Efficient Composite Event Detection Based on DHT Protocol

Amina Chaâbane*†, Salma Bradai*, Wassef Louati* and Mohamed Jmaiel‡
*University of Sfax, B.P. 1173, 3038 Sfax, Tunisia
†University of Kairouane, B.P. 471, Kasserine 1200, Tunisia
‡Research Center for Computer Science, Multimedia and Digital Data Processing of Sfax, B.P. 275, 3021 Sfax, Tunisia

Email: amina.chaabane, salma.bradai, wassef.louati, mohamed.jmaiel@redcad.org

***Abstract*—Current communication systems restrict user affordances in terms of expressiveness and interaction. Complex Event Processing (CEP) have gained much importance to overcome these shortcomings. It allows users to interact with powerful expressiveness when defining logical and temporal patterns of exchanged events. This expressiveness allows cleaning network traffic by eliminating the routing of useless events and consequently reducing the overall consumed energy. However, handling the expressiveness of desired events, filtering coming events still challenging features, especially with the growth of data size encapsulated as events in the network. In this work, we rely on Publish/Subscribe system based on Distributed Hash Tables (DHT), which offers intrinsically flexible event routing, scalability and load balancing in order to manage distributed composite events routing. For efficient event filtering, we propose a three dimensional indexing hash space named CECube as a smart data structure for rapid CEP over DHT. The CECube indexes firstly composite subscriptions, then basing on a simple binary search, it serves as publications filter and helps making the right decision for what events should be aggregated and forwarded to the adequate subscribers.**

The performance of our solution is evaluated using FreePastry simulator. The results demonstrate firstly that our approach is efficient in terms of filtering process and that the average number of routing nodes is decreased. Secondly, we prove the superiority of our approach as compared to another existing work.

***Index Terms*—Complex event processing, Composite event, Publish/Subscribe, logical and temporal pattern, P2P networks, DHT.**

I. Introduction

The rapid emergence of new technologies and systems, such as the Internet of Things (IoT) [1], offers various important services that become indispensable in people's daily life. The mobile crowd sensing in particular [2], [3] levrages people mobility and their smartphones' integrated sensors (GPS, gyroscope, air quality sensor., etc.) to produce sensitive data or events allowing the monitoring of common phenomena such as traffic condition, air quality, people density and more.

To ensure message and information dissemination between scattered users and devices, communication systems are using different protocols faced with a large scale users. However, they increase data traffic, affect network efficiency and bothers users when they receive useless data. For that, we thought of filtering exchanged data by integrating event-based systems [4], [5] that allow users to interact with others and to specify the kind of messages or notifications they want to receive. Arising as an important communication paradigm, event based systems could cover a great number of users with a great information amount. They ensure communication between clients (consumers or producers) via an event service notification. This event service is composed of brokers that are responsible for event filtering and routing. It ensures decoupling along time, space, and flow dimensions between clients which promotes scalability property. Its scalability depends also on the event service topology. It grows in importance from centralized to hierarchical and Peer-to-Peer (P2P) architectures. With P2P topology, event service topology can be structured or unstructured. Scalability is more sophisticated with structured topology based on Distributed Hash Tables such as Pastry and Chord.

The event is the main concept in these systems representing generally subscription (user's interest) or publication (produced event). Users have more and more various events for publication and attempt to search for multifarious events. With information diversity, expressiveness of these systems is a salient feature to evaluate system efficiency against user's requirements. Publish/Subscribe (Pub/Sub) system handles expressiveness mainly with topic-based and content-based systems. But, user can require or produce many events concurrently or sequentially with various contents and topics. User's interest becomes more exigent when he searches to reorganize favorite events reception or detect significant event or event resulting from occurrence of other events. Consequently, a lot of events could come with a high rate, but few of them could satisfy user's interests.

This issue can be handled through Complex Event Processing (CEP) approaches. CEP consists in collecting information produced by multiple, distributed sources, to process it in a timely way, in order to extract new knowledge or valuable event as soon as the relevant information is collected. It is based on event patterns which offer the possibilities to aggregate and correlate events together by a set of understood relationships in order to make them a source of great power, called Composite Event (CE). In fact, we can consider the following definition of event composition as proposed in [6]: "Composite subscriptions consist of atomic subscriptions linked by logical or temporal operators, and can be used to

978-1-5090-5756-6/17/$31.00 ©2017 IEEE
SERA 2017, June 7-9, 2017, London, UK

express interest in composite events. A composite subscription is matched only after all component atomic subscriptions are satisfied". The dependency between events could be represented by logical (And, Or, Xor, Not) and/ or temporal relationships (Before, After, Meets, Overlaps, Finishes, Includes, etc.). However, collecting events from scattered brokers and aggregating them to composite events according to different event patterns, brings new challenges in terms of their routing and management. Widely explored in the literature, this issue stills require improvement for distributed event service, in particular event service under P2P architecture for CEP.

