The 5th Annual IEEE International Conference on
Cyber Technology in Automation, Control and Intelligent Systems
June 8-12, 2015, Shenyang, China

Study on Distributed Complex Event Processing in Internet of Things based on Query Plan*

Lingyun Yuan, Dongdong Xu, Guili Ge, Mingli Zhu
College of Information Science and Technology
Yunnan Normal University
Kunming, China
blues520@sina.com, xudongdong.ynnuedu@163.com

***Abstract*—Complex event processing is an efficient method in data stream processing of Internet of things, but more of these methods are referred to a single complex event or a small quantity of events. Aiming at this problem, a distributed complex event processing architecture for Internet of things is presented in this paper, in which a distributed query plan of complex event process structure based on directed acyclic graph (DAG) is given, moreover, a distributed query-plan complex-event-processing algorithm based on directed acyclic graph is proposed. The complex tasks are decomposed into several simple sub-tasks which are processed in parallel with the corresponding operator nodes, to realize distributed processing and to improve the efficiency of processing and execution. The simulation results indicate that our method is more efficient in lower RAM consumption, processing time, and others, and the efficiency of data stream processing for Internet of things is improved.**

Keywords—Internet of things; complex event processing; query plan; distributed parallel processing

I. INTRODUCTION

As a new generation network technology, IOT which is known as the third wave of information after the computer and the Internet, has been widely admired by people from all walks of life and is gradually applied to health care, intelligent transportation, logistics transportation and other fields [1].The IOT data are always uncertain, magnanimous and heterogeneous, so how to quickly find the required data [2] and how to excavate the implicit internal relations among data and how to extract and abstract meaningful events have been a matter that needs to be studied urgently.

As one of the IOT technologies, complex event processing (CEP) technology has been widely used in all kinds of fields. Researchers have won some valuable achievements in the field. The most typical complex event processing system is SASE [3], which is proposed by Eugene from University of California, Berkeley. It is a query plan-based complex event detection technology that can share event operator according to the modeling characteristics of query plan. But SASE uses active instance stack (AIS) to store intermediate results, causing a large number of backtracking during detection. Aiming at the problem, WANG [4] proposes a distributed CEP architecture and a high-performance CEP method (HPCEP) based on SASE. However, other problems present that the use of large sliding windows for the constraint of data out of date needs to traverse in each AIS and the time cost is high.

In order to realize queries to complex events and processing of queries, the research in WANG [5] has made some extension based on Petri nets. But it only directs at studying query processing and optimization technology of single complex event and lack of researches on multiple complex ones.

SONG [6] proposes an extended sub-query sharing approach, which realizes query sharing of multiple RFID complex events by analyzing the rich semantics of queries. But the approach is only applicable for the condition where the amount of complex events is small, and if the amount increases, the memory space of system would be exceeded.

To cure the above problems, a distributed complex event processing architecture for Internet of things is presented in this paper, and an architecture based on directed acyclic graph (DAG) for distributed query plan complex event processing is also described. Meanwhile, based on the above, a Distributed Query Plan Complex Event Processing Algorithm based on Directed Acyclic Graph (DQCEP) is put forward, which decomposes complex task into several simple sub-tasks, and allots them to the corresponding operator nodes to process concurrently to realize distributed processing and to improve the efficiency of processing and execution. Simulation results show that, compares with SASE, the proposed DQCEP not only can process events with high efficiency and reliability, but also can process massive and real-time IOT event streams timely and efficiently.

The remainder of this paper is organized as follows. In section II, a distributed IOT complex event processing system architecture is presented. In section III, we give a complex event processing model and algorithm description based on DAG, and in section IV simulation experiments and analysis are introduced. Section V conclusions the paper.

II. DISTRIBUTED IOT COMPLEX EVENT PROCESSING SYSTEM ARCHITECTURE

A distributed complex event processing system architecture for IOT is constructed in this section. The architecture consists of event preprocessing engine, complex event processing engine, the CEP client and the query interface, as shown in Fig.1. The event preprocessing engine is mainly in charge of preprocessing the IOT events stream,

This work is supported by the National Natural Science Foundation of China (Grant No.61262071), MOE (Ministry of Education in China) Project of Humanities and Social Sciences (Grant No.13YJCZH233), the Applied Basic Research Plan on the Project in Yunnan Province (Grant No.2013) and the Yunnan Normal University Graduate Student Scientific Research Innovation Fund Project.

including out-of-order events processing, event filtering and event classification. Complex event processing engine is composed of four modules, including CEP query interface management, CEP query scheduling engine, event processing engine and distributed CEP management engine. Its main work is to process the total order events streams processed from the out-of-order ones according to the requirements of user queries.

