# *Organizing XML Data in a Wireless Broadcast System by Exploiting Structural Similarity*

Wireless Pers Commun (2018) 98:1299–1329
https://doi.org/10.1007/s11277-017-4920-x

# Organizing XML Data in a Wireless Broadcast System by Exploiting Structural Similarity

**Yongrui Qin[1] · Quan Z. Sheng[2] · Hua Wang[3] · Nickolas J. G. Falkner[4]**

Published online: 30 August 2017


**Abstract** Wireless data broadcast is an efficient way of delivering data of common interest to a large population of mobile devices within a proximate area, such as smart cities, battle fields, etc. In this work, we focus ourselves on studying the data placement problem of periodic XML data broadcast in mobile and wireless environments. This is an important issue, particularly when XML becomes prevalent in today's ubiquitous and mobile computing devices and applications. Taking advantage of the structured characteristics of XML data, effective broadcast programs can be generated based on the XML data on the server only. An XML data broadcast system is developed and a theoretical analysis on the XML data placement on a wireless channel is also presented, which forms the basis of the novel data placement algorithm in this work. The proposed algorithm is validated through a set of experiments. The results show that the proposed algorithm can effectively place XML data on air and significantly improve the overall access efficiency.

**Keywords** Mobile computing · Periodic data broadcast · XML · Multi-item query · Data placement algorithm

## 1 Introduction

Wireless technologies have become deeply embedded in our daily lives [1]. At the end of 2011, there were 6 billion mobile subscriptions, estimated by the International Telecommunication Union [2]. That is equivalent to 87% of the world population, and is a huge increase from 5.4 billion in 2010 and 4.7 billion mobile subscriptions in 2009.

✉ Yongrui Qin
y.qin2@hud.ac.uk

[1] School of Computing and Engineering, University of Huddersfield, Huddersfield, UK

[2] Department of Computing, Macquarie University, Sydney, Australia

[3] College of Engineering and Science, Victoria University, Melbourne, Australia

[4] School of Computer Science, University of Adelaide, Adelaide, Australia

Broadcast is one of the basic ways of information access via wireless technologies. In a wireless data broadcast system, the server broadcasts public information to all mobile devices within its transmission range via a downlink broadcast channel. Mobile clients "listen" to the downlink channel and access information of their interest directly when the desired information arrives. Broadcast is bandwidth efficient because all mobile clients can share the same downlink channel and retrieve data from it simultaneously. Broadcast is also energy efficient at the clients because downloading data costs much less energy than sending data [3].

Wireless data broadcast services have been available as commercial products for many years (e.g., StarBand and Hughes Network). Recently, there has been a push for such systems from industry and various standard bodies. For example, born out of the ITU "IMT-2000" initiative, the Third Generation Partnership Project 2 is developing Broadcast and Multicast Service in CDMA2000 Wireless IP network. Systems for Digital Audio Broadcast (DAB) and Digital Video Broadcast (DVB) are capable of delivering wireless data services. Recent news also reported that XM Satellite Radio [4] and Raytheon have jointly built a communication system, known as the Mobile Enhanced Situational Awareness Network (MESA), that would use XM satellites to relay information to soldiers and emergency responders during a homeland security crisis.

On the other hand, information expressed in semi-structured formats is widespread over the past years. XML has rapidly gained popularity as a de facto standard to represent semi-structured information. Popular Web browsers provide support for XML and nearly all major IT companies (e.g., Microsoft, Oracle, and IBM) have integrated XML into their software products. Delivering information in XML format is also popular in Web services and in various kinds of Publish/Subscribe systems.

Combining both trends of the proliferation of mobile computing technologies and XML data, broadcasting information in XML format in a wireless environment would be a preferable way of information delivering and sharing. Consequently, the research of XML data broadcast is of great importance and in fact it has been attracting more and more research interests [5–12]. To further demonstrate the practicability of XML data broadcasts, a potential application by detailing a real life scenario will be presented in Sect. 3.

There are two typical data broadcast modes: (1) *Periodic Broadcast Mode* and (2) *On-Demand Broadcast Mode* [3]. In the periodic broadcast mode, data is periodically broadcasted on a downlink channel via which the server sends data to clients. Clients only need to "listen" to that channel and download the data that they are interested in. In contrast, for the on-demand broadcast mode, clients send their queries to the server via an uplink channel and the server considers all submitted requests and decides the content of the next broadcast cycle. This work is focused on the periodic broadcast mode since it has many benefits. Firstly, it can save uplink bandwidth as no requests will be sent from mobile clients, thereby reducing power consumption at the clients. Secondly, it can effectively deliver information to an unlimited number of clients simultaneously. This is because the number of mobile clients will not affect the workload on the server.

Data placement algorithms determine what data items to be broadcasted by the server and their order on wireless channels, aiming to reduce average waiting time for mobile clients. To a large extent, the data placement problem of XML data is similar to that in multi-item contexts [13, 14] where mobile clients may request multiple items each time. However, there are some drawbacks of existing data placement approaches in the traditional data broadcast models.

Firstly, the previous work on multi-item placement problems generally assumes that the clients' queries are known in advance [13–16]. For example, the clients can provide a

profile of their interests to the servers [15, 16]. However, such models may limit the practicability of the proposed placement algorithms in real situations because: (1) new mobile clients may join in the network at anytime; and (2) mobile users may be reluctant to disclose their queries to the server via an uplink channel due to expensive communication cost and privacy concerns. Secondly, in the traditional data broadcast systems, data items themselves on the server are normally independent and it is difficult to discover underlying relationships between data items, which means user queries must be known in advance for the design of data placement algorithms. Alternatively, the authors of [17] applied data mining techniques to discover association rules from the history access patterns of a set of data. This avoids to obtain access patterns of mobile clients on-the-fly. However, in this approach, the availability of such history access patterns of mobile clients is a necessity, which may not always be available.

By contrast, in XML data broadcast, data items (or XML documents) usually share parts of their structure. Taking structural sharing between XML documents into consideration, it becomes feasible to analyze and estimate clients' access patterns. Then XML data can be effectively placed on wireless channels based only on the XML data on the server because the server can generate XML data placement according to the structural similarity among XML data. In this way, user queries that focus on structural information matching can benefit from such data placement. To the best of our knowledge, little work has addressed similar data placement strategies in the context of wireless data broadcast. Therefore, this paper studies the data placement problem of periodic XML data broadcast. Firstly, an XML broadcast system is described and a theoretical analysis on the data placement problem of periodic XML data broadcasts is presented. Secondly, based on the analysis, a novel greedy data placement algorithm is designed. In a nutshell, the main contributions of this paper are summarized as follows:

- In the context of periodic XML data broadcasts, by taking advantage of the structural characteristics of XML data, it is shown to be feasible to generate appropriate data placement results based only on the XML data on the server.
- A theoretical analysis on the data placement problem of periodic XML data broadcasts is presented. Based on the analysis, a novel greedy data placement algorithm which organizes XML data on air is proposed.
- Extensive experiments are conducted to show the effectiveness of the proposed data placement algorithm.

A preliminary version of this paper appeared in [18], where the emphasis is focused on the introduction of the data placement algorithms and preliminary experiments. This paper significantly extends the work in [18] from the following aspects, including: (1) a detailed theoretical analysis on the data placement problem is presented, which forms the basis of the data placement algorithms; (2) a more detailed and rigorous time complexity analysis of the proposed data placement algorithms is provided; (3) a more detailed introduction of Index Distribution Strategy facilitating the understanding of used indexing schemes in this proposed work is added; (4) experiments on two more data sets and more experimental results are presented to further show the applicability and low maintenance cost of the proposed data placement algorithms.

The remainder of this paper is organized as follows. Section 2 describes some background information of this work, including an application scenario, the wireless broadcast system and the concept of XML structural sharing. Then a theoretical analysis of the data placement problem is presented in Sect. 3. Section 4 discusses the structural sharing property of XML data and then proposes a novel greedy data placement algorithm.

Section 5 presents the related experimental study for evaluating the performance of the proposed data placement algorithm. Finally, Sect. 6 discusses the related work and Sect. 7 gives some concluding remarks.

## 2 Background

In this section, a potential application scenario is firstly described. Then an overview of the broadcast system used in this work is provided and background knowledge of XML structural sharing is also introduced.

### 2.1 Application Scenario

The following scenario can be used to show potential applications of XML data broadcast in real life.

Consider a live basketball game. Information about the game and the players on the court is usually the interest of a large number of audience. When the game progresses, the volume of such information is expected to increase, which means the information content is dynamic. Normally, a basketball stadium can accommodate 10,000 to 60,000 audience at the same time. This large number of audience are likely to be interested in similar game information. They may even want to access the game information at the same time (suppose most of them are with mobile devices). In this context, data broadcast is a preferable way of delivering latest information to the audience. Then thousands of audience in the stadium can access game information simultaneously by just "listening" to the broadcast channel. Audience do not need to contend limited bandwidth (i.e., the use of the uplink channel) and other system resources (i.e. the server processing capacity) with each other.

In this scenario, it is assumed that (1) the audience are not only interested in the real-time information about the basketball game, but also interested in some historical records/statistics information about the players, the basketball teams or the coach teams, etc; (2) the audience may want more statistics information about the current basketball game than the limited live statistics information shown on the large screens inside the stadium. Therefore, a broadcast service would be very helpful for the audience to obtain more desired information about the game.

Meanwhile, audience could be outside of the stadium, such as basketball fans who are watching live text information about the game via the Internet at their homes. Therefore, the game information could also be delivered via the Internet to online audience and other Web service providers who have subscribed this basketball game. Using XML format to represent game information can satisfy all these needs and realize simplicity, generality, and usability of game information at the same time.

### 2.2 A Wireless XML Data Broadcast System

Figure 1 shows a wireless XML data broadcast system. The system includes an XML Data Center (the broadcast server), a broadcast program scheduler, broadcast listeners (mobile clients) and a downlink channel (the server broadcasts information to mobile clients via it). If mobile clients are interested in some data on the server, they can listen to the downlink channel and download data that they need. Note that, downloading the data from the

downlink channel does not mean each mobile client would need to set up a different downlink connection. All mobile clients only need to tune in the downlink channel and listen to it for the desired information. Thus this downlink channel can be shared by all mobile clients. Mobile clients cannot send their individual queries to the server in this model as no uplink channel is available.

There are several notable advantages of such a system model. Firstly, it can serve an arbitrary large number of mobile clients simultaneously. Secondly, extra energy cost at the client side for using an uplink channel can be avoided. Further, after applying air indexing techniques [8, 19], mobile clients will also be able to determine when the desired data will be broadcasted on the wireless channel (for more details about air indexing, please refer to Sect. 4.3). Before the desired data is available, they can switch to energy saving mode to reduce power consumption. As a result, the battery life of mobile clients can be extended.

From the figure, it can be seen that the XML Data Center could be connected to the Internet and deliver information to online users, Web service providers and Publish/Subscribe systems, etc. With the use of XML data, these different applications can be integrated seamlessly with the wireless data broadcast system for the purpose of sharing and delivering the same information to different users.