After having studied existing approaches, this paper relies on Pub/Sub system based on Distributed Hash Tables (DHT) for event routing. DHT provides a lookup service similar to a hash table. The (Key, Value) pairs are stored in a DHT node (rendezvous node). Each DHT node has a unique identifier (nodeID) and stores some (key, value) pairs which have closest keys to its nodeID. Generally, nodes are organized virtually according to the order growing of nodes identifiers on a DHT ring (see Pastry [7], Chord [8], etc.).

We use topics to compute keys for routing atomic/composite events in a DHT. Our Pub/Sub communication layer will be responsible for disseminating composite topic into primitive and sub-composite topics that are mapped on nodes responsible on their hashed keys. Furthermore the paper concentrates on combining all logical and temporal relationships on a scalable P2P event-based system. We aim to provide a composite event management solution that deals with all composite event patterns and structured P2P networks. To this end, we propose a three dimensional indexing hash space named CECube for detection of composite events produced throughout a distributed network. While basing on binary search, it allows not only an efficient events matching and filtering process, but also it reduces useless transfer of atomic/primitive events throughout brokers' network. This is by checking temporal and logical constraints before their sent. Our approach provides CEP requirements of a performed P2P Pub/Sub middleware that can be integrated with IoT systems or social networks.

The remainder of this paper is organized as follows. We detail existing related works with advantages and shortcomings in section II. Then in section III, we introduce our proposed approach of composite event filtering based on DHT to overcome previous works limitations. Thereafter, we detail the indexing and routing process in subscription phase in section IV and the matching process in publication phase in section V. Results obtained from several experiments are provided in section VI. We finish this paper with a conclusion and future work in section VII.

II. Related work

In this section, we provide an overview of several research efforts in the literature focused on the CEP. Then we discuss Pub/Sub systems and CEP applied to social networks, IoT and crowdsensing systems.

A. CEP

The CEP appears as an important issue in the Pub/Sub systems due to user requirements value. In fact, taking account of composite user interest provides a flexible and energy-efficient manner and performs near real-time processing of Big Data streams.

P. R. Pietzuch et al. [9] propose a Composite Event Detectors (CED) based on a core CE language compiled on Finite State Automata (FSA). These CED are devoted to detect concurrent composite event patterns with specific parameterization and a rich time mode. However, they cannot be deployed on distributed network. While CED suffer especially from scalability issue with centralized automata detectors, authors propose to improve this solution by duplicating these detectors at favourable locations according to the network bandwidth and sources of composite events [9]. Nevertheless, this proposition remains suffer from scalability problem. In fact, when some composite events are required too much, nodes of CED would be certainly overcrowded. It suffers also from user-friendly definition of new composite event according to user requirements as he needs to define a new CED.

Others works appear to purge scalability shortcoming and enhance composite event relationships. In this context, Courtenage et al. [10] propose Composite Event Detectors and Atomic Event Detectors created on different nodes according to received subscriptions. The identity of an event detector broker is located in the network by hashing the event type and route it to a node having the closest successor identifier to the service identifier (AED/CED). Consequently, this solution lacks support for temporal relationships which are a cornerstone for CE expressiveness and usefulness.

Steven Lai et al. [11] handle composite event detection for sensor networks with temporal, logical and spatial event composition. Each sensor node is programmed to detect specific composite event according to program images conveying the composite event and the sensor nodes responsibilities. Thus, the definition of new composite event requires new program definition which is out of the simple user scope.

Last work is named JTangCSPS proposed by J. Qian et al. [12]. It is a composite and semantic Pub/Sub system over structured P2P networks. They use OWL language to describe semantic events. Since OWL does not allow defining logical and temporal operators, they use RDF graphs to describe the relations of composite event. The major shortcoming of this work is that all primitive events shall be reached and checked before the checking of the composite event relationships which is unnecessary in some cases as when using the OR and XOR operators.

There are few approaches that have succeeded to maintain CEs management, but some of them fail to comply with the event expressiveness such as the breach of some logical and temporal relationships, and others are built on unstructured or centralized event notification system. In this paper, we present a composite event pattern modeling and distributed composite event management to achieve large-scale and efficient routing

over P2P Pub/Sub system. In the next, we discuss CEP application to the IoT, Crowd sensing and social network.

