AN INFORMATION FILTERING SYSTEM THAT OPTIMIZES THE PROCESSING METHOD BASED ON MATHEMATICAL PROPERTIES

Takuya Kodera† Rie Sawai‡ Tsutomu Terada* Masahiko Tsukamoto† Shojiro Nishio†

†Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka 565-0871, Japan
E-mail:{kodera, tuka, nishio}@ist.osaka-u.ac.jp
‡NHK (Japan Broadcasting Corporation)
2-2-1 Jinnan, Shibuya-ku, Tokyo 150-8001, Japan
E-mail:sawai.r-ka@nhk.or.jp
*Cybermedia center, Osaka University
5-1 Mihogaoka, Ibaraki, Osaka 567-0047, Japan
E-mail:tsutomu@cmc.osaka-u.ac.jp

ABSTRACT

In recent years, due to the increasing popularization of various data broadcast services, the amount and the variety of broadcast data have been increasing. As a result, there is a strong demand for filtering techniques that automatically extract only the necessary data. The optimum filtering processing method changes according to such environmental factors as computational capability, number of receivers, and network load. However, to change the processing method according to the environment, filtering results must be consistent among multiple processing methods. In this paper, we describe the implementation of an information filtering system that optimizes the processing method based on mathematical properties. This system automatically changes the processing method to the optimum method according to the environment.

KEY WORDS

information filtering, filtering function, ECA rule, optimization

1 Introduction

In recent years, the number of broadcast services has increased due to the introduction of new satellite-based services and the digitization of broadcasts[10]. In this environment, not only is the amount of data broadcasted increasing but the variety of broadcasts as well. However, users often only need small amounts of specific data, and it is very difficult to retrieve the information they are interested in from a large amount of broadcast data. Therefore, various mechanisms that automatically filter data have been proposed[2, 3, 5, 11, 13].

These filtering mechanisms usually filter data by the sequential processing method that filters data whenever the receiver receives new data. However, in sequential processing, since a receiver has to process a huge amount of received data sequentially, the processing cost of filtering becomes high. To reduce such costs, it is effective to use a bulk processing method that filters all received data in bulk or a parallel processing method that filters data by multiple receivers in parallel. The optimum filtering processing method changes according to such environmental factors as computational capability, number of receivers, and network load. However, to change the processing method according to the environment, the filtering results must be consistent among multiple processing methods.

In this paper, we describe the design and the implementation of an information filtering system that optimizes the processing method based on mathematical properties. This system automatically converts processing methods and selects the optimum method according to the environment. After selecting the optimum processing method, the consistency of filtering results is assured by the mathematical properties of filtering function[8] that represents filtering as a function. Moreover, using the ECA rule, that is a behavior description language of active databases[12], the system automatically detects such changes in the environmental situation as the occupancy rate of the network bandwidth and the CPU utilization ratio and selects the processing method that can keep the processing costs low. Thus, the system efficiently achieves a filtering process in various and fluid environments.

This paper is organized as follows. Section 2 explains the design of the proposed filtering system. Section 3 explains the implementation of the system, and Section 4 considers the system. Finally, we conclude this paper in Section 5.

2 System design

Our system judges and selects optimum processing methods automatically so that filtering results are equivalent. Figure 1 shows the outline of the system. The following shows the procedure of the system.

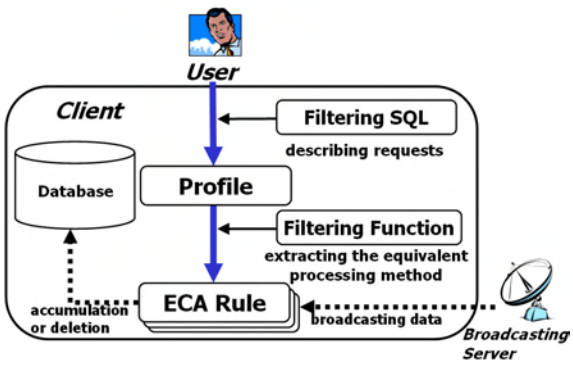

Figure 1. Outline of System

1. A user describes his/her demand as a profile. This system uses Filtering SQL[7] as the profile description language.
2. The processing methods where filtering results become equivalent are extracted by mathematical properties of filtering function in response to the user's profile.
3. The system generates ECA rules for actual processing toward all equivalent processing methods.
4. The system selects an optimum method according to current such environmental factors as computational capability and network load among equivalent processing methods. The system filters received data by using ECA rules. When the environment changes, it dynamically selects the optimum processing method by ECA rules for changing the method.
5. The system stores the filtered data in a database.

