A Novel Model for Contextual Transaction Trust Computation with Fixed Storage Space in E-commerce and E-service Environments

Haibin Zhang  
Department of Computing  
Macquarie University  
NSW 2109, Australia  
haibin.zhang@mq.edu.au

Yan Wang  
Department of Computing  
Macquarie University  
NSW 2109, Australia  
yan.wang@mq.edu.au

Abstract—In e-commerce and e-service environments, transaction context is important when evaluating the trust level of a seller or a service provider in a forthcoming transaction. However, most existing trust evaluation models compute a single value to reflect the general trust level of a seller without taking any transaction context into account. In the literature, a trust vector approach has been proposed to resolve the above problem. In particular, the trust vector contains different sets of trust values (termed as CTT values) so as to outline a seller’s reputation profile. As a result, buyers can identify the potential risk existing in a forthcoming transaction (e.g., value imbalance, i.e. a malicious seller may build up a high level of trust by selling cheap products and then deceive buyers by inducing them to purchase more expensive products) and thus avoid monetary losses.

In computing CTT values, some approaches are proposed that store the precomputed aggregation results over large-scale ratings and transaction data of a seller, so as to deliver prompt responses to a buyer’s query. Though these approaches allocate relatively small space to each seller for storing the aggregation results, if applied in a system with millions of sellers, space consumption will be intolerable. In this paper, we propose a novel model for CTT computation with fixed storage space, which provides a trade-off between aggregation detail and storage space. It is particular suitable for CTT computation where a request is regarding a seller’s trust in recent time period, e.g., the latest six months, rather than six months plus one day. We have conducted experiments on both an eBay dataset and a synthetic dataset to illustrate its good efficiency in responding to buyers’ CTT queries.

Keywords—E-Commerce; Contextual Transaction Trust; Storage Space; Data Warehouse

I. INTRODUCTION

The trust of sellers or service providers has always been a crucial issue to potential buyers in making decisions. At some e-commerce websites (such as eBay\(^1\)), a centralized trust management system has been developed to evaluate the reputation of sellers. However, as pointed out in [4], [7], the simple trust management system is fragile to malicious sellers. For example, in frauds with the notorious value im- balance problem [4], [1], a seller can gain a good reputation by honestly selling good and low value (price) products. Once having accumulated a good reputation, they may deceive buyers by inducing them to buy expensive products, but either not deliver the ordered product or deliver a fake. Such frauds occurring on eBay have been reported [7]. For instance, an Australian deceiver at eBay tricked people for more than AU$10K in total. A Californian deceiver deceived victims in transactions for exceeding US$300K in total. Note that e-commerce is also one type of e-service applications. As the value imbalance exists in both e-commerce and e-service environments, in the context of this paper, we do not explicitly differentiate them but take e-commerce as an example in analysis.

In our previous work [22], [24], we have analyzed and summarized the following reasons in most existing trust evaluation models why they cannot identify value imbalance in transactions.

- The lack of context consideration in transaction trust evaluation: In e-commerce environments, different transactions generally have different natures and contexts; even the same seller needs to be considered differently with regard to trustworthiness in different forthcoming transactions [13], [17].
- The static trust evaluation result: Most models compute a single trust value based on past transactions [5], [8], [14], [15], [16]. However, such a single value basically only reflects a seller’s general trust status, and is static with regard to any forthcoming transaction [12]. As a result, it can hardly predict the likelihood of a successful forthcoming transaction.

In view of the afore-mentioned problems, we have developed a trust vector approach to Contextual Transaction Trust (CTT) evaluation [23]. The computed trust vector of a seller considers some important transaction context dimensions, such as product category, time and price. Specifically, the trust vector includes: (a) the trust level of the seller in selling a specific product which is to be traded in a forthcoming transaction; (b) the trust level of the seller at a layer higher than the specific product in the product category hierarchy, within a price range and a time period; and (c) the trust level of the seller in a valid price range and a time period. Here, the trust values in the trust vector may change dynamically depending on the parameters (e.g., product category, price range and time period) specified in the buyer’s queries.