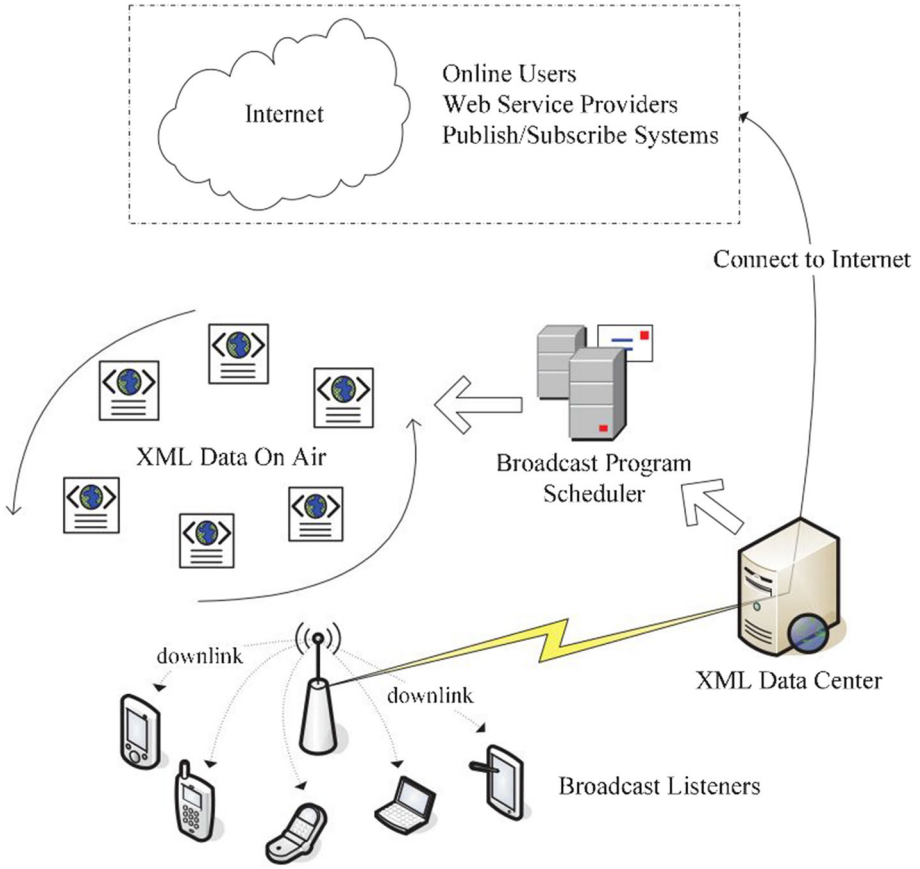



**Fig. 1** A wireless XML data broadcast system

### 2.3 XML Structural Sharing

The goal of this work is to place XML documents on the broadcast channel based on the information at the server side. This work proposes to explore structural sharing between different XML documents and place documents according to the structural sharing results. Here, structural sharing refers to overlaps of path sets (defined in the below) of XML documents.

Some existing work on measuring structural sharing between XML documents can be found in [20, 21]. The main idea of their work is based on the concept of *path sets*. Here, a path set of an XML document contains all full paths (paths that are from the root element to the leaves) and their subpaths. A simple example is depicted in Fig. 2. The path set of this example is: {`/player/name, /player/position, /player/nationality, /player/ college, /player, /name, /position, /nationality, /college`}. A path set of an XML document $d$ is denoted as $PS(d)$, while $|PS(d)|$ denotes the number of paths in $PS(d)$.

Different types of metric can be adopted, such as Jaccard metric [22, 23], Dice's coefficient [24] and Lian's metric [25], to measure the structural sharing or similarity between two XML documents $d_i$ and $d_j$. The exact forms of these metrics based on $PS$ are as follows (Jaccard metric is denoted as $J(d_i, d_j)$, Dice's coefficient is denoted as $D(d_i, d_j)$ and Lian's metric is denoted as $L(d_i, d_j)$):

$$J(d_i, d_j) = \frac{|PS(d_i) \bigcap PS(d_j)|}{|PS(d_i) \bigcup PS(d_j)|} \tag{1}$$

$$D(d_i, d_j) = \frac{2 \cdot |PS(d_i) \bigcap PS(d_j)|}{|PS(d_i)| + |PS(d_j)|} \tag{2}$$

$$L(d_i, d_j) = \frac{|PS(d_i) \bigcap PS(d_j)|}{max\{|PS(d_i)|, |PS(d_j)|\}} \tag{3}$$

Equation (1) computes the proportion of common paths over the union of all paths in two given XML documents $d_i$ and $d_j$, Eq. (2) computes the result of twice of the number of common paths over the total number of all paths in the two given XML documents, and Eq. (3) computes the proportion of common paths over all the paths in the larger path set of the two given XML documents. From the above definitions, it can be seen that both Jaccard metric and Dice's coefficient give more weights on the total structural information of two comparing documents while Lian's metric emphasizes on the difference of these documents. All three metrics can vary in the interval [0, 1]. If $PS(d_i) = PS(d_j)$, then $J(d_i, d_j) = D(d_i, d_j) = L(d_i, d_j) = 1$. Clearly, the larger the values of these metrics are, the

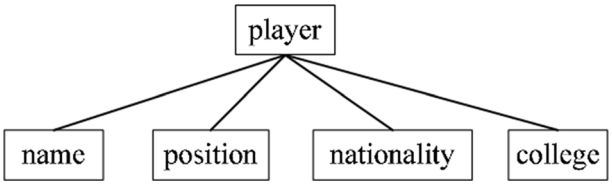



**Fig. 2** An XML structure tree

more structural sharing the two comparing XML documents have. From these equations, we can have a better understanding about what kind of structural information a specific metric emphasizes on. Further, on top of these metrics, we will introduce a new metric that is more suitable for assisting data placement of XML data in wireless broadcast later in this paper.

## 3 Analysis of the Data Placement Problem

In this section, a theoretical analysis on the data placement problem in periodic XML data broadcasts is presented.

In the literature, two critical metrics, namely *access time* and *tuning time*, are used to measure a system's performance [26]. *Access time*[1] refers to the time elapsed from the moment a query is issued to the moment it is answered, while *tuning time* refers to the time a mobile client stays in active mode to receive the requested XML documents and the index information. Data placement mainly affects access time because tuning time depends on the total content downloaded by mobile clients but not on the order of data. Hence, access time is used as the metric in this analysis. In periodic broadcast, queries are used to describe the interests of mobile clients and help mobile clients to skip irrelevant data on air, but they are not actually submitted to the broadcast server.

Table 1 lists the symbols used in the rest of the paper and Fig. 3 shows a broadcast program (or a broadcast sequence) $\sigma$ on the wireless channel which is broadcasted periodically. The broadcast program $\sigma$ can start from any XML document $d_i$. However, it is assumed that $\sigma$ starts from $d_1$ (this will then comply with the definition of $\sigma$ in Table 1) to simplify the analysis.

With the basic assumption that queries can be issued at any time with an equal probability (this means the issue time of queries follows a uniform distribution), the expected access time of $q$, denoted as $\mathcal{AT}^q_{exp}$, can be computed in the following:

$$\begin{aligned}\mathcal{AT}^q_{exp} &= \sum_{i=1}^{k}\left(\frac{\mathcal{L}_{d_{n_i}}}{\mathcal{L}_\sigma}\cdot\mathcal{L}_\sigma + \frac{\mathcal{L}_{gap_i}}{\mathcal{L}_\sigma}\cdot\left(\mathcal{L}_\sigma - \frac{1}{2}\cdot\mathcal{L}_{gap_i}\right)\right)\\ &= \sum_{i=1}^{k}\mathcal{L}_{d_{n_i}} + \sum_{i=1}^{k}\mathcal{L}_{gap_i} - \frac{1}{\mathcal{L}_\sigma}\cdot\sum_{i=1}^{k}\frac{1}{2}\cdot\mathcal{L}^2_{gap_i}\\ &= \mathcal{L}_\sigma - \frac{1}{2\cdot\mathcal{L}_\sigma}\cdot\sum_{i=1}^{k}\mathcal{L}^2_{gap_i}\end{aligned} \tag{4}$$

In this equation, $\mathcal{L}_{d_{n_i}}$ refers to the length of document $d_{n_i}$, $\mathcal{L}_\sigma$ refers to the total length of all documents in $\sigma$ and $\mathcal{L}_{gap_i}$ refers to the total length of all documents in $gap_i$.

It should be noted that, we derive the first line of Eq. (4) in the following:

1. Case 1: $q$ is issued when $d_i$ is broadcast. The probability of this case is $\frac{\mathcal{L}_{d_{n_i}}}{\mathcal{L}_\sigma}$ and the total access time will be the whole broadcast cycle, which is $\mathcal{L}_\sigma$.
2. Case 2: $q$ is issued in the middle of broadcasting $gap_i$. The probability of this case is $\frac{\mathcal{L}_{gap_i}}{\mathcal{L}_\sigma}$ and the average access time will be $\mathcal{L}_\sigma - \frac{1}{2}\cdot\mathcal{L}_{gap_i}$.

[1] Note that, in [19], *Access latency* is also used for *Access time*, which can be considered the same concept.

**Table 1** Symbols overview

| Symbol | Description |
|---|---|
| $\mathcal{D}$ | XML document set. $\{d_1, d_2, d_3, \ldots, d_n\}$ |
| $q$ | Query issued by a mobile client |
| $k$ | Number of documents required by $q$ |
| $\sigma$ | A complete broadcast program (or broadcast sequence). It is the result of a data placement algorithm running on a given $\mathcal{D}$. $<d_1, d_2, d_3, \ldots, d_n>$. (Note: $\sigma$ is different from $\mathcal{D}$ as $\mathcal{D}$ is a set of documents but $\sigma$ is a sequence of the same set of documents.) |
| $gap$ | Unmatched documents of $q$ that are placed between two adjacent matched documents in $\sigma$ |
| $\mathcal{L}$ | The length of documents |
| $\mathcal{AT}^q_{exp}$ | The expected access time of $q$ |

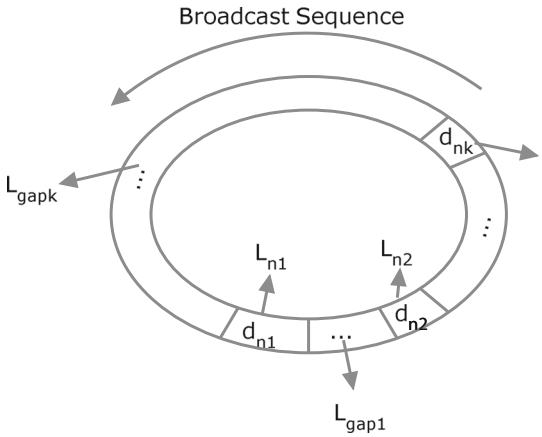



**Fig. 3** A broadcast program showing positions of documents required by query $q$

According to Eq. (4)[2] and a given broadcast program $\sigma$, $\mathcal{AT}^q_{exp}$ can be calculated simply according to the gaps between consecutive documents required by $q$. Further, from the above equation, it can be seen that in order to improve the expected access efficiency, $\sum_{i=1}^{k} \mathcal{L}^2_{gap_i}$ should be as large as possible.

Moreover, the sum of all gaps, denoted as $\mathcal{L}_{gaps}$, can be computed as

$$\mathcal{L}_{gaps} = \sum_{i=1}^{k} \mathcal{L}_{gap_i} = \mathcal{L}_{\sigma} - \sum_{i=1}^{k} \mathcal{L}_{d_{n_i}} = \mathcal{L}_{\sigma} - \mathcal{L}_{\sigma_q} \tag{5}$$

Here, $\mathcal{L}_{\sigma_q}$ refers to the total length of all documents required by $q$, which is $\sum_{i=1}^{k} \mathcal{L}_{d_{n_i}}$. Note that $\mathcal{L}_{\sigma_q}$ is independent of any data placement results. In other words, $\mathcal{L}_{\sigma_q}$ is fixed for a given $q$, which in turn indicates that $\mathcal{L}_{gaps}$ is fixed.

[2] This result is exactly the same as [27] although the deduction process is different. The further analysis on this result in the following is new.

In order to derive the lower and upper bounds of $\sum_{i=1}^{k} \mathcal{L}_{gap_i}^2$ and to analyze the data placement strategy, the following propositions for the below function $f(\mathbf{X})$ are firstly presented.