In the following, we describe each step of the detailed mechanism of our system.

2.1 Filtering SQL

The system uses Filtering SQL as a profile description language. Filtering SQL is a language for filtering based on SQL (Structured Query Language, which is the standard language for database management.) The following is the basic syntax of Filtering SQL.

```
EXTRACT    <attribute>
FROM       <data resource>
WHERE      <preferences>
```

EXTRACT specifies the attribute that the user wants. FROM specifies the broadcasting server(s). WHERE describes the user's preference. The user describes his/her preferences by the following syntax.

```
PREFER <GENRE>  [TO <GENRE>]
         [WITH Intensity | Littleness]
PROHIBIT  <GENRE>
```

The user can assign intensity level of preference by adding "WITH Intensity" and "WITH Littleness" to each preference. WHERE can also describe data management policies, such as data size restrictions and deletions.

2.2 Filtering Function

The system uses the mathematical properties of a filtering function[8] to assure consistent filtering results even if the system changes processing method. In the framework of filtering function, the properties of filtering are expressed as constraints satisfied by the filtering function, and we have defined the properties of the basic filtering processes. For example, the equivalent property states that the filtering results of sequential and batch processing methods are equivalent (SE: Sequential Equivalence). Therefore, for example, if filtering function that shows a certain filtering method satisfies sequential equivalence, we can assure consistency of filtering results, even if the system changes the processing method between the sequential and the batch processing in that filtering method.

2.2.1 Filtering processes

In the framework of filtering function, we deal with the following four processing methods[8].

[Sequential processing]
In a system that uses the sequential processing, the newly received data and the previously filtered results are merged and filtered every time.

[Batch processing]
In a system that uses the batch processing, a receiver accumulates broadcast data and filters them in bulk.

[Distributed processing]
In a system that uses the distributed processing, the received data set is divided into multiple arbitrary data subsets, and each subset is filtered separately before the results are merged.

[Parallel process]
In a system that uses the parallel processing, the merged filtering results of distributed processing are re-filtered.

2.2.2 Filtering methods

In the framework of filtering function, we deal with various filtering methods. The characteristics of the following filterings are especially clarified as representative filtering methods[9].

[Selection]
Filtering by selection specifies whether each broadcast data is to be stored. Examples of filtering by selection include keyword matching and filtering by threshold. Keyword matching carries out a logical operation on keywords included in the data; filtering by threshold estimates the value of data according to its content and carries out a logical operation on that value and a threshold set by the user.

[Ranking]
Filtering by ranking arranges the received data in order of

Table 1. The equivalent processing methods in each filtering method

The filtering method			The equivalent processing method
Selection			Sequential, Batch, Parallel, Distributed
Ranking			Sequential, Batch, Parallel, Distributed
Selection → Selection			Sequential, Batch, Parallel, Distributed
Ranking(n) → Ranking(n')	Combination of ranking based on same attributes		Sequential, Batch, Parallel
	Combination of ranking based on different attributes	$n \leq n'$	
		$n > n'$	None
Selection → Ranking			Sequential, Batch, Parallel
Ranking → Selection			None

Table 2. List of events

Name	Content
SELECT	Retrieval of data
INSERT	Insertion of data
DELETE	Deletion of data
UPDATE	Renewal of data
META_RECEIVE	Receiving metadata
CONTENT_RECEIVE	Receiving content data
RULE_RECEIVE	Receiving an ECA rule
NET_RECEIVE	Receiving data packets
TIMER	Firing a timer

Table 3. List of actions

Name	Content
QUERY	Database operation
SETTIMER	Set a timer
BROADCAST_SEND	Broadcast packets
FILE_SEND	Send a file
STORE_FILE	Store a file
DELETE_FILE	Delete a file
ADD_RULE	Add a rule to the system
ENABLE_RULE	Enable a rule
DISABLE_RULE	Disable a rule

importance according to the user's preferences and extracts a particular quantity of top-ranked data. Ranking is frequently used when the number of data to be stored is specified or some important data needs to be shown to the user in order of importance.