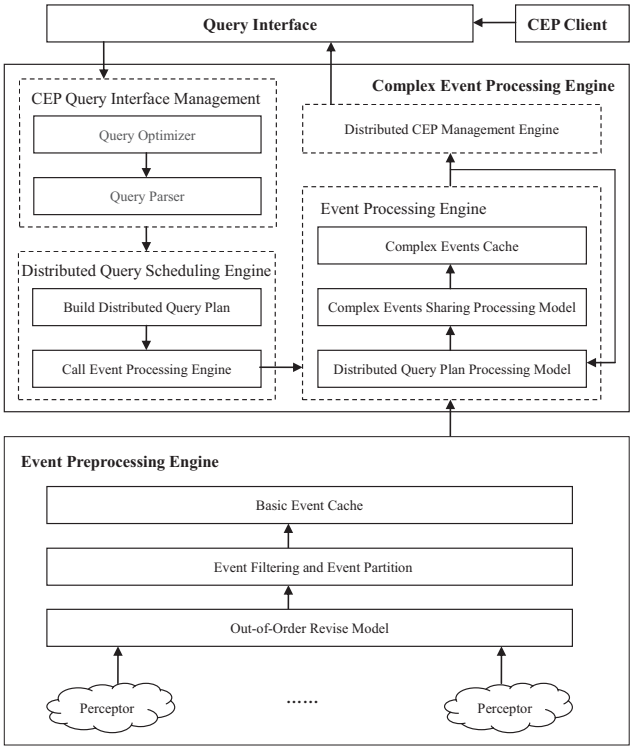

Fig.1. Distributed IOT CEP system architecture

(1) Event preprocessing engine (EPE). The module includes three parts, out-of-order event revise model (ORM), event filtering and event partition (EF&EP) and the basic event conservation (BEC). Aimed at the original event collection, it is doing simple pretreatment before entering the event processing engine, to prepare for the scheduling engine calls, while reducing the workload of events processing. The sequence of event processing engine, mainly processes in sequence of the perceptron perceptual information, to prevent event disorder phenomenon such as network latency and computer failure, and etc. The main function of event filtering and event partition is operating the filter cleaning, fusion to the event, and then use the appropriate time window perceiving information division of the perceptron, by event parallel detection to improve the performance of event processing, and reduce the response time, finally form the basic event.

(2) The CEP client and the query interface (CEPC&QI). This module is chiefly used for users to send the complex event query request through CEP client. The query interface submit query requests to the query interface management module which is responsible for processing the query requests from CEP client, including event parse and other processing.

(3) The CEP query interface management (CEPQIM). This module is mainly used for processing CEP queries from CEP client, including query optimization and query parse. Among them, the main work of the query optimization is to rewrite the query expression and the common parts of it would be reorganized, merged, shared and reused to realize the sharing of queries. Query parse mainly analyzes the users' query and parses them into atomic events, event operators and the predicate constraints among related event type according to the predefined event types and operator set.

(4) Distributed query scheduling engine (DQSE). The module is used to construct the distributed query plan and call the event processing engine module to execute query plan.

(5) The event processing engine (EPE). The module includes 3 parts: query plan based execution engine (QPBEE), complex event sharing engine (CESE) and complex event cache (CEC). Among them, QPBEE is the key part of the module and its main function is to use query plan based methods to process event streams and convert them into complex events according to the descriptions of query language and IOT data streams collected. The main function of CESE is to make simple supplementary processing to the complex events generated currently, so that it can participate in the follow-up construction of more complex events. The main function of CEC is to store events processed above for facilitating subsequent use.

(6) Distributed CEP management engine (DCEPME). The module needs to integrate the results of event processing on the whole according to the distributed query and some other processing.

III. COMPLEX EVENT PROCESSING MODEL OF DISTRIBUTED QUERY PLAN

In order to support parallel detection and processing of complex events, we use directed acyclic graph to represent the hierarchical complex events. The constructing a directed acyclic graph according to the query expression can be referred to the complex event process sharing model that we have done in [7]. On the basis of this research, a DAG approach based distributed query plan for complex event processing is proposed.

A. The Definition

The preliminary definitions of the basic events and complex events have conducted in literature [7], and this paper carries out the following extended definitions.

Definition 1: Sub-events. Sub-events are events that defined in a complex event and used to participate in compositing a complex event.

Definition 2: Sub-query. Sub-query is decomposed from the whole query and is a sub-event query to one complex event.

Definition 3: Simplest sub-query. The simplest sub-query is a sub-query that contains an event operator and two operands (basic events or complex events).