Function $f(\mathbf{X}) = \sum_{i=1}^{k} x_i{}^2$ is with the following constraints:

1. $x_1 + x_2 + \cdots + x_k = M$
2. $x_1, x_2, \ldots, x_k \geq 0$

where $M$ is a positive constant. The lower bound and the upper bound of $f(\mathbf{X})$ can be denoted as $\underline{f(\mathbf{X})}$ and $\overline{f(\mathbf{X})}$, respectively.

**Proposition 1** *Given $f(\mathbf{X})$ defined as above, then*

$$f(\mathbf{X}) \leq M^2$$

*When $x_1 = x_2 = \cdots = x_{k-1} = 0$ and $x_k = M$ (or any other kind of combinations with $k-1$ zeros and only one $M$), $f(\mathbf{X})$ reaches its upper bound, i.e., $\overline{f(\mathbf{X})} = M^2$.*

**Proposition 2** *Given $f(\mathbf{X})$ defined as above, then*

$$f(\mathbf{X}) \geq \frac{M^2}{k}$$

*When $x_1 = x_2 = \cdots = x_k = \frac{M}{k}$, $f(\mathbf{X})$ reaches its lower bound, i.e., $\underline{f(\mathbf{X})} = \frac{M^2}{k}$.*

Note that, the proof of Proposition 1 and 2 is in the "Appendix".

Moreover, given $f(\mathbf{X})$ defined as above and suppose that $m$ variables, i.e. $x_1, x_2, \ldots, x_m$, have been determined ($m < k$) while the rest $k-m$ variables are not. $M - \sum_{i=1}^{m} x_i$ can be denoted as $M'$. The next step is to determine the next variable. Without loss of generality, $x_{m+1}$ can be used as the next variable to be determined, which aims to maximize or minimize $f(\mathbf{X})$. $f(\mathbf{X})_{x_{m+1}}$ can be denoted as the function with $m+1$ determined variables ($x_i$, $1 \leq i \leq m+1$) and $k-m-1$ undetermined variables. Then given two values of this variable, i.e. $x_{m+1}$ and $x'_{m+1}$ and suppose $x_{m+1} < x'_{m+1}$, the following propositions are true:

**Proposition 3** *For $\overline{f(\mathbf{X})}_{x_{m+1}}$ and $\overline{f(\mathbf{X})}_{x'_{m+1}}$, it is true that*

$$\begin{cases} \overline{f(\mathbf{X})}_{x_{m+1}} > \overline{f(\mathbf{X})}_{x'_{m+1}} & x_{m+1} < x'_{m+1} \leq \dfrac{M'}{2} \\ \overline{f(\mathbf{X})}_{x_{m+1}} < \overline{f(\mathbf{X})}_{x'_{m+1}} & \dfrac{M'}{2} \leq x_{m+1} < x'_{m+1} \\ \textit{Indefinite} & \textit{Otherwise} \end{cases}$$

**Proposition 4** *For $\underline{f(\mathbf{X})}_{x_{m+1}}$ and $\underline{f(\mathbf{X})}_{x'_{m+1}}$, it is true that*

$$\begin{cases} \underline{f(\mathbf{X})}_{x_{m+1}} > \underline{f(\mathbf{X})}_{x'_{m+1}} & x_{m+1} < x'_{m+1} \leq \dfrac{M'}{k-m} \\ \underline{f(\mathbf{X})}_{x_{m+1}} < \underline{f(\mathbf{X})}_{x'_{m+1}} & \dfrac{M'}{k-m} \leq x_{m+1} < x'_{m+1} \\ \textit{Indefinite} & \textit{Otherwise} \end{cases}$$

The proof of these propositions can be found in the "Appendix". Now according to Propositions 1 and 2, it is true that

$$\frac{\mathcal{L}_{gaps}^2}{k} \leq \sum_{i=1}^{k} \mathcal{L}_{gap_i}^2 \leq \mathcal{L}_{gaps}^2 \tag{6}$$

Then according to Eq. (4), it is true that

$$\mathcal{L}_\sigma - \frac{\mathcal{L}_{gaps}^2}{2 \cdot \mathcal{L}_\sigma} \leq \mathcal{AT}_{exp}^q \leq \mathcal{L}_\sigma - \frac{\mathcal{L}_{gaps}^2}{2 \cdot k \cdot \mathcal{L}_\sigma} \tag{7}$$

From the above two inequations, it can be seen that in order to improve the expected access efficiency, $\sum_{i=1}^{k} \mathcal{L}_{gap_i}^2$ should be as large as possible. According to Proposition 1, when one of the gaps equals to $\mathcal{L}_{gaps}$ and all other gaps equal to 0, the best expected access efficiency can be achieved. Thus, when all XML documents required by $q$ are placed together and broadcasted in sequence, $\mathcal{AT}_{exp}^q$ can be minimized. Also, according to Eq. (5), the above inequation can be rewritten to

$$\mathcal{L}_\sigma - \frac{(\mathcal{L}_\sigma - \mathcal{L}_{\sigma_q})^2}{2 \cdot \mathcal{L}_\sigma} \leq \mathcal{AT}_{exp}^q \leq \mathcal{L}_\sigma - \frac{(\mathcal{L}_\sigma - \mathcal{L}_{\sigma_q})^2}{2 \cdot k \cdot \mathcal{L}_\sigma} \tag{8}$$

Here Inequation (8) shows both the lower and upper bounds of $\mathcal{AT}_{exp}^q$ for $q$ in another form. It is worth mentioning that both bounds are independent of any data placement results. Moreover, it can be inferred that when $k$ increases, $\sigma_q$ will include more documents. Then $\mathcal{L}_{\sigma_q}$ increases as well. However, the decrease of difference $\mathcal{L}_\sigma - \mathcal{L}_{\sigma_q}$ leads to larger lower and upper bounds of $\mathcal{AT}_{exp}^q$, which means the system's overall performance will degrade.

The above analysis focuses on a single query. However, generalizing it to multiple queries would be much more complicated. Actually, determining an optimal broadcast sequence for multiple multi-item queries is an NP-Complete problem [27].

When there are multiple queries to consider for a broadcast program, these queries are not likely to require the same XML documents. In such cases, Proposition 1 and Inequation (6), which minimizes expected access time for a single query, cannot help to find an optimal solution for all queries.

However, from Inequation (6) and (8) it can be inferred that the upper and lower bounds of $f(\mathbf{X})$ should be as large as possible which can lead to larger probability of having large results of $f(\mathbf{X})$. This in turn enlarges the probability that $\mathcal{AT}_{exp}^q$ is smaller for $q$ according to Inequation (6) and (8). Then, for multiple queries, according to Propositions 3 and 4, when $x_{m+1} \leq \frac{M'}{2}$ and $x_{m+1} \leq \frac{M'}{k-m}$, it is desirable to decrease $x_{m+1}$ to have larger $\overline{f(\mathbf{X})}_{x_{m+1}}$ and $\underline{f(\mathbf{X})}_{x_{m+1}}$. In other words, if each gap can be progressively reduced as much as possible, larger lower and upper bounds of $\sum_{i=1}^{k} \mathcal{L}_{gap_i}^2$ can be obtained. In this way, both the lower and upper bounds of the overall expected access time can be reduced with a higher probability.

For example, if it is required to minimize $\sum_{i=1}^{k} \mathcal{L}_{gap_i}^2$ for each query in $\{q_1, q_2, q_3\}$, for the first step, the system should place XML documents that are required by all three queries together to form an initial broadcast program. In the second step, the system should place XML documents required by only two of the three queries together and append them to the

initial broadcast program. After that, the system should append XML documents required by only one query to the broadcast program to form a final broadcast program. Hence, the system can construct a final broadcast program in a greedy style.

Now the problem becomes how the system can determine which documents should be placed together first as the system cannot obtain queries in advance. Related solution will be discussed in the next section.

## 4 The Data Placement Algorithm

This section introduces the data placement algorithm for periodic XML data broadcasts, based on the theoretical analysis in the previous section. Firstly the structural sharing property of XML data is discussed, which is used to estimate the potential access patterns of mobile clients, i.e., the probability of accessing a small set of similar XML documents simultaneously. Then a novel greedy data placement algorithm is put forward based on it.

### 4.1 Structural Sharing in XML Data

Intuitively, for any two given XML documents, the system can utilize one of the three structural similarity metrics described in Sect. 2.3 to calculate the similarity between them and the similarity results can be used to approximate the probability that a specific query is matched with both documents at the same time. For example, if two XML elements are under structurally similar paths, then it is more likely that that either both elements satisfy, or none satisfies, a given query [20]. Therefore, if two XML documents are with larger structural similarity, i.e. $d_1$ and $d_2$, then they would have a higher probability to be required simultaneously. However, there are still three other cases to be considered, such as requiring $d_1$ but not $d_2$, requiring $d_2$ but not $d_1$ and requiring neither of $d_1$ and $d_2$. Therefore, the above similarity metrics consider only successful match probabilities of both XML documents and do not consider unsuccessful match probabilities.

Nonetheless, unsuccessful match cases have effects on the expected access time as well. According to Propositions 1 and 2, in order to have better access efficiency, the gaps between any two required documents by a single query should be as less uniform as possible. Based on this, it can be inferred that in the above example, cases of required $d_1$ but not $d_2$ and required $d_2$ but not $d_1$ are likely to generate more uniform gaps while other two cases (required both documents or neither) are likely to have less uniform gaps. Observing this, a new similarity metric called *Cohesion* is defined to give a more accurate estimation of access patterns of mobile clients in the following.

Note that, for any query $q$ requiring at least one of the documents in $\mathcal{D}$, $q$ must match some paths in $PS(\mathcal{D})$ and it has a probability of $\frac{|PS(d)|}{|PS(\mathcal{D})|}$ to match $d$. If a query $q$ fails to match any document in $\mathcal{D}$, the issuer of $q$ only needs to locate and download air index[3] to confirm that his/her query does not match any document. Then he/she can stop waiting for the result to be broadcasted. All these queries only need to access the index information on air and therefore, their expected access time depends heavily on the index distribution, which is not the focus of this work. To estimate their expected access time, interested readers are

[3] The basic idea of air index is that the broadcast server pre-computes index information (including searchable attributes and delivery time of data items) and interleaves it with data items (e.g., XML documents) on the broadcast channel. For more details, please refer to [19].

referred to [19] for more details. Hence, this work only considers successful queries. This is because the access time of unsuccessful queries will depend on the indexing method but not on the data placement results. That means data placement algorithms will not affect the access time of unsuccessful queries.

Now suppose there is a set of $n$ XML documents $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$ on the server, the access probability of any document $d$ for queries which successfully match at least one document in the set $\mathcal{D}$ can be approximated as follows:

$$Pr(d) = \frac{|PS(d)|}{|PS(\mathcal{D})|} \tag{9}$$

and for any $i, j$ $(1 \leq i, j \leq n)$

$$Pr(d_i - d_j) = \frac{|PS(d_i) - PS(d_j)|}{|PS(\mathcal{D})|} \tag{10}$$

Here, $PS(\mathcal{D}) = \bigcup_{i=1}^{n} PS(d_i)$. According to this equation, $Pr(d_i - d_j)$ refers to the probability of a given query matching $d_i$ but not matching $d_j$.

There would be many different matching cases for a given set $\mathcal{D}$. Take two XML documents $d_1$ and $d_2$ in $\mathcal{D}$ as an example. As mentioned previously, there would be four cases of matching of them and the probability of each case is shown in Table 2. This table also includes positive and negative effects on the expected access time ($AT_{exp}$) for each case. Positive effects refer to the situation that, if we put the two documents with high probability $Pr(d_1 \bigcap d_2)$ or $1 - Pr(d_1 \bigcup d_2)$ together, according to our previous analysis, we should expect to achieve smaller $AT_{exp}$. In terms of negative effects, it means if we put the two documents with high probability $Pr(d_1 - d_2)$ or $Pr(d_2 - d_1)$, we would expect to achieve larger $AT_{exp}$.