B. CEP could be applied to IoT, Crowd Sensing systems and social networks

IoT and mobile Crowd Sensing are responsible to supervise and collect data over a large network of senors and mobile devices which results high traffic. Recently, few approaches propose to use CEP to detect valuable events for real time IoT application. Chen et al. propose a distributed CEP engine for IoT applications [13]. They use only logical operators to define complex event patterns.

The CenceMe application retrieves and publishes automatically sensing people's presence to social networks through mobile phones [14]. It generates a lot of traffic by sensing data, but without any filtering on generating data. It uses the phones and the backend servers to achieve scalable inference. In the same context, PEIR [15] is also a participatory sensing application based on GPS location data collection using mobile phone. It estimates personal exposure to pollution and environmental impact. It uses client-server architecture which affects scalability of the system. It is also not real time application and does not filter exchanged data.

Therefore, some IoT applications need real time processing for handling big data streams. For that, they use Pub/Sub system to collect interested data by simple event filtering. Pogo middleware [16], is an application used on mobile phones to facilitate the access to sensor data for the research community, uses Pub/Sub system with simple topic-based for filtering. It aims to achieve energy-saving on mobile devices by simple filtering of sensed data on mobile devices. Tong et al. [17] propose an ubiquitous Pub/Sub Platform for wireless sensor networks. They provide content-based Pub/Sub with high level of abstraction from the underlying sensors and network infrastructures. Users can subscribe for sensing data by simply specifying the target area, sensing types and data ranges of interest. CUPUS [18] for CloUd-based PUblish/Subscribe middleware, is a mobile crowd sensing system that reduces energy consumption significantly on mobile devices and sensors by suppressing the transmission of redundant and irrelevant data into the cloud. To summarize, there are few IoT applications using Pub/Sub system with simple filtering which reduce energy consumption and traffic. These objectives can be more satisfactory with CEP. Especially for the social networks that attract a majority of the Internet users which increase significantly User Generated Content [19]. Unfortunately, users are not always satisfied by received contents so the checking of composite user interests reduces the network traffic and improve the network efficiency.

In the next, we detail our CEP approach for event filtering over DHT.

III. The proposed approach of composite event filtering based on DHT

A. Complex event modeling

In our work, we define a complex event based on a composite event definition as an aggregation of primitive or sub-composite events with a set of logical operators and temporal constraints. Therefore, We formulate a Composite Event (CE) made up of the set of events E, where E can be a primitive or a sub-composite event with as follows:

$$CE = [op_i(E_{i_{/Ti}}, E_{j_{/Tj}})] \quad (1)$$

With:

- E_i / E_j : primitif or sub-composite event; so that a CE could be the aggregation of two sub-composite events, two primitive events or the aggregation of a primitive event and a sub-composite event;
- T_i / T_j : temporal constraint of E_i / E_j;
- op_i : logical/temporal (Log/Temp) relationship between E_i and E_j.

For example to compose three primitive events with logical operator "and", according to our formula, the generated CE will be as follows: CE=[and ($E_{3_{/T1}}$, [and ($E_{1_{/T1}}$, $E_{2_{/T2}}$)]]

B. Approach overview

In our approach, we extend structured P2P Pub/Sub system for topic-based event in order to support composite topic-based event while relying on DHT protocol. The specific contributions of our work revolve around three main pillars.

Firstly, brokers could act either as event sources and event consumers. They act also as rendez-vous/filter nodes between publishers and subscribers. In fact, matching publications into subscriptions needs that subscriptions and publications for particular events meet at a certain nodes in the system where they can be compared.

Secondly, In case of a composite subscription (CS) (subscription with a composite event), our Pub/Sub communication layer is responsible on its decomposing into primitive and sub-composite events called its members. Those members are mapped later to their root nodes responsible on their hashed identifiers. The decomposition and mapping process follows a tree structure shown by Fig. 1. The example in the figure shows that subscriber1 desires the reception of CE_1 which is is the aggregation by the logical operator "and" of two primitives events e_1: foot ball match and e_2: comments in english. According to this example, the tree structure is built in subscription phase from parent (N_1) to roots ($n_1 and\ n_2$). In publication phase, it is followed inversely so that the event composition will be checked gradually on rendez-vous nodes. Useless events that have not m et any primitive and/or composite subscription stop their dissemination over the tree, and hence reducing the network traffic.

Moreover, our proposed decomposition process can detect shared or repeated primitive or composite events. As shown the Fig. 2, the composite subscription Sub_1 is reused in the composite subscription sub_2. At this stage, the re-decomposition

of sub_1 and re-routing of its members is unnecessary as it was alerady performed when handling sub_2. Consequently, we avoid again unnecessary events transfer.