\(^1\)http://www.ebay.com/
Suppose a buyer plans to buy a ‘Canon EOS T3i SLR Digital Camera’ at a price of around $700’ from a seller. When computing the elements (b) and (c), a buyer can specify and adjust the ‘product category’ within the category hierarchy, such as ‘Canon DSLR camera’ and ‘DSLR camera’ in order. Meanwhile, the buyer may also specify and adjust the price range (e.g., ‘$600-$800’) and the time range (e.g., the latest six months). We use granularity to represent the above differences in transaction contexts determined by a layer in the product category hierarchy, a price range and/or a time period. In addition, such a query on different CTT values is referred to as a CTT query. The computation of the elements in the trust vector is referred to as the CTT computation. All the computed results in response to a buyer’s CTT queries can outline a seller’s reputation profile, which can greatly help identify the context imbalance problem2 potentially existing in a forthcoming transaction and assist buyers in avoiding potential monetary losses.

Moreover, the comprehensive solutions for CTT computation have been proposed in [25]. In particular, CTT computation is first modeled as an extended RA (Range Aggregate) problem in data warehousing. Then, three approaches, namely, eaR-tree, eaP-tree and eH-tree are proposed for efficient CTT computation. A brief review on these approaches is given in Section IV. Generally speaking, they all dynamically precompute aggregates over large-scale transaction data and ratings of a seller, with all necessary combinations of three hierarchical or dynamic dimensions of product category, transaction price and transaction time. Then, the aggregation results are stored appropriately in the specialized index forming a tree structure. Here the index containing some aggregates is also referred to as the aggregation index. Based on the above tree structure, the prompt responses can be achieved to a buyer’s requests on a seller’s different CTT values.

However, the previous approaches set the aggregation in the transaction time dimension at a single fine time granularity, namely, the x-axis represents the transaction time dimension in days (see Section IV-C), which leads to the problem of space consumption when storing aggregates. Clearly, with continuous growth in transaction time (e.g., one year) and significant increase of historical transaction data and ratings, the aggregation index under a single fine time granularity (e.g., by days) can easily become too large in size of consumed space. If these approaches are applied in the systems with millions sellers, the space consumption will become intolerable. In real applications, where each seller has large-scale transaction data and ratings over a long time period (e.g., one year), it is reasonable to allocate fixed or limited storage space to each seller. To some extent, the deletion operations can be used to save the storage space of aggregation index (i.e., transaction data and ratings older than one year ago can be eliminated). However, our work focuses on proposing a new structure for CTT computation with limited storage space, and the deletion operations can still be used as an ordinary means to further reduce space consumption.

In e-commerce environments, the recent performance of a seller is crucial when evaluating his/her trust level. Therefore, most trust evaluation models assign higher weights to the ratings of recent transactions [5], [8], [15], [11]. This characteristic will be utilized in our proposed model. In particular, we aggregate recent ratings at a finer time granularity (e.g., on a basis of per day), and aggregate earlier ratings that are less important at a coarser time granularity (e.g., on a basis of per month).

In the literature, Zhang et al. have proposed a Hierarchical Temporal Aggregation model with fixed storage space (HTA$^{FS}$) to deal with temporal aggregation over data streams [18]. According to our analysis, we firstly point out the limitations of HTA$^{FS}$ model in solving our targeted problem. Then, we present a new solution CTT$^{FS}$ model for CTT computation. The CTT$^{FS}$ model guarantees the fixed storage space for aggregation index as well as good performance in responding to CTT queries (see Section VII). Finally, we present the results of experiments conducted on both an eBay dataset and a large-scale synthetic dataset to validate our proposed structure and approach.

The paper is organized as follows. Section II reviews related work. Section III gives a detailed explanation of the transaction context. Section IV reviews our proposed approaches for CTT computation. A novel model using fixed storage space to store aggregation index is given in Section V. Section VI evaluates our approach experimentally, and Section VII concludes our paper.

II. RELATED WORK

A. Contextual Transaction Trust (CTT) Computation

In Section I, we have pointed out the lack of context consideration as the drawback in most existing trust evaluation models. In recent years, more studies start to focus on introducing transaction context in trust evaluation in e-commerce environments [12], [13], [17], [22].

In [24], we propose a preliminary trust vector taking transaction context factors into account. Meanwhile, we study its efficiency compared with some existing trust evaluation models. In [23], we simplify the above trust vector, and then propose first technical solutions to compute CTT values in vector. In particular, the CTT computation is further modeled as an extended RA (Range Aggregate) problem. In addition, in [25], we improve the work in [23]. Specifically, a new product category hierarchy with seven layers is introduced (see Section III) to support finer-grained analysis of contextual transaction trust, and then eaR-tree, eaP-tree and eH-tree are proposed for efficient CTT computation. Here we need to emphasize that, as pointed out in Section I, all the above approaches adopt a single fine time granularity, namely, the x-axis represents the transaction time dimension.

