.

[4] S. Chandrasekaran, O. Cooper, A. Deshpande, et al, TelegraphCQ: Continuous dataflow processing for an uncertain world, Proc. of CIDR, Asilomar, CA, USA, 2003.

[5] D. Abadi, D. Carney, U. Cetintemel, et al, Aurora: A new model and architecture for data stream management, Journal of VLDB, 2003,12(2):120-139.

[6] D. Carney, U. Cetintemel, M. Cherniack, Monitoring streams: A new class of data management applications, Proc. of VLDB, Hongkong , China, 2002.

[7] T.M. Sutherland, B. Pielech, Y. Zhu, et al, An adaptive multi-objective scheduling selection framework for continuous query processing, Proc. of the 9th Int'l Database Engineering and Applications Symp. Montreal, Canada, 2005.

[8] S. Schmidt, H. Berthold, W.Lehner, Qstream: Deterministic querying of data streams, Proc. of VLDB, Trondheim, Norway, 2005.

[9] S. Schmidt, T. Legler, S. Schar, W. Lehner, Robust real-time query processing with Qstream, Proc. of VLDB, Toronto, Canada, 2004.

[10] J. Wu, K.L. Tan, Y. Zhou, QoS-Oriented Multi-query Scheduling over Data Streams, Proc. of DASFAA, Brisbane, Australia, 2009.

[11] D. Gyllstrom, E. Wu, H. J. Chae, et al, SASE: complex event processing over streams, Proc. of CIDR, Asilomar, CA, USA, 2007.

[12] L. Brenna, A. Demers, J. Gehrke, et al, Cayuga: A high-performance event processing engine, Proc. of SIGMOD, Beijing, China, 2007.

[13] A. Demers, J. Gehrke, B. Panda, et al, Cayuga: a general purpose event monitoring system, Proc. of CIDR 2007, Asilomar, CA, USA, 2007.

[14] Q. Chen, Z.H. Li, H.L. Liu, Optimizing complex event processing over RFID Data Streams, Proc. of ICDE, Cancun, Mexico, 2008.

[15] F.S. Wang, S.R. Liu, P.Y.Liu, et al, Bridge physical and virtual worlds: complex event processing for RFID data streams, EDBT, Munich, Germany, 2006.

[16] M. Akdere, U. Cetintemel, N. Tatbul, Plan-based complex event detection across distributed sources, Proc. of VLDB, Auckland, New Zealand, 2008.

ACCEPTED MANUSCRIPT

[17] V. Garg, Estream: an integration of event and stream processing. Master Thesis, University of Texas at Arlington, Dec. 2005.

[18] G. Cybenko G, V.H. Berk, Process query system, IEEE Computer, 2007, 40(1):62-70.

[19] E. Welbourne, N. Khoussainova, J. Letchner, et al, Cascadia: a system for specifying, detecting, and managing RFID events, Proc. of Mobisys., Breckenridge, CO, USA , 2008.

[20] C.Re, J. Letchner, M. Balazinska, et al, Event queries on correlated probabilistic streams, Proc. of SIGMOD, Vancouver, BC, Canada, 2008.

[21] M. Liu, E.A. RunRundensteiner, K. Greenfield, et al, E-Cube: Multi-Dimensional Event Sequence Processing Using Concept and Pattern Hierarchies, ICDE, Shanghai, China, 2010.

[22] D. Wang, E.A. Rundensteiner, R.T.Ellison, Active Complex Event Processing over Event Streams. PVLDB, 2011, 4(100): 634-645,.

[23] S.Chakravarthy, V.Krishnaprasad, E.Anwar, et al, Composite events for active database: semantics, contexts and detection, Proc. of VLDB, Santiago, Chile, 1994.

[24] R.B.Cooper, Introduction to queueing theory, 2nd edition. London: Edward Arnold, 1981.

ACCEPTED MANUSCRIPT