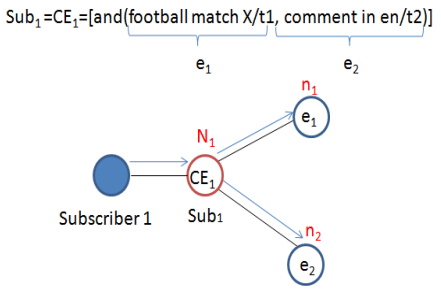

Fig. 1. Decomposition tree of CE_1

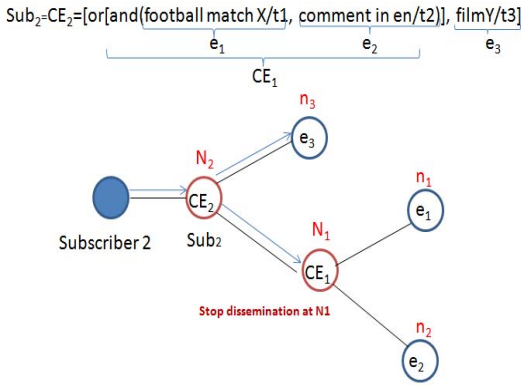

Fig. 2. Decomposition tree with CE_1 as a common event

Thirdly, we propose two smart structures, the "plane" and "CECube" for indexing the primitive and composite subscriptions respectively. For both of them, the indexing process is based on an EventFlag $\in \{0, 1\}$ to indicate the subscription existence in the current rooting node. Our objective behind using those structures is to make matching events easier and more efficient in the same time. Regarding the "Plane" structure, its main role is to check logical and temporal constraints of a primitive subscription.
Regarding the "CECube", it allows CS indexing according to its members. More details of the "plane" and "CECube" indexing and matching processes are given in section IV and V respectively.

IV. Primitive/Composite Event Indexing in Subscription Phase

A. Indexing primitive subscription through Plane structure

To store primitive events on a responsible broker, we define a Plane structure with two axes $I(e)$ and $I(T)$ as shown in Fig. 3, where $I(e)$ is the hashed value index of primitive event and $I(T)$ is the hashed value index of the event occurrence time. We use the uniform hashing function sha-1 that can accommodate all IDs without conflict. Each cell in this Plane contains a value denoted as PlaneValue. It consists of two pieces of information. The first is the EventFlag, which is a bit value that indicates whether a subscription contains a primitive event "e" at time T. Therefore, the EventFlag is set to "1" in order to indicate that there is a subscription for an event e_i at time T_i. The second is a Subscriber Identifier Vector (SIV) that stores identifiers of subscribers to the primitive event. Subscribers could be either nodes responsible on composite or simple subscriptions. In fact, primitive events can match primitive subscriptions or belong to one or more composite subscriptions, to be aggregated later (using the cube structure) with other members of the same CE.

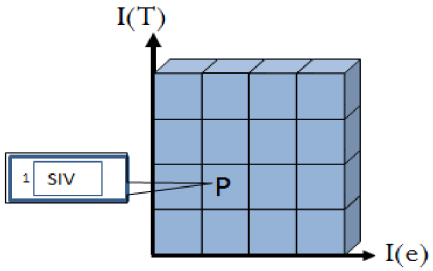

Fig. 3. Plan structure

B. Indexing Composite subscription through CECube and Composite Event Matching Vector (CEMV)

To provide efficient matching operations in a distributed event service, we propose a three dimensional indexing hash space named CECube. The CECube is maintained by each broker that is responsible for a composite subscription. It is used to store composite topic pattern, i.e. its aggregated members, and its logical and temporal relationships. The axes respectively represent the hashed value index for composite event CE, primitive or sub-CE belonging to CE denoted E and the occurrence time T. The axes indexing process is performed similarly to the Plane axes. We note that sub-CE or CE hashing is performed without event relationships and events time, which reduces the number of built cube. The occurence time of sub-CE determines the last occurence time of its events E.

When a broker receives a composite subscription, it begins by defining its members (primitive events and/or sub-CEs) in addition to their Log/Temp relationships. Then, it uses its cube structure in order to index and map all members into their corresponding cells.

As shown in Fig. 4, a Plane perpendicular to the I(CE) axis is identified as [I(CE),*,*] and denoted as "CellMatrix". It represents the composition of the corresponding CE at any time. Similarly, a line parallel to the I(E) axis forms the composition of CE at a specific time T. It represents the CellSequence and is identified as [I(CE),*,I(T)]. Each cell of the cube [I(CE),I(E),I(T)] is denoted as cubeValueSet and maintains a set of values as follows:

- The EventFlag $\in \{0, 1\}$: as a bit that indicates whether a composite subscription with (CE) contains an event E (primitive or composite) at time T.