Based on Table 2 and the previous analysis, it can be seen that, given two documents, if both documents are matched against the same query or neither of the documents is matched, the two documents will tend to be more similar to each other, i.e., $Pr(d_1 \bigcap d_2)$ or $1 - Pr(d_1 \bigcup d_2)$ is higher. On the other hand, if only one of the two documents is matched against a given query, the similarity between them tends to be smaller, i.e., $1/Pr(d_1 - d_2)$ or $1/Pr(d_2 - d_1)$ is smaller. Based on this observation, Cohesion $C(d_i, d_j)$ of XML documents $d_i$ and $d_j$ is defined as follows:

$$C(d_i, d_j) = \frac{Pr(d_i \bigcap d_j) \cdot (1 - Pr(d_i \bigcup d_j))}{max\{Pr(d_i - d_j), Pr(d_j - d_i)\}} \tag{11}$$

Here $d_i$ and $d_j$ are both in set $\mathcal{D}$. It is easy to see that $C(d_i, d_j) = C(d_j, d_i)$. According to Eqs. (9), (10) and (11), $C(d_i, d_j)$ can be calculated after finding path sets of $d_i$, $d_j$ in $\mathcal{D}$. Cohesion values can vary in a wide range which exceeds interval [0, 1]. Strictly speaking, Cohesion values only vary in interval $[0, \frac{|PS(\mathcal{D})|}{4}]$ if we define that $C(d_i, d_j) = \frac{|PS(\mathcal{D})|}{4}$ when $PS(d_i) = PS(d_j)$. The reason of doing so is: if $PS(d_i) \neq PS(d_j)$, the upper bound should be $\frac{|PS(\mathcal{D})|}{4}$ as shown in the following. The lower bound 0 is trivial. In order to obtain the upper bound, this work only considers cases that have $PS(d_i) \neq PS(d_j)$, from which it can be inferred that $max\{|PS(d_i - d_j)|, |PS(d_j - d_i)|\} \geq 1$, and thus, we have $max\{Pr$

**Table 2** Matching cases for document $d_1$ and $d_2$ in a document set $\mathcal{D}$

| Case | Probability | Effect on $AT_{exp}$ |
|---|---|---|
| Matched both $d_1, d_2$ | $Pr(d_1 \bigcap d_2)$ | Positive |
| Matched none of $d_1, d_2$ | $1 - Pr(d_1 \bigcup d_2)$ | Positive |
| Matched $d_1$, but not $d_2$ | $Pr(d_1 - d_2)$ | Negative |
| Matched $d_2$, but not $d_1$ | $Pr(d_2 - d_1)$ | Negative |

$(d_i - d_j), Pr(d_j - d_i)\} \geq \frac{1}{|PS(\mathcal{D})|}$. Without loss of generality, let $|PS(d_i)| \geq |PS(d_j)|$, according to Eqs. (9) and (10), (11) can be rewritten as follows:

$$\begin{aligned} C(d_i, d_j) &\leq \frac{\frac{|PS(d_i \bigcap d_j)|}{|PS(\mathcal{D})|} \cdot (1 - \frac{|PS(d_i \bigcup d_j)|}{|PS(\mathcal{D})|})}{\frac{1}{|PS(\mathcal{D})|}} \\ &< \frac{|PS(d_i)| \cdot (|PS(\mathcal{D})| - |PS(d_i)|)}{|PS(\mathcal{D})|} \\ &= \frac{-(|PS(d_i)| - \frac{|PS(\mathcal{D})|}{2})^2 + \frac{|PS(D)|^2}{4}}{|PS(\mathcal{D})|} \\ &\leq \frac{|PS(\mathcal{D})|}{4} \end{aligned}$$

Then the above result gives the upper bound of Cohesion $C(d_i, d_j)$. From this upper bound, we can infer that Cohesion $C(d_i, d_j)$ may reach $\frac{|PS(\mathcal{D})|}{4}$ when (1) $d_j$ is very small and has little structural overlapping with $d_i$, and (2) $d_i$ is big and contains half of the paths of all documents. Now Cohesion values can be normalized to interval [0, 1] in the following

$$C'(d_i, d_j) = \begin{cases} \frac{4 \cdot C(d_i, d_j)}{|PS(\mathcal{D})|} & PS(d_i) \neq PS(d_j) \\ 1 & PS(d_i) = PS(d_j) \end{cases} \quad (12)$$

It can also be inferred that $C'(d_i, d_j) = 1$ if and only if $PS(d_i) = PS(d_j)$. Similar to the other three similarity metrics, the larger the value of Cohesion is, the more structural sharing the two comparing XML documents have.

**Algorithm 1** Initialize structural sharing matrix $S[n][n]$

**Input:** A set of XML documents $\mathcal{D} : \{d_1, d_2, ..., d_n\}$
**Output:** Structural sharing matrix $S[n][n]$
1. create matrix $S[n][n]$
2. **for** each document $d$ in $\mathcal{D}$ **do**
3. compute $PS(d)$
4. **end for**
5. **for** each pair of documents $< d_i, d_j >$ in $\mathcal{D}$ $(i < j)$ **do**
6. $S[i][j] \Leftarrow$ structural sharing between $d_i$ and $d_j$
7. $S[j][i] \Leftarrow S[i][j]$
8. **end for**

**Algorithm 2** GDPA

**Input:** Structural sharing matrix $S[n][n]$
**Output:** A broadcast program $\sigma$ for $\mathcal{D}$

1. $\sigma \Leftarrow$ empty sequence
2. select a pair of documents $< d_i, d_j >$ with maximum value $S[i][j]$ in matrix $S[n][n]$
3. **if** $L_{d_i} <= L_{d_j}$ **then**
4. add $< d_i, d_j >$ into $\sigma$
5. **else**
6. add $< d_j, d_i >$ into $\sigma$
7. **end if**
8. $\mathcal{D}' \Leftarrow \mathcal{D} - d_i - d_j$
9. **while** $\mathcal{D}'$ is not empty **do**
10. $d_{head} \Leftarrow$ the first document in $\sigma$
11. select a pair of documents $< d_{i_{max}}, d_{head} >$ with maximum value $S[i_{max}][head]$ $(d_{i_{max}} \in \mathcal{D}')$
12. $d_{rear} \Leftarrow$ the last document in $\sigma$
13. select a pair of documents $< d_{j_{max}}, d_{rear} >$ with maximum value $S[j_{max}][rear]$ $(d_{j_{max}} \in \mathcal{D}')$
14. **if** $S[i_{max}][head] >= S[j_{max}][rear]$ **then**
15. append $d_{i_{max}}$ into $\sigma$ from head
16. $\mathcal{D}' \Leftarrow \mathcal{D}' - d_{i_{max}}$
17. **else**
18. append $d_{j_{max}}$ into $\sigma$ from rear
19. $\mathcal{D}' \Leftarrow \mathcal{D}' - d_{j_{max}}$
20. **end if**
21. **end while**

### 4.2 The Greedy Data Placement Algorithm

Based on the discussions of structural sharing between XML documents, it becomes feasible to generate a broadcast program for periodic data broadcasts in a greedy way. From previous discussions, if more structural sharing of two XML documents is observed, the probability to match both XML documents simultaneously will become larger. As a result, the Greedy Data Placement Algorithm (GDPA) places XML documents with most structural sharing together first as an initial broadcast program. Then it progressively appends other XML documents to the broadcast program in a descendant order of structural sharing. Detailed steps of GDPA are shown in Algorithms 1 and 2.

Algorithm 1 initializes a structural sharing matrix $S[n][n]$ for $n$ XML documents on the broadcast server. Note that, all four similarity metrics defined in Sects. 2.3 and 4.1 can be used in Algorithm 1 to compute structural sharing between two documents (Line 6). All of them are symmetric which means for any one of these metrics, it must be true that $S[j][i] = S[i][j]$. Also it is true that $J(d_i, d_j) = D(d_i, d_j) = L(d_i, d_j) = C'(d_i, d_j) = 1$ if $i = j$. Therefore, the algorithm only needs to calculate matrix $S$ for entries $S[i][j]$ where $i < j$.

Algorithm 2 adds the most similar XML document to the broadcast program at each time. It finds the most similar document based on structure similarity to the head and rear documents in the current broadcast program. That means, it will compare documents with the head and rear documents and then select the document with maximum similarity value. To be specific, based on matrix $S$, Algorithm 2 finds the pair of XML documents with maximum structural sharing value and adds them into the initial empty broadcast program $\sigma$ (Line 2). As discussed in Sect. 3, the expected access time is determined by the gaps

between the required documents but not by the sequence of them. Therefore, the sequence of the first pair of XML documents can be simply placed according to the ascendant order of document lengths (Line 3 to 7). Then from Line 9 to Line 21, Algorithm 2 appends the XML document with the maximum structural sharing to the head document $d_{head}$ or the rear document $d_{rear}$ of $\sigma$. If the maximum structural sharing is derived between document $d$ and document $d_{head}$, $d$ will be appended into $\sigma$ from head; otherwise, $d$ will be appended into $\sigma$ from rear. Once the selected document is added to $\sigma$, that document will be removed from $D'$. Then a similar process on $D'$ will be repeated until all XML documents are placed into $\sigma$ in order.

*Time Complexity* Suppose there are totally $n$ documents on the server and each document contains $p$ elements (or in other words, $p$ paths) on average. Then in Algorithm 1, the **for** loop from Line 2 to Line 4 takes $O(np \log p)$ time (it is proposed to use sorted sets here in order to speed up the set operations in the following lines). Note that intersections or differences between two sorted sets take linear time, which meas Line 6 in Algorithm 1 takes $O(p)$ time. Therefore, the **for** loop from Line 5 to Line 8 takes $O(n^2 p)$ time since there are $O(n^2)$ pairs of documents. Finally, in Algorithm 2, Line 2 takes $O(n^2)$ times and the **while** loop from Line 9 to Line 21 takes $O(n^2)$ time as well. Putting all the times together, the total time complexity of both Algorithms 1 and 2 is $O(np \log p + n^2 p)$.

Now consider the scenarios of adding a new document or removing an existing document from the document set on the server. For adding a new document, the update time complexity would be $O(p \log p + np + n^2)$. This is because it takes $O(p \log p)$ time to compute the sorted path set for the new document, $O(np)$ time to intersect with the $n$ existing documents on the server and finally, it takes $O(n^2)$ to recompute the whole data placement. For removing an existing document from the server, the update time complexity would be just $O(n^2)$ as the algorithm only needs to recompute the data placement for the rest $n - 1$ documents.

### 4.3 Index Distribution Strategy

In order to improve energy conservation, smart mobile devices can switch between two operation modes: *active mode* and *doze mode*. In the active mode, a device can listen, compare, and download the required data; while in the doze mode, it turns off antennas and some processes to save energy. The energy consumed in active mode can be up to 100 times of that in doze mode [28]. However, after a broadcast program $\sigma$ is generated, mobile clients cannot locate information of their interests as there is no auxiliary information to assist them, which means mobile clients would need to stay in active mode all the time. Air indexing can help to solve this problem.

Air index is a small amount of auxiliary information of the broadcast program $\sigma$ and is used to assist mobile clients to calculate the arrival time of information that they are interested in [19]. As mentioned, without air index, mobile clients would have to download all XML documents on air to examine which ones satisfy their requests. Therefore, the system needs to apply air indexing technique to the generated broadcast program $\sigma$. With the technique, mobile clients can avoid to examine documents on air one by one. Instead, they can switch to doze mode when uninterested documents are broadcasted and switch to active mode only when interested documents arrive. After the interested documents have been retrieved, they can switch back to doze mode again. In this way, energy can be significantly saved for mobile clients.

This work adopts Compact Index (CI) [8] as the index structure and (1, $m$) index scheme [19] as the index distribution strategy.