---

2The value imbalance problem is only a type of the context imbalance problem [24] in transactions, where imbalance can also exist in product categories.
in days. Therefore, they have the problem of large space consumption.

B. RA Query

In the literature, the RA (Range Aggregate) problem is relatively close to CTT computation. The traditional RA query [6] in a two-dimensional spatial data warehouse is to compute the total number of points that fall into a query region q surrounded by $[x_1, x_2]$ and $[y_1, y_2]$ (see Figure 1).

Some well-known approaches, such as the aR-tree [2], [6], the aP-tree [10] and the MVSB-tree [20], [19] are used for solving RA queries. However, the MVSB-tree focuses on aggregating the same data that overlooks the inserted points themselves, thus cannot be easily applied to our targeted problem. As a result, the work proposed in [23], [25] mainly extend the aR-tree and the aP-tree. A more detailed review on these approaches will be given in Section IV.

![Figure 1. An example of an RA query](http://www.eclass.de/)

C. Hierarchical Temporal Aggregation Model with Fixed Storage Space ($HTA^{FS}$)

More general problems in range aggregate (RA) are the range-temporal (two dimensions with one as the time dimension) and spatio-temporal (three dimensions with one as the time dimension) aggregations. In the literature, some approaches have been proposed to solve the above aggregation problems [9], [19], [20], [21]. However, all these approaches use a single time granularity without space limitation.

In contrast to most existing approaches using a single time granularity, Zhang et al. propose a Hierarchical Temporal Aggregation model with fixed storage space (denote as $HTA^{FS}$) [18]. Figure 2 depicts the general idea of the $HTA^{FS}$ model to deal with points aggregation in a one-dimensional space. A $k$-level time hierarchy, where $gran_1$ is at the coarsest time granularity and $gran_k$ is at the finest granularity. Suppose that the $HTA^{FS}$ model divides the time space $[begin, now]$ into $k$ segments. Each segment $seg_i$ $(i = 1, 2, ..., k)$ maintains the corresponding aggregations with the time granularity $gran_i$ (see Figure 2(a)). The term $begin$ denotes the start time and $now$ denotes the increasing current time. The new objects are inserted with the point ‘now’ moving to the right. The constraint for the $HTA^{FS}$ model is that the available storage space is limited. When the total storage of aggregation index becomes more than a threshold $S$, older information is aggregated at a coarser granularity.

For example, if the time space is divided into two segments $seg_1$ and $seg_2$ (see Figure 2(b)), according to the

![Figure 2. A general idea of $HTA^{FS}$ model](http://www.eclass.de/)

$HTA^{FS}$ model, the points in $seg_1$ are aggregated by month while the points in $seg_2$ are aggregated by day. Also, the $seg_1$ takes up $S_1$ disk pages and $seg_2$ takes up $S_2$ disk pages, where $S_1 + S_2 = S$. The corresponding dividing time is denoted as $t_{div}$. The $agg_1$ denotes the total value of all points inserted before $t_{div}$. Assume that the size of aggregation index for $seg_2$ becomes too large. In such a case, we need to choose a new dividing time $t_{newdiv}$ (see Figure 2(c)) and move the information from $seg_2$ to $seg_1$, namely, the points in span $t_{div}$ to $t_{newdiv}$ are aggregated at coarser time granularity instead of a finer granularity. In [18], Zhang et al. also propose two algorithms for moving the information from $seg_2$ to $seg_1$ as well as updating $seg_1$.

In Section III, we will discuss that three context dimensions product category, transaction amount and transaction time should be taken into account in CTT computation. The general idea of the $HTA^{FS}$ model will be used to deal with transaction time dimension in our proposed model. In addition, we also analyze the specific requirements in CTT computation that cannot be fully satisfied by $HTA^{FS}$ model, based on which we then propose our new solution in Section V.

III. TRANSACTION CONTEXT

In our previous work [23], [24], we have identified three context dimensions with influence on the trustworthiness of a forthcoming transaction. They are product category, transaction amount and transaction time. The context of a transaction can be represented at different hierarchical layers and ranges in these context dimensions.