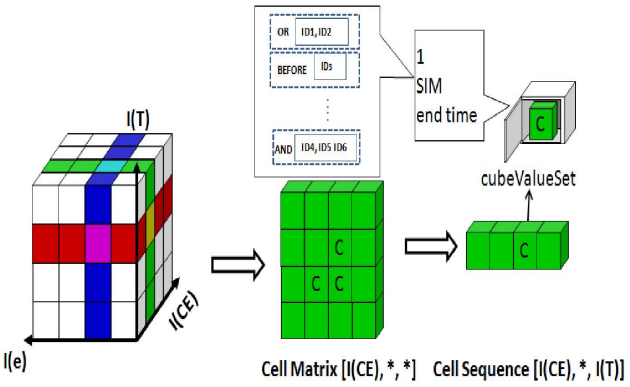

Fig. 4. CECube structure

- Subscriber Identifier Map denoted as SIM: It contains pairs of (key, value) as follows:
  - The key is the relationship;
  - the value is the the corresponding vector of subscribers' identifiers (SIV). In other words, they are subscribers to the concerned CE aggregated with the same relationship.
- The end time of the event/sub-CE: necessarily for temporal relationships checking.

To check event composition total matching efficiently, we define a *Composite Event Matching Vector* (CEMV) for each CE. Our aim is to check easily the total matching of a given CE with the coming of its members. As shown the Fig. 5,

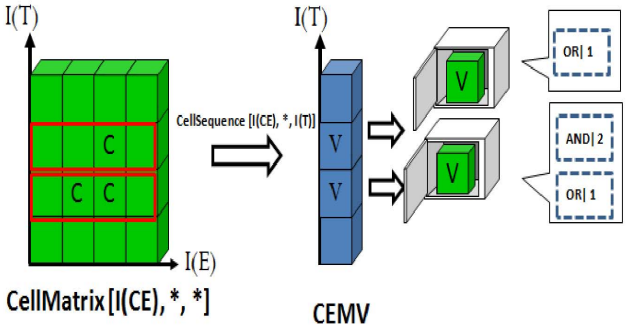

Fig. 5. Composite Event Matching Vector modeling

values of each element in the CEMV will contain pairs of (key, value) as follows:

- The key: is relationship
- The value: is the EventFlag summation in the CellSequence [I(CE),*,I(T)] of the concerned CE aggregated with the same relationship indicated in the key.

Those two information will make the matching easier in the publication phase as explained in section V.

C. Primitive/composite event handling in subscription phase

While basing on the Plane and CECube structures, the subscription processing of a composite and primitive subscriptions are shown in Algorithm 1.

On receiving a composite subscription according to the CE pattern defined by formula (1), we check if the same composite subscription (Sub) is already indexed before (line 3). In case that the concerned routing node has never received similar composition of Sub, it starts by decomposing the subscription to reveal its members (E). Then, According to the hash indexes of CE and E, their start time, it indexes those members in their corresponding cells in the cube. Therefore, each cell of [I(Sub),I(E),I(T)] will contain the EventFlag set to 1, the SIM containing pairs of [[relationship, Vector of subscriber's ID (SIV)] and the end time of the event (E). Then the concerned Composite Event Matching Vector is updated with adding the pair [relationship, summation of EventFlag]. Then, the events (E) will be routed to their rooting nodes, that will process the same in case that (E) is itself a composition (lines 4-8).

If Sub is already indexed before, the process of cube indexing will just check the relationship. If it is the same as the already indexed subscription, it will just add the subscribers' identifier to the adequate pair in the SIM. Otherwise, it adds an other pair [relationship, SIV] in the same SIM. Then, the decomposition is stopped as events subscription are already routed by the already indexed subscription (lines 10-11).

When a rendezvous node receives a primitive subscription (line 15), it updates its Plane structure by setting corresponding EventFlag to "1" and adding sender node to the SIV using $plane$ function (line 16).

Algorithm 1: Subscription processing

```
   Data: subscription Sub, sender S
   Result: E1, E2, op
1  switch type of Sub do
2      case Composite do
3          if Sub never mapped then
4              decompose(Sub, E1, E2, op);
5              cube(Sub, E1, E2, op, S);
6              updateCompositeEventMatchingVector
                (Sub, E1, E2);
7              send(E1);
8              send(E2);
9          end
10         else
11             updateSIM(S);
12         end
13         break;
14     end
15     case Primitive do
16         plane(Sub, S);
17         break;
18     end
19 end
```

Fig. 6 depicts the example of composite subscription shown previously by Fig. 2. For composite subscription sub_2, a cube structure is created in N_2 as the root node of the CE_2.