CI provides a two-tier air index scheme [8]. The basic structure of CI is the combination of all the DataGuides (containing structural information only and supporting simple XPath queries, while branching queries are not supported) of XML documents in $\sigma$, which utilizes RoXSum [29] technique to integrate these DataGuides, where common structural information amongst different XML documents can be combined to form much smaller structural representations about the original XML document set. In this basic index structure, every unique-label path (with only structural information) of a document appears exactly once for supporting simple XPath queries. Thus it contains the entire unique-label paths in all of the XML documents. CI also includes a two-tier structure which enables efficient access protocol at the client which facilitates the index access. Normally, CI is only 1.5% of the original XML data in terms of size. More details can be found in [8].

The idea of (1, $m$) index scheme [19] is shown in Figure 4. From the figure, it can be seen that the generated broadcast program $\sigma$ is divided into $m$ data segments and before broadcasting each data segment, an index segment will be broadcasted first. The optimal value $m$ is given by $\sqrt{\frac{\mathcal{L}_{data}}{\mathcal{L}_{index}}}$. Here, $\mathcal{L}_{data}$ refers to the total length of the raw broadcast program $\sigma$ without any index information and $\mathcal{L}_{index}$ refers to the length of index, which is CI in this work. (1, $m$) index scheme can best balance access time and tuning time of mobile clients. More details can be found in [19].

## 5 Experiments

This section reports the performance of the proposed data placement algorithm. The algorithm efficiency is shown in terms of access time, which is a common measure of performance in data broadcasts. Since this is the first work that determines broadcast schedules based only on XML data on the server, the proposed algorithm is compared with a common random data placement algorithm (RDPA).

### 5.1 Experimental Setup

The experiments are run on three data sets (*DS*1, *DS*2, *DS*3) each with 250 XML documents and two more data sets (*DS*4, *DS*5) with 500 and 1000 XML documents, respectively. All generated documents are defined by News Industry Text Format (NITF) DTD

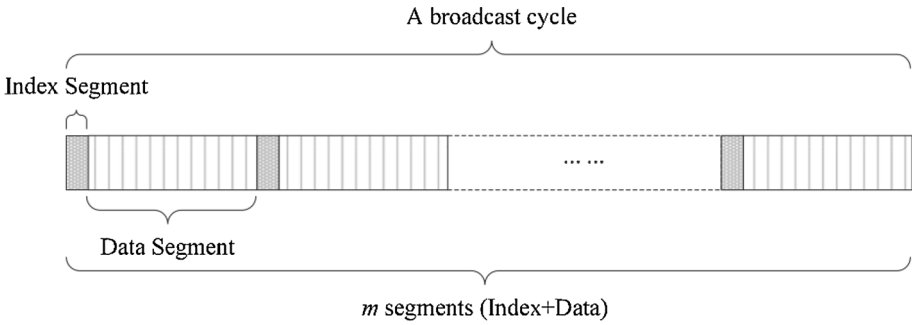



**Fig. 4** (1, $m$) index distribution

[30]. This DTD is published for news copy production, press releases, and Web-based news organizations. The average depth of all five document sets is between 6 and 8 while the maximum depth is 22.

The details of these data sets are shown in Table 3. Data in *DS*1 can be well clustered into 6 clusters. Moreover, for any two documents $d_i, d_j$ in two different clusters of *DS*1, the minimum similarity values, the maximum similarity values and the average similarity values of all four metrics (normalized Cohesion is adopted here) are shown in Table 4. It can be seen that all clusters are quite different from each other and share very little structural information. Data in *DS*2 are miscellaneous. Documents in *DS*2 cannot be classified into fine clusters. Data in *DS*3 are a mix of well-clustered data and miscellaneous data, which include 125 XML documents from *DS*1 and 125 XML documents from *DS*2. Similar to *DS*2, data in *DS*4, *DS*5 are miscellaneous as well.

In the experiments, XPath queries are generated using the generator developed by [31]. Queries are allowed to repeat. The generator provides several parameters to generate different types of XPath queries, such as query depth, probability of * and // and so on. The probability of * and // appearing in each query's step is between 5% and 30% (denoted *PROB*, and the default value is 10%). Here, * operator will match any element node in the document while // operator will select nodes in the document from the current node that match the selection no matter where they are. Query Incoming Rate (denoted *QIR*) means the number of newly issued queries from mobile clients in a unit of time. This unit of time is measured by the time that mobile wireless system takes to broadcast a block of 1024-byte XML data. The maximum depth of generated XPath queries (denoted *MQD*) is between 5 and 8. Table 5 shows the value range of parameters in the experiments. It should be noted that the user queries are assumed to follow a uniform distribution.

The random data placement algorithm (denoted RDPA) is compared with GDPA [implemented using all four similarity metrics defined in Equations (1), (2), (3) and (12)]. In RDPA, the server broadcasts XML documents in a random order. This random order is implemented by a Java class `Random`. When applying a series of pseudorandom numbers on the order of XML documents in a broadcast program, there would be conflicts. For example, suppose there are totally $N$ documents to be broadcasted. It is needed to generate pseudorandom numbers between 1 and $N$. After $k$ out of $N$ documents having been randomly placed, the next chosen document may be one of the first $k$ documents. If such case happens, it is simply ignored and Random class is used to generate pseudorandom numbers between 1 and $N-k$ for the rest $N-k$ documents. In this way, a random order of $N$ documents can be simulated.

Both RDPA and GDPA are implemented on Java Platform Standard Edition 6 running on Windows 7 Enterprise, 64-bit Operating System. All the experimental results are obtained by running 30 consecutive broadcast cycles. When varying *PROB*, *QIR* and *MQD*

**Table 3** Data sets in the experiments

| Set name | Length | | | Remark |
|---|---|---|---|---|
| | Minimum | Maximum | Average | |
| *DS*1 | 2.4KB | 8.1KB | 5.0KB | 6 clusters |
| *DS*2 | 0.5KB | 45.9KB | 12.4KB | Miscellaneous |
| *DS*3 | 2.4KB | 24.8KB | 9.9KB | Hybrid |
| *DS*4 | 0.5KB | 55.8KB | 12.3KB | Miscellaneous |
| *DS*5 | 0.3KB | 65.6KB | 12.7KB | Miscellaneous |

**Table 4** Similarity between clusters in *DS*1

| Metric | Similarity | | |
|---|---|---|---|
| | Minimum | Maximum | Average |
| *Jaccard* | 0.0097 | 0.1102 | 0.0435 |
| *Dice* | 0.0049 | 0.0583 | 0.0225 |
| *Lian* | 0.0057 | 0.1039 | 0.0345 |
| *Cohesion* | 0.0229 | 0.4620 | 0.1457 |

**Table 5** Workload parameters for the experiments

| Parameter | Range | Default | Description |
|---|---|---|---|
| *PROB* | 5–30% | 10% | Probability(* and //) |
| *QIR* | 0.1–5 | 1 | Query incoming rate |
| *MQD* | 5–8 | 7 | Maximum query depth |

are set to their default values. When varying *QIR*, *PROB* and *MQD* are set to their default values. Similarly, when varying *MQD*, *PROB* and *QIR* are set to their default values.

Regarding air indexing and index distribution strategy, as mentioned in Sect. 4, in the experiments, Compact Index (CI) [8] is adopted as the index structure and (1, *m*) index scheme [19] is adopted as the index distribution strategy. This is because CI is the state-of-the-art indexing technique for XML data broadcast and (1, *m*) index scheme is the most popular index distribution strategy for traditional periodic data broadcast. More details can be found in [8, 19].

## 5.2 Performance of GDPA

The experimental results are shown in Figs. 5, 6, 7, 8 and 9. Average access time (*AAT*) is the performance metric. Also only *AAT* is considered for all successful matched queries and abandon unsuccessful matched queries. The main reason for this is that, *AAT* of unsuccessful queries is determined by index distribution but not by data placement results (more details about this can be found in [19]). Note that, GDPA can be implemented with four different similarity metrics defined in Sect. 4, which are Jaccard metric, Dice's coefficient, Lian's metric and the proposed Cohesion. Through the experiments, Jaccard metric and Dice's coefficient always yield the same results. Therefore, GDPA implemented with them is denoted as *J/D* method in all figures. Meanwhile, GDPA implemented with Lian's metric is denoted as *Lian* method and GDPA implemented with Cohesion is denoted as *Cohesion* method.

Figure 5 shows the results on *DS*1. From the figure it can be seen that all GDPA methods outperform RDPA significantly. Specifically, J/D method achieves the best results while Lian method and Cohesion method provides similar results. This indicates that J/D method better fits well-clustered data. Also, the reason for Lian method and Cohesion method showing similar results is that both methods emphasize on the common structure of both comparing XML documents against the larger document [see also Eqs. (3) and (11)]. Further, since *DS*1 is well-clustered, from the definitions of Lian and Cohesion metrics, it can also be inferred that the partial order of similarity results using these two methods are similar to each other. Note that, according to the data placement algorithm described in

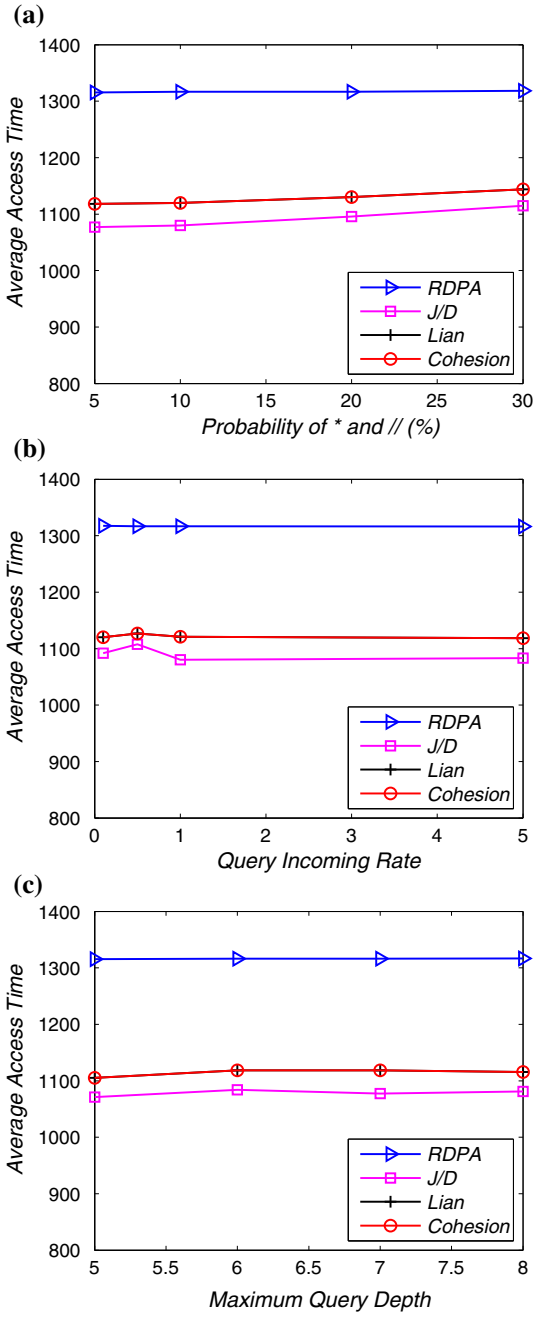



**Fig. 5** Evaluating *AAT* Performance on *DS*1: well-clustered data set with 250 documents. **a** Varying *PROB*, **b** varying *QIR*, **c** varying *MQD*

Sect. 4, the data placement results are determined by the partial order of the similarity results. Hence, for *DS*1, Lian method and Cohesion method yield similar results.