Event operators used in this paper is referred to the related literatures [7-8] and get from the practical applications in IOT,

including: AND (△), OR (△), SEQUENTIAL (SEQ/;), NOT (¬). We use the language proposed in SASE [4] as event language used in this paper.

B. The Architecture of DQCEP Based on DAG

The system architecture is shown in Fig.2. The event query expressions are split into multiple internal operator trees to represent, and each operator tree corresponds to a simplest sub-query and each simplest sub-query owns an operator. The simplest sub-queries are assigned to the corresponding operator processor to process and each operator processor corresponds to a circular processing queue.

The simplest sub-queries split are assigned to the corresponding circular processing queue to wait for executing, and the processing results will be submitted to the general processor. According to the processing conditions, the general processor decides whether the processing results are mapped to circular processing queues for further processing or output to CEP client.

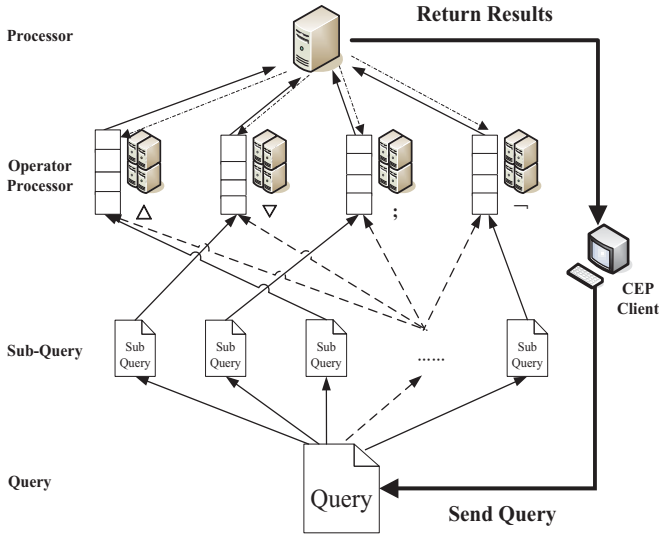

Fig.2. Architecture of DQCEP based on directed acyclic graph

C. Distributed Parallel Processing Method Based on DAG

The complex task is decomposed into several simple sub-tasks which are processed in parallel by multiple operator processing nodes, so as to realize distributed processing, and to improve processing and execution efficiency.

a. Parallel Processing Strategy

The approach that a DAG which is constructed from the query expression is split into multiple internal operator trees from bottom to top, and left to right, is proposed in [7]. Each operator tree corresponds to one simplest sub-query. The distributed query scheduling engine is responsible for assigning the simplest sub-query to the corresponding operator tree to be carried out; each of the simplest sub-queries corresponds to an operator tree. Each type of operators is dealt with a particular processor, and each operator processor has a corresponding circular processing queue. The simplest sub-query split is assigned to the circular processing queue corresponding to operator processor in accordance with the order, waiting to execute. The results processed by operator processors are sub-sets of the overall results, which can be output directly, or as intermediate results mapped to the circular processing queues for further processing.

Moreover, considering the dependence relationship between the operator of query plan and lower operand of the simplest subquery, and the simplest sub-query corresponding operator tree, namely the set membership between the lower and the upper layer and the incidence relation between the same layer operands through the upper operator, integrated optimization scheduling of processing capacity and data transmission capabilities is concern according to the coupling relationship between the sub query and nodes [8].

b. The Node Structure

In order to effectively deal with the complex events, in this paper, we use orthogonal list proposed in [6] to express the DAG which is converted from the above queries. Data assigned to the operator circular processing queue can be represented by a node which consists of six domains: nodes domain (Node), data domain, left child node domain (lchild), left value domain (lchild_data), right child node domain (rchild) and right value domain (rchild_data). Among them, 'Node' indicates the sequence number of nodes that is to be processed; 'data domain' represents the vertex information, that is, the operator; 'lchild' and 'rchild' respectively represent the sequence number of left and right child node; 'lchild_data' and 'rchild_data' respectively indicate the value delivered to the left and right child node. If there is no value in 'lchild_data' and 'rchild_data' domain, then it is empty. 'parent' indicates the sequence number of the current node's parent nodes. If the operator is '¬', it only has child domain and value domain. If the operator is ';', it needs to respectively insert an arrival time label for 'lchild' and 'rchild'. Here we only take the general operator for example.