- **Product Category (a static but hierarchical dimension):** The category of transaction items has a hierarchical structure. In our proposed trust model, we extend the existing four-layer hierarchical structure of eCl@ss³ standards and use a product category hierarchy with seven layers [24] to support finer-grained analysis on contextual transaction trust with ‘drill down’ and ‘roll up’ operations in the hierarchy. Figure 3 presents a small part of our extended product category hierarchy. Under each brand, there are corresponding products that belong to this brand.

- **Transaction amount and Transaction time (two dynamic dimensions):** Transaction amount and transaction time are two dynamic dimensions. The transaction amount

³http://www.eclass.de/
parameters of a CTT query, a set of a transaction price range and a specific time period. With a pair of product category hierarchy with a specific transaction amount of a product that a buyer wants to buy from a seller is price may also vary. For instance, the transaction amount varies from product to product, and a buyer's CTT query on "Information, communication and media" technology forms an N-ary tree structure in Figure 3. In general, the product category hierarchy (see Figure 3). When computing PCT, the price range and a time range can be specified as the parameters.

(c) Similar Transaction Amount based Trust (STAT): STAT is the trust value of a seller in a specific price range and the time range. STAT is important for analyzing the trust level of a seller in different price ranges.

In [25], we introduced another element Transaction Proportion based Trust (TPT) in the trust vector. TPT is specifically designed to indicate the risk of context imbalance in a forthcoming transaction. Its intent is to compute the trust level of the seller with regard to the proportion of the past transactions that are similar to the new transaction.

In summary, all the values in the above-introduced trust vector are associated with a forthcoming transaction. They may dynamically change because a buyer can specify and adjust the layer in the product category hierarchy, the price range and the time period for particularly computing PCT and STAT. With the combinations of all three arguments, different sets of trust values can be computed, all of which can outline the reputation profile of the seller with regard to various types of transactions.

C. Our Approaches to CTT Computation

As discussed in Section III, transaction context includes a static dimension, i.e., product category hierarchy, and two dynamic dimensions, i.e., transaction amount and transaction time. Obviously, a CTT query in the transaction amount dimension and transaction time dimension can be converted to an RA problem, namely, the x-axis represents the transaction time dimension in days while the y-axis represents the transaction amount dimension. Each point in such a two-dimensional space represents a transaction as shown in Figure 1. Consequently, a CTT query on a seller in a time range $[t_1, t_2]$ and a transaction amount range $[a_1, a_2]$ can be converted by computing the number of the ratings $\text{count}_{ra}$ and the sum of the ratings $\text{sum}_{ra}$ of the transactions that fall into the query range. In addition, by taking into account the product category hierarchy as a third dimension, we extend the traditional RA problem to compute the CTT values in proposed trust vector as shown in Figure 3.

Next, we give a general description on the completed tree structure in Figure 3. In general, the product category hierarchy forms an N-ary tree. In this N-ary tree, each node contains N entries (the number N depending on the node

![Figure 3. Part of the extended product category hierarchy of the segment "Information, communication and media".](image-url)

IV. The Approaches to CTT Computation

In this section, we briefly review some approaches proposed in [25] for CTT computation. In particular, we focus on explaining the reason why CTT computation can be regarded as extended RA problem.

A. CTT metrics

In the literature, existing studies average the rating values for calculating the trust value [3], [17], [15], [11], [8], [5]. Following this idea, we calculate each CTT value as the average of all the ratings in the past transactions for selling the products at a specific layer (e.g., 'DSLR cameras') in the product category hierarchy (see Figure 3). When computing PCT, the price range and a time range can be specified as the parameters.

\[
\text{PCT} = \frac{\text{count}_{ra}}{\text{sum}_{ra}}
\]

Essentially, the approaches in [23], [25] aim to compute the values in our proposed trust vector effectively. The trust vector contains:

(a) Transaction Item Specific Trust (TIST): TIST is the average of all the ratings in the past transactions for trading the same transaction item as in a forthcoming transaction.

(b) Product Category based Trust (PCT): PCT is the average of all the ratings of the past transactions for selling the products at a specific layer (e.g., 'DSLR cameras') in the product category hierarchy (see Figure 3). When computing PCT, the price range and a time range can be specified as the parameters.