Then, members of CE_2 are mapped in the cube according to their hash index of events and occurence time. The occurence time of the sub-CE (CE_1) is the last occurence time of its members e_1 and e_2 (they have the same occurence time in the example as the match should coincide with comments). The Composite Event Matching Vector responsible on the Plane [I(CE_2),*,*] is updated by adding the pair [logical relationship ("OR" in the example), summation of the EventFlag=1] for the CellSequence [I(CE_2),*,I(t)]. Then event members are routed to their root nodes (N_1 and n_3 respectively). The same scenario is performed in N_1 responsible on CE_1.

When primitive events are routed to their root nodes, a Plane structure is created to index those events according to their ID and time occurence. For example, for e_1 the concerned PlaneValue will contain the EventFlag set to "1" and the SIV containing the ID of its subscriber which is N_1 in the example.

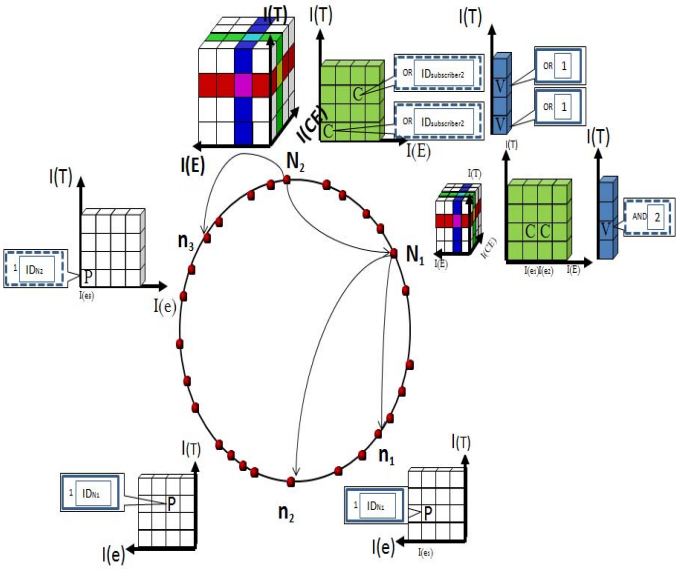

Fig. 6. Composite subscription processing

V. Matching of primitive/Composite event in publication phase

The publication phase consists in collecting and aggregating all primitive and/or composite topics from different brokers to satisfy composite subscriptions. The aggregation process will respect tree decomposition from leafs to the root in order to avoid unnecessary event notification when primitive event does not satisfy logical and temporal constraints.

The Algorithm 2 details steps of the publication phase. We distinguish primitive event from composite event handling. In the first case, a primitive publication is received on the root node of primitive event (line 1). We check through the Plane created in the subscription phase if the corresponding EventFlag is set to "1" (line 2). Then, we use SIV of the PlaneValue for sending primitive events to their subscribers (lines 3-5).

In the second case, an event E (primitive or sub-CE) is received by a node that maintains a cube structure (line 7). We look at the Plane [*, I(E), *] perpendicular to the I(E) axis to check if it belongs to any composite subscription (line 8). When EventFlag is set to "1", we look up the corresponding [I(CE),*,*] CellMatrixs and scan all cubeValueSets for event matching and relationships checking and to found following destinations (SIV) (lines 9, 10). As the event E is reached, the EventFlag in the concerned cell which was alerady set to 1 in the indexing process will be set to 0 in this matching process (line 11). Note that we take into account the relationship when updating the EventFlag. For example for the logical relationship "OR", once an event is reached, we update also the EventFlag of the other member and set it to 0. After verifying Log/Tem relationship with other events (line 12), we update the CEMV to check the total matching of the concerned CE (line 13). As the CEMV cells contain pairs of [Log/Temp relationship, summation of the EventFlag] and EventFlag of reached events are set to 0, when all summations of the EventFlag for the same relationship are set to 0, we reveal that the concerned CE is matched. In this stage, a notification is sent to the subscriber using the adequate SIV in the adequate SIM stored in the cubeValueSet. Finally, we send composite event to those subscribers (lines 14-16-17).