c. The Distributed Complex Event Processing Based on Query Plan

The complex event is stored in the buffer of operator nodes after being composited, and mapped to the corresponding operator processors' circular processing queues whose parents' sequence number equals to the value in the operator nodes' 'parent' domain to realize the sharing of the same sub-events. The processing procedure for the queue front is as follows: First of all, whether the queue's front element's lchild_data and rchild _data domain is empty. If at least one of them is empty, then we need to wait. If both of them are not empty, then we can composite the operator's two children's values (If the operator is ';', we need to increase the judgment of the arrival time of left and right node). After composition, the composite results, the value of 'data domain' and 'Node' need to be sent to the general processor. Finally, the general processor, according to the value of 'data domain', matches from the corresponding operator queue whose operator is equal to the value of 'data domain' to match the value of 'lchild' or 'rchild' domain with the value of 'Node'. Then we can assign the composite results to the corresponding 'lchild_data' domain or 'rchild_data' domain.

The new results produced by the compound of the lower operator tree can be used as local results, which can prepare for triggering upper processor corresponding to the operator to execute related operations [5]. The global results are achieved bottom-up through compositing local results. When dealing with simplest sub-query in circular processing queue and the correlated condition is satisfied, the operator can perform the corresponding operation.

D. Distributed Complex Event Processing Algorithm Based on Query Plan

The construction of query plan during the distributed IOT complex event query processing has been studied in the above sections. In this section, we propose the distributed query plan processing model based on DAG. The model can convert query expressions automatically into a DAG structure, and process the query predicate, and construct the distributed query plan based on DAG, to realize automatic detection and processing of complex events. The process of constructing distributed IOT complex event processing query plan is as follows.

Step1 Constructing DAG hierarchical model. Analyzing the structure of query expressions from lexical and grammar, and converting it into a DAG structure, and representing it using orthogonal list.

Step2 Predicate parse. It aimed at two types of query predicate [4, 8]: Simple predicate and parameterized predicates. These two types of predicates are respectively arranged near the corresponding vertexes on DAG to filter sequence of events.

Step3 Time correlation constraints. In view of time constraints in queries, the time constraints are mapped to the end point [7] of DAG. Contrasting the time span between the tail events and the first event with the time window T, which can determine whether the event sequence should be output.

Step4 Query decomposition. The query expression is decomposed into the simplest sub-query, and the internal operator tree (the simplest sub-query) can be constructed, and the simplest sub-query is assigned to the corresponding operator processor.

Step5 Data reading. When an event of input event stream reaches the handler node, system will read the data.

Step6 Constructing a distributed query plan. Using the distributed complex event query plan proposed in this paper, we can process these events.

Step7 Sequence converting. Relating all the sequences' properties which satisfy all the constraint on the end point, and converting the event sequence into a complex event, and outputing them.

IV. SIMULATION EXPERIMENTS AND ANALYSIS

In this section, we exhibit a detailed performance analysis of the DQCEP algorithm and testify the effectiveness and timeliness of its query processing through the comparison between DQCEP and SASE.

A. Experiment Settings

We implemented the algorithm presented in the previous sections in a C-based prototype system. The handling performance of the algorithm was tested and the results were analyzed. All the experiments were performed on a workstation with an Intel Core 3.10 GHz processor and 2G memory running Visual C++ 6.0 on Windows 7.

For there is no universal data set associated with IOT at home and abroad and it is difficult to obtain massive application data of IOT, we implemented an event sequence generator that can be used in different application systems for simulation. In accordance with the description of the model described in the section III, we compared the DQCEP algorithm with SASE and some analysis is also made to test the processing efficiency of the DQCEP algorithm presented in this paper.

B. The Algorithm Memory Usage Comparison

In the experiment, we analyzed the memory usage of the two algorithms in the same data sets to study the processing performance of the proposed distributed complex event processing approach in this paper. We can see from Fig.3 that with the increasing of the amount of basic events, the memory usage of the two algorithms presents an increasing trend. And it is apparent that DQCEP is superior to SASE. This is because SASE using AIS to store intermediate results, which causes a large number of backtracking during detection. While DQCEP needs to use orthogonal list to store directed acyclic graph at the beginning of the process, and the memory usage in the subsequent processing will increase with the increasing of the amount of complex events, but the change will not be obvious. Therefore, with the increase of the amount of basic events, the memory usage of DQCEP algorithm is smaller than SASE. As a consequence, the advantages of the DQCEP algorithm become more obvious, when the data set increases.