Essentially, the approaches in [23], [25] aim to compute the values in our proposed trust vector effectively. The trust vector contains:

(a) Transaction Item Specific Trust (TIST): TIST is the average of all the ratings in the past transactions for trading the same transaction item as in a forthcoming transaction.

(b) Product Category based Trust (PCT): PCT is the average of all the ratings of the past transactions for selling the products at a specific layer (e.g., 'DSLR cameras') in the product category hierarchy (see Figure 3). When computing PCT, the price range and a time range can be specified as the parameters.

(c) Similar Transaction Amount based Trust (STAT): STAT is the trust value of a seller in a specific price range and the time range. STAT is important for analyzing the trust level of a seller in different price ranges.

In [25], we introduced another element Transaction Proportion based Trust (TPT) in the trust vector. TPT is specifically designed to indicate the risk of context imbalance in a forthcoming transaction. Its intent is to compute the trust level of the seller with regard to the proportion of the past transactions that are similar to the new transaction.

In summary, all the values in the above-introduced trust vector are associated with a forthcoming transaction. They may dynamically change because a buyer can specify and adjust the layer in the product category hierarchy, the price range and the time period for particularly computing PCT and STAT. With the combinations of all three arguments, different sets of trust values can be computed, all of which can outline the reputation profile of the seller with regard to various types of transactions.
capacity, and each entry represents a product category. Thus, multiple product categories in the same layer of the product category hierarchy form a node. In addition, each entry also maintains the aggregates count_r and sum_r that are obtained from the past transactions selling products in this product category. Meanwhile, each entry at a brand-based product category layer (e.g., aR-tree and aP-tree) is also a root of a subtree that is external to the product category hierarchy (see Figure 3). This subtree is similar to the original tree structures (e.g., aR-tree and aP-tree) used for solving the RA problem, which records the aggregates count_r and sum_r in the corresponding transaction amount and transaction time dimensions. The subtree can be of multiple layers, depending on the number of transactions and the distributions in transaction amount and transaction time in the corresponding brand-based product category.

Following the above ideas, we have proposed eaR-tree, eaP-tree and eH-tree for CTT computation. Due to space constraint, we will not include detailed descriptions on these approaches but refer readers to [23], [25]. As illustrated before, all these approaches adopt a single fine time granularity and aggregate the number of ratings (count_r) and the sum of the ratings (sum_r) in days. Therefore, as pointed out in Section I, they lead to the problem of large space consumption when dealing with large volume of sellers. To solve this problem, in the next section, we will present our proposed new model CTTFS for CTT computation.

V. A NOVEL MODEL FOR CTT COMPUTATION WITH FIXED STORAGE SPACE

In this section, we first analyze the specific requirements in CTT computation. Then, we propose the CTTFS model with fixed storage space to store the aggregation index containing count_r and sum_r. Finally, we take the eaR-tree [25] as an example to explain how to apply the CTTFS model in CTT computation.

A. Specific Requirements in CTT Computation

In [18], Zhang et al. proposed a reduction technique that transforms a RA query to several dominance sum4 queries. By this way, the HTA FS model can be applied to deal with three or higher dimensional aggregation problem. However, there are some specific requirements in CTT computation that have not been addressed in the HTA FS model.

- First, in the traditional multiple dimensional aggregation problem, each dimension is linear. By contrast, the dimension of product category in CTT computation has a hierarchical structure, which cannot be converted to the dominance-sum problem.
- Second, when inserting a new entry (i.e., a new product category) in the product category hierarchy, it may affect the allocated storage space of all the subtrees that are external to the hierarchy.

Now, let us consider two cases. Suppose that all the past transactions of a seller belong to m brand-based product categories (see Figure 3). Hence, m subtrees are generated corresponding to the m brand-based product categories. Note that, in this paper, the storage space used for storing aggregation index in the product category dimension, i.e., N-ary tree, will be ignored, and we assume the aggregation index formed by all the subtrees that are external to product category hierarchy consume relatively larger storage space. For Case 1, if the newly occurred transaction belongs to m existing brand-based product categories, it is unreasonable to allocate average storage space to each subtree, since a seller should have imbalanced number of transactions in each brand-based product category. For Case 2, if a newly occurred transaction belongs to a new product category on which the seller has no prior transactions, it is necessary to allocate additional storage space to store the aggregation index as to know the trust value of the seller in this product category or the corresponding sub-category. As a result, the insertion operation surely affects space allocation for the previous m subtrees.