Algorithm 2: Publication processing

```
   Data: publication e, receiver R
   Result: E1, E2, op
1  if R_has_a_plane_structure and e_is_primitive then
2      SIV ⟵ scanPlan(e);
3      for dest ∈ SIV do
4          send(e, dest);
5      end
6  end
7  if R_has_Cube_structure then
8      CE ⟵ scanPlan([∗, I(E), ∗]);
9      for ce ∈ CE do
10         if scanCubeValue([I(CE), *, *]) then
11             updateEventFlag;
12             checkLog/TemRelationship;
13             updateCompositeEventMatchingVector(ce);
14             if (CompositeEventMatchingVector(ce)==0)
                then
15                 SIV ⟵ scanPlan(ce);
16                 for dest ∈ SIV do
17                     send(ce, dest);
18                 end
19             end
20         end
21     end
22 end
```

VI. Experimental results

Experimental results consist in two parts. The first part provides an experimental evaluation and explanation of the

benefits of our solution. In the second part, we compare it to JTangCSPS system to check its performance in terms of routing delay.

We have implemented our system over Scribe Pub/Sub system based on Pastry DHT. Our evaluation uses FreePastry simulator with almost the same conditions of JTangCSPS. Our measurements take place also on a standard PC installation with Linux libraries and a hardware configuration comprising Intel core i5 CPU 2.53 GHz, 4 GB RAM. The experimentations are made with a static DHT network that does not suffer from node failure.

A. Network Traffic reduction

To achieve our goal, we propose a CECube structure for event composition through P2P Pub/Sub system. Our approach allows to meet the needs of users and consequently reduce the network traffic by avoiding unnecessary transfers between users.

We propose two scenarios to evaluate our approach as far as traffic reduction is concerned. In the first scenario, we inject 200 composite subscriptions, each one is composed of five primitives subscriptions. This scenario allows to compare a Pub/Sub system with event composition over existing communication networks. In the second scenario, we inject the 1000 atomic subscriptions that make up the 200 composite subscriptions of the first scenario. This scenario helps to show the contribution of event composition with distributed Pub/Sub system. We measure the number of notifications received by customers in the different scenarios after 4000 publications sent over the network.

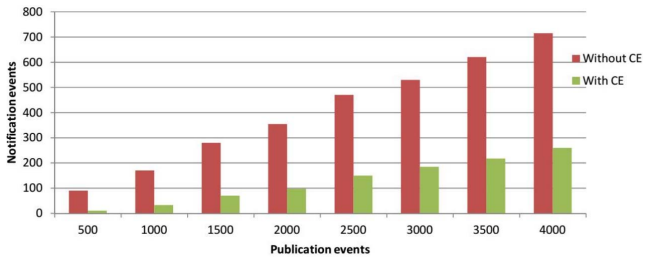

Fig. 7. Traffic reduction by Pub/Sub system with event composition

The result shows that the number of notifications sent to the customer is greatly reduced by using an event-based system. Indeed, the flow of publications can reach 100 % if we consider the case of Facebook where sharing is done between a group of friends. This type of sharing does not consider the interests of customers and as result we observe user frustration. However, considering the interests of customers, the reduction can reach 82 %, as shown by the curve. Moreover, the injection of composite subscriptions reduced more the overall traffic. The reduction reaches 65 % compared to a system using Pub/Sub without composition (first scenario). These results are shown by curves of Fig. 7.

B. CECube vs JTangCSPS

Finally, we compare our approach against JTangCSPS, explained in section II in terms of routing delay in order to evaluate our system performance. The routing delay of a composite subscription includes the network delay and the time to build the composite subscription tree and cube in JtangCSPS and CECube respectively and to split the composite subscription at each node. As JtangCSPS evaluation, we have tested with different scenarios when the number of primitive subscriptions of the composite subscription varies from 2 to 20. Each broker randomly makes 20/i composite subscriptions, each consisting of i primitive subscriptions where i = 2, 5, 10 and 20. To simplify the comparison, only operator conjunction "AND" is used in composite subscription, and each primitive subscription only contains one type constraint.

Fig. 8 shows that CECube has a lower routing delay than JTangCSPS. This is because, in CECube, index construction in the cube architecture is less expensive than the tree construction with JTangCSPS. With JTangCSPS, the routing delay increases fast before NoPS reaches 5 and then it increases slowly. It is the same aspect with our CECube. Scalability of our system is explained by the stability of routing delay which is almost the same with CECube regardless of the number of nodes. However, it increases by 800 ms between 500 and 1000 nodes with JTangCSPS.