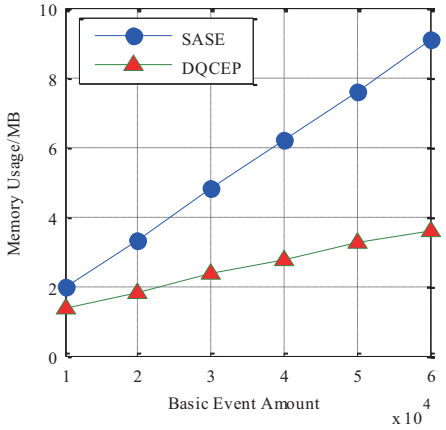

Fig.3. Influence of basic event amount to memory usage

C. Algorithm Processing Time Comparison

In the experiment, we analyzed the execution time of the two algorithms in the same data sets to study the processing performance of the proposed distributed complex event processing approach. We can see from Fig.4 that with the increasing of the amount of basic events, the execution time of the two algorithms presents an increasing trend. But when the

amount of basic events is the same, the execution time of the proposed DQCEP is significantly shorter than that of SASE. Because SASE algorithm which use large sliding windows for the constraint of data out of date needs to traverse in each AIS and the time cost is high. While the DQCEP algorithm has a fixed processing mode and the intermediate results can be mapped to the circular processing queue for sharing, so the execution time is proportional to the amount of basic events. In practical, the events that need to be addressed are magnanimous, so when the data set increases, the advantages of the DQCEP algorithm become more obvious.

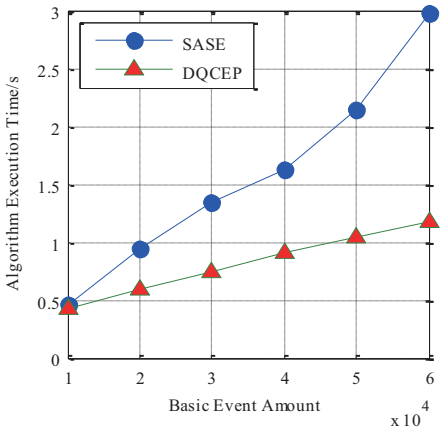

Fig.4. Influence of basic event amount to execution time

V. CONCLUSIONS

With the applying and development of IOT technologies, data stream process is becoming more and more important. Complex event processing is an efficient method in data stream processing of Internet of things. There are some researches about this, but more of these methods are referred to single complex event or a small quantity of events, multiple complex events can still not be processed well. In order to solve this problem, a distributed complex event processing architecture for Internet of things is presented in this paper, in which a distributed query plan of complex event process structure based on directed acyclic graph is given, moreover, a distributed query plan complex event processing algorithm based on directed acyclic graph is proposed. The complex tasks are divided into several single sub-tasks which are processed in parallel with the corresponding operator nodes. The simulation results indicate that our method is more efficient. In the future work, we will apply this method to some scenarios to validate the effectiveness, and continue our research on sharing the common sub-events in the process of queries on event streams and provide a corresponding cost evaluation models and optimization methods.

[1] L. Y. Yuan, X. C. Wang. “Study on IOT spatio-temporal data description model based on semantics,” Proceedings of the 2013 International Conference on Control Engineering and Communication Technology. China: IEEE Computer Society, pp. 759-764, August 2013.

[2] X. Y. Shi, D. H. Sun, X. X. Song. “A CEP-based RFID data processing model,” Techniques of Automation and Applications. China, vol. 27, no. 4, pp. 74-76, April 2008.

[3] E. Wu, Y. Diao, S. Rizyi. “High-performance complex event processing over streams,” Proceedings of the 2006 ACM SIGMOD international conference on management of data. Chicago: ACM, pp. 407-418, June 2006.

[4] Y. H. Wang, S. H. YANG. “High-Performance complex event processing for large-scale rfid applications,” Proceedings of 2010 the 2nd international conference on signal processing systems (ICSPS). China: IEEE, pp. 127-131, July 2010.

[5] F. S. Wang, S. R. Liu, P. Y. Liu. “Complex RFID event processing,” The Very Large Databases Journal, vol. 18, no. 4, pp. 913-931，August 2009.

[6] B. Y. SONG, H. Z. LOU, M. TANG, et al. “An extended sub-query sharing approach over RFID event streams,” Journal of Chinese Computer Systems. China, vol. 33, no. 9, pp. 1898-1902, September 2012.

[7] D. D. Xu, L. Y. Yuan. “Method of IOT complex event processing based on event sharing mechanism,” Journal of Computer Applications. China, vol. 35, no. 2, pp. 326-331, February 2015.

[8] L. J. Hu, Y. Luo, Y. Y. Wang. “Mapping transform of query plan model in grid database based on Petri net,” Journal of Computer Applications. China, Vol. 27, no. 6, pp. 1378-1381, June 2007.