To solve the afore-mentioned specific demands in CTT computation, we propose a method that dynamically allocates storage space for each subtree based on the number of transactions in the corresponding brand-based product category. Our approach guarantees the total storage space is fixed during allocation. In addition, in each subtree, hierarchical temporal aggregation will be adopted to further restrict the store space.

B. Our Proposed CTTFS Model

Our proposed CTTFS model consists of two parts: dynamical storage space allocation (Algorithm 1) and hierarchical temporal aggregation (Algorithm 2).

Algorithm 1: Dynamical storage space allocation

This algorithm aims to appropriately allocate storage space for aggregation index formed by all the subtrees that are external to product category hierarchy.

As illustrated in subsection IV-C, in our proposed tree structure for CTT computation (see Figure 3), each entry in the N-ary tree also records the number of ratings count_r and the sum of ratings sum_r in the corresponding product category. Suppose that the products sold by a seller in all past transactions belong to m brand-based product categories. Each brand-based product category includes count_rj (j ∈ 1, 2, ..., m) transactions. Thus, \( \sum_{j=1}^{m} \text{count}_r^j \) equals the total number of all past transactions. If the total storage size allocated to a seller is S, we further consider space allocation in two cases as illustrated before:

- **Case 1**: The newly occurred transaction belongs to an existing brand-based product category. For the existing m brand-based product categories, each subtree should
be allocated $\frac{\text{count}_{FS} \cdot S}{\sum_{j=1}^{m} \text{count}_{FSj}}$ ($j \in 1, 2, ..., m$) disk pages for storing aggregation index.

- **Case 2:** The newly occurred transaction belongs to a new brand-based product category. In such a case, there are $m + 1$ brand-based product categories, and each subtree should be allocated $\frac{\text{count}_{FS} \cdot S}{\sum_{j=1}^{m} \text{count}_{FSj}}$ ($j \in 1, 2, ..., m, m + 1$) disk pages for storing aggregation index.

Algorithm 2: Hierarchical temporal aggregation

However, each of two cases in Algorithm 1 leads to a problem of compression of the allocated storage space (i.e., disk pages). Hence, the general idea of the HTA	extsuperscript{FS} model will be adopted to deal with the subtree the size of which larger than the allocated storage space. For simplicity, we assume the allocated storage space for each subtree to store aggregation index is $S_j$. For $m$ brand-based product categories, we have $\sum_{j=1}^{m} S_j = S$.

If the size of the aggregation index of a subtree becomes larger than the corresponding allocated space $S_j$, we need to have a further division. As illustrated before, each subtree that is external to product category hierarchy maintains the aggregates $\text{count}_r$ and $\text{sum}_r$ in transaction amount and transaction time dimensions. At the beginning, each subtree uses a single finer time granularity $\text{gran}_2$ in the transaction time dimension. When performing hierarchical temporal aggregation operations, we choose a new dividing time $t_{\text{newdiv}}$. Here, different from HTA	extsuperscript{FS} model, the information larger than the size $S_j/2$ will be aggregated at a coarser time granularity $\text{gran}_1$. As shown in Figure 2(b), the CTT-FS model forms two segments $\text{seg}_1$ and $\text{seg}_2$ with different time granularities in the transaction time dimension. In addition, more time granularities $\text{gran}_k$ ($k > 2$) can be introduced to further compress the storage space.

For update operations in the CTT-FS model, we should first find all the transactions and corresponding ratings before $t_{\text{newdiv}}$ in $\text{seg}_2$. Then, we standardize these transactions and the corresponding ratings at a coarser time granularity $\text{gran}_1$. Finally, we move standardized results to $\text{seg}_1$. If the $\text{seg}_1$ is not null, $\text{seg}_1$ is updated; otherwise we directly move them to $\text{seg}_1$. The above operations are similar to Algorithm 1 proposed in the HTA	extsuperscript{FS} model [18].