In Fig. 5a, GDPA methods become slightly worse when *PROB* increases. Since *DS*1 is well-clustered, most queries only require documents in the same clusters. Thus *PROB* has

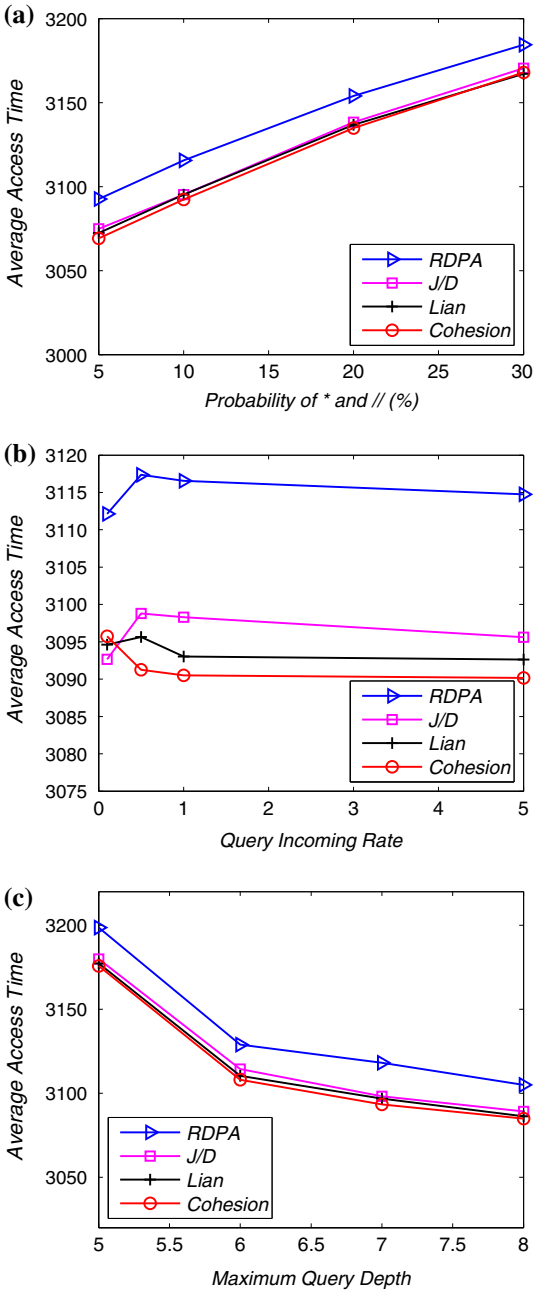



**Fig. 6** Evaluating *AAT* Performance on *DS*2: miscellaneous data set with 250 documents. **a** Varying *PROB*, **b** varying *QIR*, **c** varying *MQD*

less effect on *AAT*. In Fig. 5b, when *QIR* increases, J/D method becomes slightly better. This indicates that J/D method can achieve better scalability than other methods when accessing well-clustered data. Figure 5c shows that all GDPA methods remain stable as *MQD* increases. It is interesting to note that for RDPA, *AAT* always remains stable.

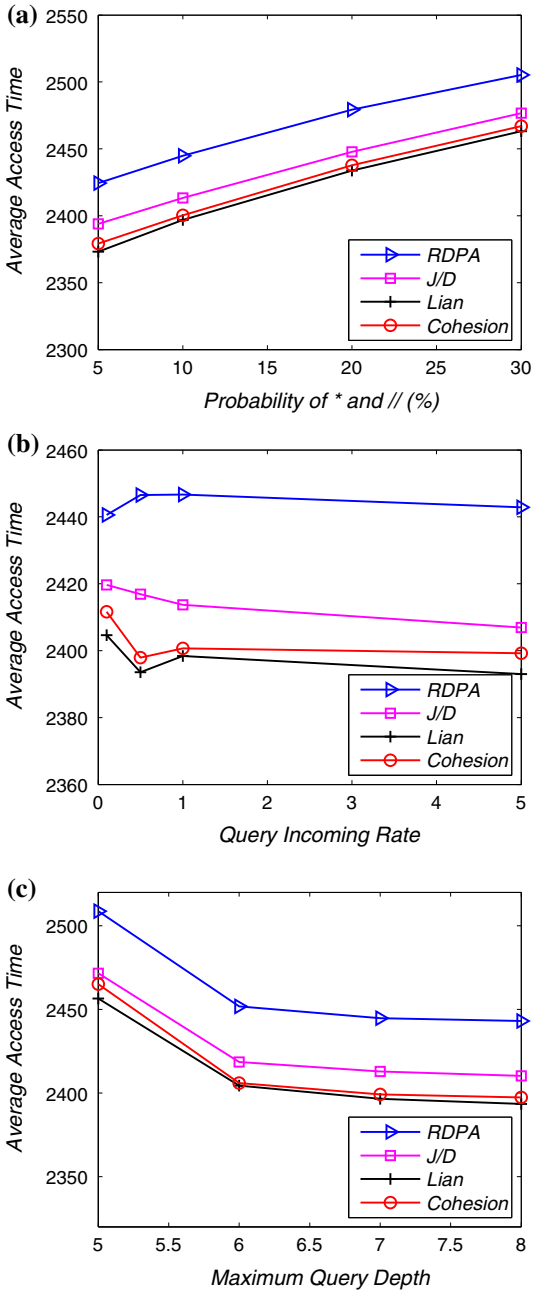



**Fig. 7** Evaluating *AAT* Performance on *DS*3: a mixed set of well-clustered data and miscellaneous data with 250 documents. **a** Varying *PROB*, **b** varying *QIR*, **c** varying *MQD*

Figure 6 shows the results on *DS*2. From the figure it can be seen that all GDPA methods achieve better performance when compared with RDPA. Specifically, Cohesion method achieves the best results while J/D method achieves the worst results among GDPA methods. This indicates that Cohesion method better fits miscellaneous data. In Fig. 6a, both GDPA methods and RDPA become worse when *PROB* increases. It is clear that

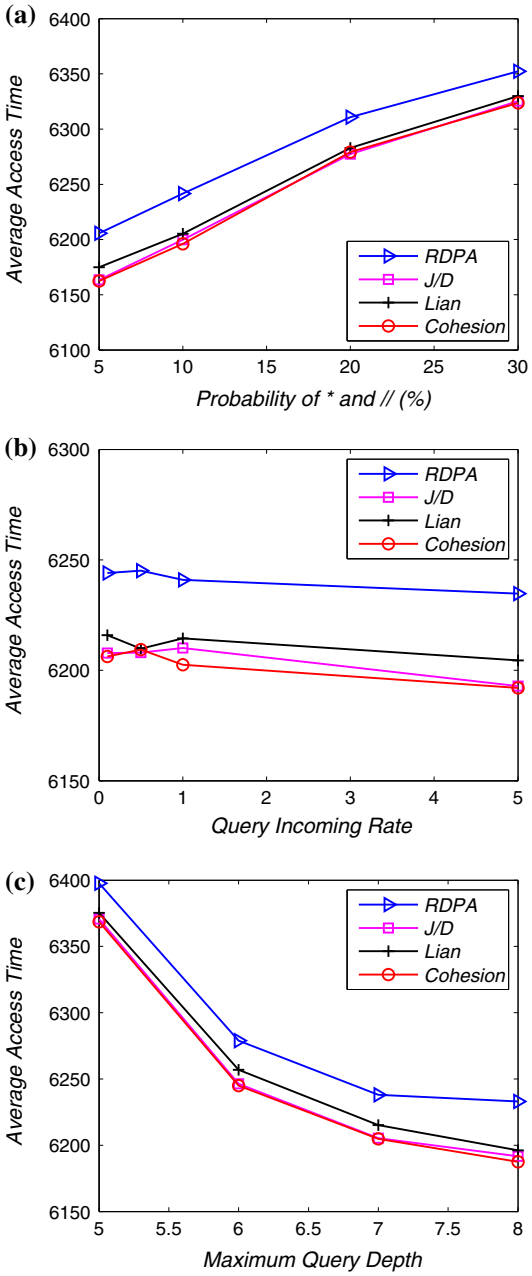



**Fig. 8** Evaluating *AAT* performance on *DS*4: miscellaneous data set with 500 documents. **a** Varying *PROB*, **b** varying *QIR*, **c** varying *MQD*

*PROB* has more effect on *AAT* for miscellaneous data. In Fig. 6b, when *QIR* increases from 0.1 to 0.5, GDPA methods J/D and Lian together with RDPA become worse while Cohesion method still becomes better. After that, when *QIR* increases, all methods become slightly better. This shows that Cohesion method can achieve best scalability when accessing miscellaneous data. Note that, we have seen some intersecting lines when the

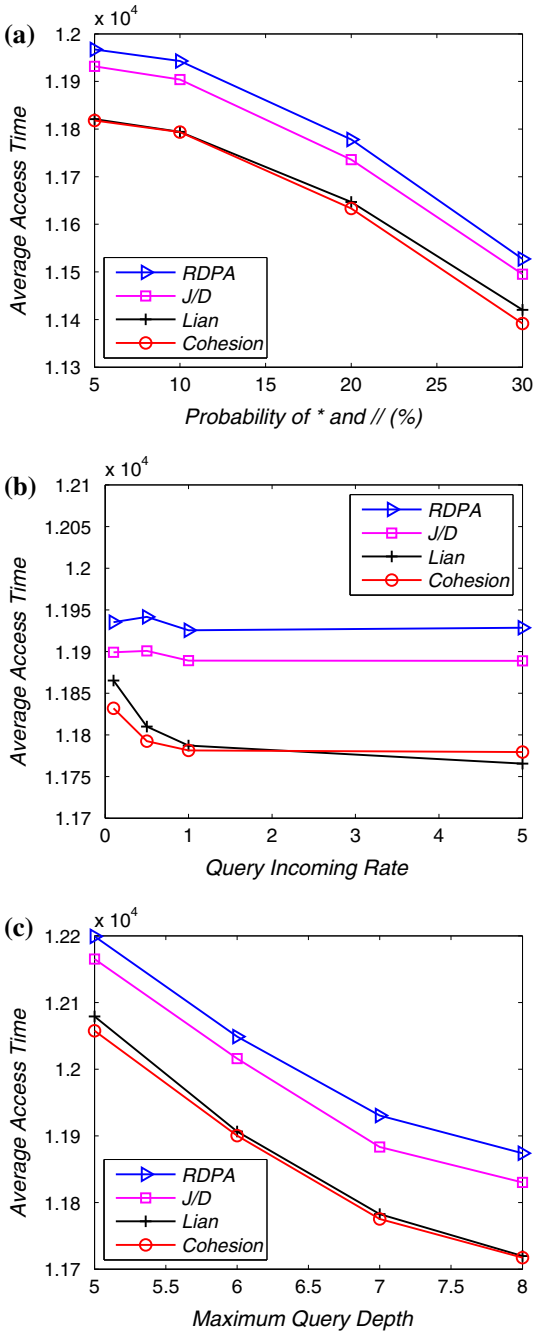



**Fig. 9** Evaluating *AAT* Performance on *DS*5: miscellaneous data set with 1000 documents. **a** Varying *PROB*, **b** varying *QIR*, **c** varying *MQD*

*QIR* is very small. The main reason for this is that when *QIR* is very small, the average (or the expected) access time is unstable as the total number of queries is too small. Fig. 6c shows that all methods achieve better *AAT* as *MQD* increases since selectivity of queries drops with the increase of *MQD*.

Figure 7 shows the results on *DS*3. Similarly, all GDPA methods achieve better performance when compared with RDPA. Specifically, Lian method achieves the best results while J/D method provides the worst results among GDPA methods. This shows that Lian method better fits hybrid data. However, Cohesion method achieves very similar performance of Lian method. In Fig. 7a, both GDPA methods and RDPA become worse when *PROB* increases. *PROB* has more effect on *AAT* for hybrid data. In Fig. 7b, when *QIR* increases, all GDPA methods become slightly better and still Lian method provides the best results. Figure 7c shows that all methods achieve better *AAT* as *MQD* increases since selectivity of queries drops with the increase of *MQD*.