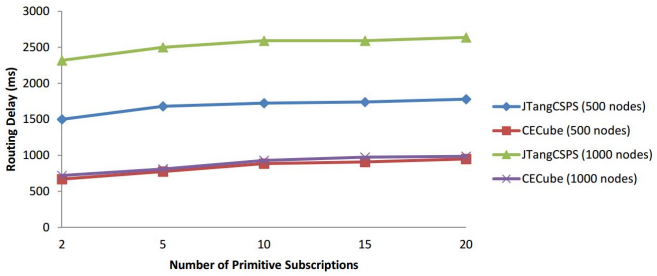

Fig. 8. Routing delay vs. number of primitive subscriptions with JtangCSPS and CECube

VII. Conclusion and Future Work

In this paper, we present a new three dimensional hash space, the CECube to perform CEP distributed over structured P2P network. Our approach proposes a plane and cube structures. The plane structure is used to index primitive event according to time occurrence and event ID hashing with bit indication. It simplifies event occurrence checking. The CECube structure indexes composite events based on a binary information in the subscription phase. In publication phase, matching process is performed while basing on this simple binary information. This approach reduces the required amount of events that has to be transferred on the network. With experimental results, we demonstrate that our approach reduces significantly the routing delay comparing to JtangCSPS system and it stills almost the same with increased network size.

An interesting direction of future research therefore would be to enhance semantic aspect of our approach in order

to fulfill user interests. It is also interesting to exploit our approach in various application domains such as the social network and IoT.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] M. Marjanovic, L. Skorin-Kapov, K. Pripukzic, A. Antonic, and I. P. Zarko, "Energy-aware and quality-driven sensor management for green mobile crowd sensing," *Journal of Network and Computer Applications*, vol. 59, pp. 95–108, 2016.

[3] J. An, X. Gui, J. Yang, Y. Sun, and X. He, "Mobile crowd sensing for internet of things: A credible crowdsourcing model in mobile-sense service," in *2015 IEEE International Conference on Multimedia Big Data, BigMM 2015, Beijing, China*, 2015, pp. 92–99.

[4] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf, "Design and evaluation of a wide-area event notification service," *ACM Transactions on Information and System Security (TISSEC)*, vol. 19, no. 3, pp. 332–383, 2001.

[5] Y. Liu and B. Plale, "Survey of publish subscribe event systems," 2003.

[6] A. Hinze and A. Buchmann, Eds., *Principles and Applications of Distributed Event-Based Systems*. IGI Global, 2010.

[7] A. I. T. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*. Springer-Verlag, 2001, pp. 329–350.

[8] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. ACM, 2001, pp. 149–160.

[9] P. R. Pietzuch, B. Shand, and J. Bacon, "A framework for event composition in distributed systems," in *Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware*, ser. Middleware '03. Springer-Verlag New York, Inc., 2003, pp. 62–82.

[10] S. Courtenage and S. Williams, "The design and implementation of a p2p-based composite event notification system," in *Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 01*, ser. AINA '06. IEEE Computer Society, 2006, pp. 701–706.

[11] S. Lai, J. Cao, and Y. Zheng, "Psware: a publish / subscribe middleware supporting composite event in wireless sensor network," in *Seventh Annual IEEE International Conference on Pervasive Computing and Communications - Workshops (PerCom Workshops), Galveston, TX, USA*, 2009, pp. 1–6.

[12] J. Qian, J. Yin, J. Dong, and D. Shi, "Jtangcsps: A composite and semantic publish/subscribe system over structured p2p networks," *Eng. Appl. Artif. Intell.*, vol. 24, no. 8, pp. 1487–1498, 2011.

[13] C. Chen, J. H. Fu, T. Sung, P. Wang, E. Jou, and M. Feng, "Complex event processing for the internet of things and its applications," pp. 1144–1149, 2014.

[14] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell, "Sensing meets mobile social networks: The design, implementation and evaluation of the cenceme application," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '08. ACM, 2008, pp. 337–350.

[15] M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. Hansen, E. Howard, R. West, and P. Boda, "Peir, the personal environmental impact report, as a platform for participatory sensing systems research," in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '09. ACM, 2009, pp. 55–68.

[16] N. Brouwers and K. Langendoen, "Pogo, a middleware for mobile phone sensing," in *Proceedings of the 13th International Middleware Conference*, ser. Middleware '12. Springer-Verlag New York, Inc., 2012, pp. 21–40.

[17] X. Tong and E. C. H. Ngai, "A ubiquitous publish/subscribe platform for wireless sensor networks with mobile mules," in *IEEE 8th International Conference on Distributed Computing in Sensor Systems, DCOSS 2012, Hangzhou, China*, 2012, pp. 99–108.

[18] A. Antonic, M. Marjanovic, K. Pripuzic, and I. Podnar Zarko, "A mobile crowd sensing ecosystem enabled by cupus," *Future Gener. Comput. Syst.*, vol. 56, no. C, pp. 607–622, 2016.

[19] C. W. Paper, "Cisco visual networking index: Global mobile data traffic forecast update, 2015-2020 white paper," Cisco, Tech. Rep., 2015.