C. eaR-tree	extsuperscript{FS} – A New eaR-tree with fixed Storage Space for CTT Computation

In this subsection, we use the eaR-tree as an example to explain how to apply our proposed CTT-FS model with fixed storage space in CTT computation. As illustrated before, based on ar-tree [2], [6], we proposed an eaR-tree for CTT computation [25]. In the eaR-tree, each subtree that is external to the product category hierarchy is an eaR-tree containing aggregates $\text{count}_r$ and $\text{sum}_r$. Next, we illustrate how to construct a new eaR-tree with fixed storage space (eaR-tree	extsuperscript{FS}) for CTT computation.

**Insertion:** When inserting a newly occurred transaction into the eaR-tree	extsuperscript{FS}, Algorithm 1 first reallocates storage space for the aggregation index formed by each subtree (i.e., ar-tree). If the size of aggregation index of a subtree larger than allocated space, then Algorithm 2 will be applied to restrict the storage space. For simplicity, in this paper, eaR-tree	extsuperscript{FS} only adopt 2-level time granularities and set the coarser time granularity at month level and the finer time granularity at day level. As a result, different from original eaR-tree, each subtree in eaR-tree	extsuperscript{FS} will be divided into 2 ar-trees logically with different time granularities (see $\text{seg}_1$ and $\text{seg}_2$ in Figure 2(b)). Therefore, for $m$ brand-based product categories, there are at most $2m$ ar-trees with limited storage space in the complete eaR-tree	extsuperscript{FS} model.

**Query:** In responding to a CTT query, in the new structure eaR-tree	extsuperscript{FS}, the search is first conducted on the N-ary tree, i.e., the product category hierarchy. However, there may have an additional query in a subtree of eaR-tree	extsuperscript{FS} that is external to the product category hierarchy depending on the query range in the transaction time dimension. That is because a subtree of eaR-tree	extsuperscript{FS} may have two logically ar-trees with different time granularities as illustrated above. For example, if a CTT query in transaction time dimension is within range $[t_{\text{newdiv}}, \text{now}]$, only an ar-tree in the subtree need to be searched in corresponding brand-based product category; otherwise we need to search 2 ar-trees in the subtree. In the following section, we will illustrate that the number of access nodes can be significantly reduced in our proposed new structure when responding to a CTT query.

VI. EXPERIMENTS

In this section, we introduce the experiments conducted on an eBay dataset and a large-scale synthetic dataset, which aim to evaluate the efficiency of our proposed model.

A. Datasets

eBay has released APIs, with which the detailed feedback and transaction data of sellers for up to 90 days can be accessed. We have developed a program using eBay APIs and obtained the transaction data with ratings of some popular sellers. We finally selected a seller $s_A$ who has the largest trading volume among these popular sellers. $s_A$ had around 12000 transactions (approx. 4000 transactions per month) within 90 days in total. The products sold by $s_A$ distribute in multiple product categories; but most products are in the category of ‘Information, Communication and Media technology’ (see Figure 3).

Meanwhile, our experiments are also conducted on a synthetic dataset based on $s_A$. In the synthetic dataset, we generated the transaction data for a given day at 10 times the rate as that of $s_A$, and then duplicated these transactions for 90 days to form the transaction data for a 12 month period. Thus, there are about 480,000 transactions in total. The synthetic dataset guarantees that each product sold in a year has the same proportion of occurrences as that in the eBay dataset.

\footnote{developer.ebay.com/support/docs}
B. Experiment Setup

The experiments compared the proposed eaR-treeFS (see Section V-C) with the original eaR-tree. The page size is set to 512 bytes. In addition, each of eaR-treeFS and eaR-tree is implemented using C++ running on a Lenovo Y560 laptop with an Intel Core i5 CPU (2.20GHz), 2GB RAM, Windows 7 Professional operation system and MySql 5.1.35 relational database.

We measure the number of accessed nodes, instead of computation time for computing CTT values in the trust vector (see subsection IV-B). This is because the difference in computation time is usually a few milliseconds or less. Though the difference looks minor, any improvement is meaningful as millions users may be accessing CTT values simultaneously. Due to space constraint, we only introduce the results for computing Product Category based Trust (PCT). For computing Transaction Item Specific Trust (TIST) and Similar Transaction Amount based Trust (STAT), it easily draw the same conclusion as the search processes for computing these two values are almost the same as PCT. The only difference is that in the computation of TIST, there is a need to further search the actual records in the database. In the computation of STAT, the search needs to be performed in all the categories of the products sold by the seller.