Similar experiments were also conducted using larger data sets (including 500 and 1000 XML documents respectively). The results of experiments are shown in Figs. 8 and 9. From these figures, similar trends can be observed in Fig. 6.

Therefore, from the above experiments, it can be seen that GDPA methods always achieve better *AAT* when compared with RDPA. When accessing well-clustered data, J/D method achieves the best performance. When accessing miscellaneous data, Cohesion method provides better performance in most cases. This is because the J/D metric emphasizes on the common paths between two XML documents according to their definitions (the Jaccard and Dice similarity definitions). This also means, the J/D metric is more suitable for well-clustered data sets. In contrast, the Cohesion metric emphasizes on both the common paths (representing the probability that two documents are matched against a given query at the same time) and the difference in the path sets of two documents (representing the probability that two documents are not matched against the same query at the same time). Therefore, the Cohesion metric is more suitable for miscellaneous data sets. Finally when accessing hybrid data, Lian method shows better performance in most cases since Lian method emphasizes on the difference between XML documents but not on common paths.

Query selectivity and document coverage rate were also investigated in the experiments. Here, query selectivity refers to the average proportion of documents matched with a user query and document coverage rate refers to the proportion of documents in the entire document set on the server required by at least one user query. The results are obtained using all workload parameters at default values. As can be seen from Table 6, the query selectivity is ranging from around 32 to 45%. The main reason for this is that the probability of * and // is 10% by default. Further, for all data sets. the document coverage rate is 100%, which is mainly due to the same reason of query selectivity. This shows that all documents on the server are covered in all the experiments.

Finally, the maintenance cost of the data placement algorithms is studied. The maintenance cost is measured when adding a new document to the server or removing an existing document from the server. The maintenance results are shown in Table 7. From the table it can be seen that, when adding a new document in a document set with 250 documents on the server, the time cost to calculate the similarity between the new document and all the existing documents (shown as "Similarity Time" in the table) is ranging from 209 to 238 ms on average for the Dice, Jaccard and Lian metrics. For the Cohesion metric, it takes a bit longer which is 378 ms on average due to fact that the Cohesion metric requires more set operations that the other three metrics. Meanwhile, the time cost to readjust the data placement (shown as "Placement Time" in the table) is quite similar among all four metrics, which is ranging from 168 to 181 ms. A similar pattern is observed when adding a new document to a larger document set.

On the other hand, when removing an existing document from the server, only Placement Time will be incurred. From the table it can be seen that, similarly, the Placement

**Table 6** Query selectivity and document coverage rate

| Data set | Query selectivity (%) | Coverage rate (%) |
|---|---|---|
| *DS*1 | 44.8 | 100 |
| *DS*2 | 33.6 | 100 |
| *DS*3 | 35.6 | 100 |
| *DS*4 | 32.6 | 100 |
| *DS*5 | 33.2 | 100 |

Time costs for four metrics are quite similar to each other. For example, when removing an existing document from a document set with 250 documents, the Placement Time is ranging from 165 to 185 ms. Again, a similar pattern is observed when removing an existing document from a larger document set.

## 6 Related Work

Many studies have been done to investigate data placement techniques to reduce access time [32–34]. These studies generally assume that each user query requires one data item only. Other studies handle data placement problems for queries that may require multiple data items.

Multi-item data placement problem is related to the data placement problem of XML data which is the focus of this work. It is proved to be a NP-Complete problem [27]. A data placement method for multi-item queries called QEM is introduced in [35], which opened up a new perspective in this field. In addition, several improved methods are proposed [17, 36]. The above work is all within the scope of periodic broadcast and generally assumes that the clients' queries are already known in advance. However, in some applications, the user demands may be either unknown or costly to collect the related information due to the mobility of mobile users and privacy concerns.

Multi-item data placement problem in the on-demand broadcast mode has also attracted lots of interests [13, 37]. These approaches are in pure on-demand broadcast mode and

**Table 7** Data placement update time (in ms)

| No. of documents | Metric | Add | | Remove |
|---|---|---|---|---|
| | | Similarity time | Placement time | Placement time |
| 250 | Dice | 228 | 168 | 165 |
| | Jaccard | 209 | 169 | 174 |
| | Lian | 238 | 174 | 171 |
| | Cohesion | 378 | 181 | 185 |
| 500 | Dice | 343 | 745 | 751 |
| | Jaccard | 359 | 814 | 811 |
| | Lian | 419 | 844 | 837 |
| | Cohesion | 763 | 811 | 801 |
| 1000 | Dice | 781 | 1212 | 1233 |
| | Jaccard | 640 | 1152 | 1187 |
| | Lian | 691 | 1192 | 1202 |
| | Cohesion | 1130 | 1298 | 1275 |

strictly require that mobile clients submit their queries to the server for desired data. Otherwise, the server will not broadcast related data on air. This is because the server filters and schedules data solely based on submitted queries. However, frequent use of uplink channel leads to high communication cost via uplink channel, which can shorten battery life of mobile clients dramatically.

The above mentioned studies focus on flat data broadcasts, in which indices of data items are generally key-based and data do not contain structural information. Recently, besides the traditional flat data broadcast, a wealth of work dealing with XML data broadcast has appeared. Some work addresses the performance optimization of query processing of XML streams in wireless broadcast [6, 38], while other work designs indexing techniques for XML data broadcast based on existing XML indexing techniques [7, 8]. However, their work mainly focuses on air indexing techniques similar to content based indexing techniques in XML stream processing or XPath query based indexing techniques. Moreover, this kind of work does not study the data placement problem for XML data broadcast.

Data placement problem for XML data broadcast is investigated in [39]. In that work, the broadcast schedules are generated based on clustering results of XML data on the server. However, the clustering process requires manually specifying the number of clusters and has to compare different clustering results based on clients' query distribution in order to find the optimal clustering result, which differs from this work.

## 7 Conclusion

In this paper, the data placement problem of periodic XML data broadcast has been studied. Taking advantage of the structured characteristics of XML data, it becomes feasible to generate effective broadcast programs based only on XML data on the server. This not only makes the proposed approach distinguished from previous studies, but also enables it to have broader applicability. A theoretical analysis of the problem is presented and structural sharing in XML data is also discussed, which forms the basis of the novel greedy data placement algorithm (GDPA). The experiments demonstrated that the proposed algorithm could improve access efficiency and achieve better scalability.

In the near future, some on-going work includes further improving system's performance by investigating the insights of structural sharing among XML documents. For example, details on how to measure structural sharing distribution in an XML document set can be explored to show how the distribution affects the expected access time of queries and how to choose a similarity metric based on structural sharing distribution in a set of XML documents, etc.



## Appendix

### Proof of Proposition 1

*Proof* According to the two constraints of function $f(\mathbf{X})$, it is true that

$$f(\mathbf{X}) = (x_1 + x_2 + \cdots + x_n)^2 - 2 \cdot \sum_{i \neq j} x_i \cdot x_j \leq M^2$$

When $x_1 = x_2 = \cdots = x_{k-1} = 0$ and $x_k = M$ (or any other kind of combinations with $k-1$ zeros and one $M$, which means only 1 positive variable exists[4].), $f(\mathbf{X})$ reaches its upper bound $\overline{f(\mathbf{X})} = M^2$. In all other cases, if two or more variables are positive, i.e. $x_1 > 0$ and $x_2 > 0$, then $2 \cdot \sum_{i \neq j} x_i \cdot x_j \geq 2 \cdot (x_1 \cdot x_2) > 0$ which indicates $f(\mathbf{X}) < M^2$. □

**Proof of Proposition 2**

*Proof* Mathematical induction can be used to prove the proposition (for all $k \geq 1$).

For the base step, when $k = 1, f(\mathbf{X}) = x_1^2$, it is trivial to prove that $f(\mathbf{X}) = M^2 \geq 1 \cdot (\frac{M}{1})^2$ and $\underline{f(\mathbf{X})} = M^2$ since $x_1 = M = \frac{M}{1}$.

For the inductive step, it is assumed that when $k = n$, $f(\mathbf{X}) \geq \frac{M^2}{n}$ and when $x_1 = x_2 = \cdots = x_n = \frac{M}{n}, f(\mathbf{X})$ reaches its lower bound $\underline{f(\mathbf{X})} = \frac{M^2}{n}$. Then for $k = n+1$, it is true that

$$\begin{aligned} f(\mathbf{X}) &= \sum_{i=1}^{n} x_i^2 + x_{n+1}^2 \\ &\geq n \cdot \left(\frac{M - x_{n+1}}{n}\right)^2 + x_{n+1}^2 \\ &= \frac{(n+1) \cdot x_{n+1}^2 - 2 \cdot M \cdot x_{n+1} + M^2}{n} \\ &= \frac{(n+1) \cdot (x_{n+1} - \frac{M}{n+1})^2 + \frac{n \cdot M^2}{n+1}}{n} \\ &\geq \frac{M^2}{n+1} \end{aligned}$$

From the above induction, it can be seen that when $x_{n+1} = \frac{M}{n+1}$ and $x_1 = x_2 = \cdots = x_n = \frac{M - x_{n+1}}{n} = \frac{M}{n+1}, f(\mathbf{X})$ reaches its lower bound $\underline{f(\mathbf{X})} = \frac{M^2}{n+1}$.

Because both the base step and the inductive step have been shown, by the principle of mathematical induction the proposition is true. □

**Proof of Proposition 3**

*Proof* According to the definitions of $f(\mathbf{X})_{x_{m+1}}$, it is true that

$$f(\mathbf{X})_{x_{m+1}} = \sum_{i=1}^{m} x_i^2 + x_{m+1}^2 + \sum_{i=m+2}^{k} x_i^2$$

Since $M' = M - \sum_{i=1}^{m} x_i = \sum_{i=m+1}^{k} x_i$, according to Proposition 1, it is true that

[4] Note that it is not possible to have all variables to be 0 since $M$ is positive.

$$\begin{aligned}\overline{f(\mathbf{X})}_{x_{m+1}} &= \sum_{i=1}^{m} x_i^2 + x_{m+1}^2 + (M' - x_{m+1})^2 \\ &= \sum_{i=1}^{m} x_i^2 + 2 \cdot x_{m+1}^2 - 2 \cdot M' \cdot x_{m+1} + M'^2 \\ &= \sum_{i=1}^{m} x_i^2 + 2 \cdot \left( x_{m+1} - \frac{M'}{2} \right)^2 + \frac{M'^2}{2}\end{aligned}$$

According to the above result, for any $x_{m+1} < x'_{m+1}$ it can be inferred that

1. $\overline{f(\mathbf{X})}_{x_{m+1}} > \overline{f(\mathbf{X})}_{x'_{m+1}}$, if $x_{m+1} < x'_{m+1} \leq \frac{M'}{2}$
2. $\overline{f(\mathbf{X})}_{x_{m+1}} < \overline{f(\mathbf{X})}_{x'_{m+1}}$, if $\frac{M'}{2} \leq x_{m+1} < x'_{m+1}$
3. Indefinite for all other cases □

**Proof of Proposition 4**

*Proof* According to the definitions of $f(\mathbf{X})_{x_{m+1}}$, it is true that

$$f(\mathbf{X})_{x_{m+1}} = \sum_{i=1}^{m} x_i^2 + x_{m+1}^2 + \sum_{i=m+2}^{k} x_i^2$$

Since $M' = M - \sum_{i=1}^{m} x_i = \sum_{i=m+1}^{k} x_i$, according to Proposition 2, it is true that

$$\begin{aligned}\underline{f(\mathbf{X})}_{x_{m+1}} &= \sum_{i=1}^{m} x_i^2 + x_{m+1}^2 \\ &\quad + (k - m - 1) \cdot \left( \frac{M' - x_{m+1}}{k - m - 1} \right)^2 \\ &= \sum_{i=1}^{m} x_i^2 \\ &\quad + \frac{(k-m)x_{m+1}^2 - 2 \cdot M' \cdot x_{m+1} + M'^2}{k - m - 1} \\ &= \sum_{i=1}^{m} x_i^2 \\ &\quad + \frac{(k-m)\left(x_{m+1} - \frac{M'}{k-m}\right)^2 + \frac{(k-m-1)M'^2}{k-m}}{k - m - 1}\end{aligned}$$