C. Results

Considering $s_A$ had numerous transactions in the product categories ‘Apple MP3 player (iPod)’ at layer 5 and ‘Audio device’ at layer 3 in the product category hierarchy (see Figure 3). Thus, we assume a scenario that a buyer plans to buy an ‘Apple mc526ll/a iPod nano 16GB’ for about $150 from this seller, and then he/she specifies and adjusts the ‘product category’ at the above two layers in category hierarchy.

Before presenting the experimental results, we need to emphasize that aggregation index cannot be compressed without any restriction in our model, i.e., the size of aggregation index for eaR-treeFS should be larger than the size of by-month aggregation in the transaction time dimension for eaR-tree. Meanwhile, the size of aggregation index for eaR-treeFS should be smaller than the size of original eaR-tree (aggregation by-day). Table I lists the sizes of aggregation index of the eaR-tree aggregated by-day and the eaR-tree aggregated by-month over eBay dataset and synthetic dataset, respectively.

**Results on the eBay dataset:** When computing the value of PCT (Product Category based Trust) on the eBay dataset, we also assume that the buyer specifies a transaction amount range of [$100,$200], and time ranges from ‘one month ago’ to ‘now’, from ‘two months ago’ to ‘now’, and from ‘three months ago’ to ‘now’, respectively. For the eaR-treeFS model, according to the Table I, we set the fixed storage space $S$ to be 100 KB and 150 KB, respectively.

As shown in Figure 4, for all three tree structures, when the time range in a query becomes larger, the number of accessed nodes increases linearly. This is in line with the nature of the traditional aR-tree [2], [6]. As the amount of space allocated to the eaR-tree decreases, i.e., from eaR-tree (150 KB) to eaR-tree (100 KB), relatively more transactions and ratings in history are aggregated by month. Thus, the complete index size becomes smaller, and the corresponding number of accessed nodes decreases. On average, compared with eaR-tree, the number of accessed nodes for computing PCT at layer 5 (see Figure 4(a)) decreases by 29% for eaR-treeFS(150 KB) and 44% for eaR-treeFS(100 KB). The number of accessed nodes for computing PCT at layer 3 (see Figure 4(b)) decreases by 16% for eaR-treeFS(150 KB) and 19% for eaR-treeFS(100 KB).

Furthermore, in practice, a buyer’s CTT query in the transaction time dimension usually has the granularity of ‘month’, for example, from ‘three months ago’ to ‘now’ or from ‘six months ago’ to ‘now’. It is essentially not necessary to differentiate the trust level of a seller between a half year and a half year plus one day. Therefore, our proposed CTTFS model is particularly meaningful and practical to CTT computation.

**Results on the synthetic dataset:** We also conducted the same experiments on the synthetic dataset, which contain transactions distributed over a 12 month period for comput-
ing the PCT, we assume that the buyer specifies time ranges from ‘3 months ago’ to ‘now’, from ‘6 months ago’ to ‘now’, and from ‘12 months ago’ to ‘now’, respectively. For the eaR-tree model, according to the Table I, we set the fixed storage space S to be 500 KB and 750 KB, respectively.

From the results plotted in Figures 5, we can draw the same conclusion as the experiments on the eBay dataset. On average, compared with eaR-tree, the number of accessed nodes for computing PCT at layer 5 (see Figure 5(a)) decreases by 68% for eaR-tree(500 KB) and 74% for eaR-tree(750 KB). Similarly, the number of accessed nodes for computing PCT at layer 3 (see Figure 5(b)) decreases by 71% for eaR-tree(750 KB) and 76% for eaR-tree(500 KB).

VII. CONCLUSION

In the literature, we have proposed some approaches to Contextual Transaction Trust (CTT) computation in e-commerce environments [23], [25]. However, they all adopted a single fine time granularity leading to large storage space consumption in real applications. To solve this problem, in this paper, we proposed a new model with fixed storage space for CTT computation. The model aggregates the past transactions and ratings at different time granularities, which provides a trade-off between aggregation hierarchies and storage space. From the result of our experiments conducted on both an eBay dataset and a synthetic dataset, we can draw a conclusion that, the new model not only restricts the allocated storage space, but also significantly reduces the number of accessed nodes in responding to buyers’ CTT queries. Thus, our proposed model is particular applicable to large-scale e-commerce and e-service websites.

REFERENCES