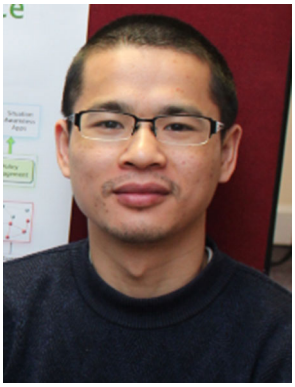
According to the above result, for any $x_{m+1} < x'_{m+1}$ it can be inferred that

1. $\underline{f(\mathbf{X})}_{x_{m+1}} > \underline{f(\mathbf{X})}_{x'_{m+1}}$, if $x_{m+1} < x'_{m+1} \leq \frac{M'}{k-m}$
2. $\underline{f(\mathbf{X})}_{x_{m+1}} < \underline{f(\mathbf{X})}_{x'_{m+1}}$, if $\frac{M'}{k-m} \leq x_{m+1} < x'_{m+1}$
3. Indefinite for all other cases □

## References

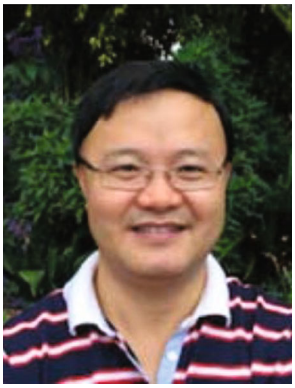

1. Shaddad, R. Q., Mohammad, A. B., Al-Gailani, S. A., Al-hetar, A. M., & Elmagzoub, M. A. (2014). A survey on access technologies for broadband optical and wireless networks. *Journal of Network and Computer Applications*, *41*, 459–472.
2. The International Telecommunication Union (ITU). (2011) Measuring the information society 2012. http://www.itu.int/dms_pub/itu-d/opb/ind/D-IND-ICTOI-2012-SUM-PDF-E.pdf
3. Jianliang, X., Lee, D.-L., Qinglong, H., & Lee, W.-C. (2002). *Handbook of wireless networks and mobile computing* (pp. 243–265). Hoboken: Wiley.
4. SiriusXM. (2017). Accessed on April 3, 2017. http://www.siriusxm.com
5. Park, C.-S., Kim, C. S., & Chung, Y. D. (2005). Efficient stream organization for wireless broadcasting of XML data. In *ASIAN* (pp. 223–235).
6. Chung, Y. D., & Lee, J. Y. (2007). An indexing method for wireless broadcast XML data. *Information Sciences*, *177*(9), 1931–1953.
7. Qin, Y., Sun, W., Zhang, Z., Yu, P., He, Z., & Chen, W. (2009). A novel air index scheme for twig queries in on-demand XML data broadcast. In *International conference on database and expert systems applications (DEXA)* (pp. 412–426).
8. Sun, W., Yu, P., Qin, Y., Zhang, Z., & Zheng, B. (2009). Two-tier air indexing for on-demand XML data broadcast. In *IEEE international conference on distributed computing systems (ICDCS)* (pp. 199–206).
9. Park, J. P., Park, C.-S., & Chung, Y. D. (2010). Energy and latency efficient access of wireless XML stream. *Journal of Database Management*, *21*(1), 58–79.
10. Wu, J., Liu, P., Gan, L., Qin, Y., & Sun, W. (2011). Energy-conserving fragment methods for skewed XML data access in push-based broadcast. In *International conference on Web-Age Information Management (WAIM)* (pp. 590–601).
11. Sun, W., Liu, P., Wu, J., Qin, Y., & Zheng, B. (2012). An automaton-based index scheme for on-demand XML data broadcast. In *International conference on database systems for advanced applications (DASFAA) (2)* (pp. 96–110).
12. Park, C.-S., Park, J. P., & Chung, Y. D. (2012). PrefixSummary: A directory structure for selective probing on wireless stream of heterogeneous XML data. *IEICE Transactions on Information and Systems*, *95–D*(5), 1427–1435.
13. Sun, W., Zhang, Z., Yu, P., & Qin, Y. (2008). Efficient data scheduling for multi-item queries in on-demand broadcast. In *The IEEE/IFIP international conference on embedded and ubiquitous computing (EUC) (1)* (pp. 499–505).
14. Chen, J., Lee, V. C. S., & Liu, K. (2010). On the performance of real-time multi-item request scheduling in data broadcast environments. *Journal of Systems and Software*, *83*(8), 1337–1345.
15. Acharya, S., Alonso, R., Franklin, M. J., & Zdonik, S. B. (1995). Broadcast disks: Data management for asymmetric communications environments. In *The ACM international conference on management of data (SIGMOD)* (pp. 199–210).
16. Acharya, S., Franklin, M. J., & Zdonik, S. B. (1997). Balancing push and pull for data broadcast. In *The ACM international conference on management of data (SIGMOD)* (pp. 183–194).
17. Chang, Y.-I., & Hsieh, W.-H. (2004). An efficient scheduling method for query-set-based broadcasting in mobile environments. In *IEEE international conference on distributed computing systems (ICDCS): Workshops* (pp. 478–483).
18. Qin, Y., Sheng, Q. Z., Mehdi, M., Wang, H., & Xie, D. (2013). Effectively delivering XML information in periodic broadcast environments. In *International conference on database and expert systems applications (DEXA)* (pp. 165–179).
19. Imielinski, T., Viswanathan, S., & Badrinath, B. R. (1997). Data on air: Organization and access. *IEEE Transactions on Knowledge and Data Engineering*, *9*(3), 353–372.
20. Rafiei, D., Moise, D. L., & Sun, D. (2006). Finding syntactic similarities between XML documents. In *International conference on database and expert systems applications (DEXA): Workshops* (pp. 512–516).
21. Helmer, S. (2007). Measuring the structural similarity of semistructured documents using entropy. In *International conference on very large data bases (VLDB)* (pp. 1022–1032).
22. Lin, D. (1998). An information-theoretic definition of similarity. In *International conference on machine learning (ICML)* (pp. 296–304).
23. Ganesan, P., Garcia-Molina, H., & Widom, J. (2003). Exploiting hierarchical domain structure to compute similarity. *ACM Transactions on Information Systems*, *21*(1), 64–93.
24. Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology*, *26*(3), 297–302.

25. Lian, W., Cheung, D. W.-L., Mamoulis, N., & Yiu, S.-M. (2004). An efficient and scalable algorithm for clustering XML documents by structure. *IEEE Transactions on Knowledge and Data Engineering*, *16*(1), 82–96.
26. Imielinski, T., Viswanathan, S., & Badrinath, B. R. (1994). Power efficient filtering of data an air. In *International conference on extending database technology (EDBT)* (pp. 245–258).
27. Chung, Y. D., & Kim, M.-H. (2001). Effective data placement for wireless broadcast. *Distributed and Parallel Databases*, *9*(2), 133–150.
28. Viredaz, M. A., Brakmo, L. S., & Hamburgen, W. R. (2003). Energy management on handheld devices. *ACM Queue*, *1*(7), 44–52.
29. Vagena, Z., Moro, M. M., Tsotras, V. J. (2007). RoXSum: Leveraging data aggregation and batch processing for XML routing. In *IEEE international conference on data engineering (ICDE)* (pp. 1466–1470).
30. IPTC. (2000). International Press Telecommunications Council, News Industry Text Format (NITF), version 2.5. http://www.nitf.org, visited on 20/09/2012.
31. Diao, Y., Altinel, M., Franklin, M. J., Zhang, H., & Fischer, P. M. (2003). Path sharing and predicate evaluation for high-performance XML filtering. *ACM Transactions on Database Systems*, *28*(4), 467–516.
32. Sun, W., Shi, W., Shi, B., & Yijun, Y. (2003). A cost-efficient scheduling algorithm of on-demand broadcasts. *Wireless Networks*, *9*(3), 239–247.
33. Sun, B., Hurson, A. R., & Hannan, J. (2004). Energy-efficient scheduling algorithms of object retrieval on indexed parallel broadcast channels. In *International conference on parallel processing (ICPP)* (pp. 440–447).
34. Kang, S. H. (2011). Wireless data broadcast scheduling with utility metric based on soft deadline. *IEICE Transactions*, *94–B*(5), 1424–1431.
35. Chung, Y. D., & Kim, M.-H. (1999). QEM: A scheduling method for wireless broadcast data. In *International conference on database systems for advanced applications (DASFAA)* (pp. 135–142).
36. Lee, G., Yeh, M.-S., Lo, S.-C., & Chen, A. L. (2002). A strategy for efficient access of multiple data items in mobile environments. In *IEEE international conference on mobile data management (MDM)* (pp. 71–78).
37. Qin, Y., Wang, H., & Xiao, J. (2011). Effective scheduling algorithm for on-demand XML data broadcasts in wireless environments. In *The Australasian Database Conference (ADC)* (pp. 95–102).
38. Park, S.-H., Choi, J.-H., & Lee, S.K. (2006). An effective, efficient XML data broadcasting method in a mobile wireless network. In *International conference on database and expert systems applications (DEXA)* (pp. 358–367).
39. Qin, Y., Wang, H., & Sun, L. (2011). Cluster-based scheduling algorithm for periodic XML data broadcast in wireless environments. In *International conference on advanced information networking and applications (AINA) workshops* (pp. 855–860).



**Yongrui Qin** is currently a Lecturer in School of Computing and Engineering, University of Huddersfield, United Kingdom. His main research interests include Internet of Things, Graph Data Management, Data Stream Processing, Data Mining, Information Retrieval, Semantic Web, Computer Networks, and Mobile Computing. Dr. Qin has published more than 40 refereed technical papers, including publications in prestigious journals, such as ACM Computing Surveys, IEEE Trans. on Parallel Distributed Systems, World Wide Web Journal, Journal of Network and Computer Applications, and IEEE Internet Computing, as well as top international conferences, such as SIGIR, EDBT, CIKM, WISE, DASFAA, and SSDBM.

**Quan Z. Sheng** is a full professor and Head of Department of Computing, Macquarie University. Prof. Sheng's research interests include Internet of Things, Big Data Analytics, Smart Cities, Service-Oriented Computing, and Pervasive Computing. Prof. Michael Sheng has published more than 300 publications in premier journals and conferences such as ACM Computing Surveys, VLDB Journal, TSC, ACM TOIT, TIST, TKDD, IEEE TPDS, TKDE, Communication of the ACM, IEEE Computer, IEEE Internet Computing, VLDB, WWW, ICDE, CIKM, ICSE, ICWS, and ICSOC. Prof. Sheng is the recipient of several major awards including ARC Future Fellowship in 2014, the Chris Wallace Award for Outstanding Research Contribution in 2012, and Microsoft Research Fellowship in 2003.



**Hua Wang** received the Ph.D. degree in computer science from the University of Southern Queensland (USQ) in 2004. He was a Professor with USQ from 2011 to 2013. He is currently a Full-Time Professor with the Centre for Applied Informatics, Victoria University. He has authored or co-authored over 200 peer-reviewed research papers mainly in data security, data mining, access control, privacy and Web services, as well as their applications in the fields of e-health and e-environment.

**Nickolas J. G. Falkner** received the Ph.D. degree in discovery and classification of information in large systems from the University of Adelaide, where he is an Associate Professor with the School of Computer Science. His research interests include automated network configuration, applications of cryptography, and data stream management. He is also active in educational research, with a focus on increasing student participation, retention, and enthusiasm.