[Combination of selection and ranking]
More complicated processing can be performed by combining selection and ranking, such as filtering that combines two selection methods which perform keyword matching in a certain word after performing keyword matching in another word, and a filtering that combines two ranking methods which select the 10 highest data with an evaluation value order after selecting the 100 newest data.

Table 1 shows the processing methods in which the filtering results become equivalent by each filtering method[8, 9]. In the table, "filtering method 1 → filtering method 2" denotes that the system filters data by filtering method 2 after filtering method 1. "$n \leq n'$" denotes that the filtering method which the number of data (n) accumulated by the first ranking processing is less than the number of data (n') accumulated by the second ranking processing.

2.3 ECA rule

The system employs an ECA rule to describe the system behaviors. ECA rule is a behavior description language used in an active database system which is one of the database technologies[12]. Each ECA rule consists of three parts: the event, the condition, and the action. The event part describes an occurring event in the system. The condition part describes conditions for executing the following action. The action part describes operations to be carried out when the event occurs and conditions have been satisfied. Various advantages arise by using an ECA rule for a description of the system behaviors[6]. For example, since the system behaviors are described in an event-driven manner, we can describe the flexible processing that corresponds to changes of the situation such as the change of system environment and the arrival of data. Moreover, since system functions consist of a set of ECA rules, it is easy to customize the filtering policy and to change the processing method by adding or deleting ECA rules. Furthermore, the policy and the functions are movable to other terminals by transmitting ECA rules. Table 2 and Table 3 show the list of events and actions used in the system respectively.

This system uses ECA rules for the filtering process which extracts necessary data and also for the changes of the filtering processing method into the optimal one according to the environment. Hereafter, we describe the realization of these two processings.

2.3.1 ECA rule for filtering

This section describes the ECA rules which perform filtering. As an example, we describe the ECA rules which fulfill the following demand.

```
EXTRACT *
FROM    A broadcasting, B broadcasting
WHERE   GENRE = ''NEWS''
```

This description means that the system accumulates only

```
[Rule 1]
E  META_RECEIVE
C  NEW.RESOURCE = A broadcasting
   OR NEW.RESOURCE = B broadcasting
   AND NEW.GENRE = NEWS
A  QUERY("INSERT INTO accumulation table")
[Rule 2]
E  CONTENT_RECEIVE
C  DB.StoreTable.ID ID = NEW.ID
A  STORE_FILE
```

Figure 2. ECA rules for the sequential processing

```
[Rule 3]
E  META_RECEIVE
C  NEW.RESOURCE = A broadcasting
A  QUERY("INSERT INTO temporary table")
[Rule 4]
E  TIMER
A  QUERY("SELECT * FROM temporary table")
[Rule 5]
E  SELECT temporary table
C  NEW.GENRE = NEWS
A  QUERY("INSERT INTO accumulation table")
   STORE_FILE
[Rule 6]
E  META_RECEIVE
C  NEW.ADDRESS = requested receiver
A  QUERY("INSERT INTO accumulation table")
[Rule 7]
E  CONTENT_RECEIVE
C  NEW.ADDRESS = requested receiver
A  STORE_FILE
```

Figure 3. ECA rules for the distributed processing for main receiver

the data whose attribute "GENRE" is "NEWS" from broadcasts A and B. Since this demand is a kind of selection method, a user can obtain equivalent results even if the system changes the processing method among sequential processing, batch processing, distribution processing, or parallel processing. In response to this, the system automatically creates ECA rules for each processing method.

Here, we assume that a broadcast server transmits the metadata which includes information about content before broadcasting the content.

[Sequential processing]

Figure 2 shows ECA rules for filtering by the sequential processing. "NEW" is a variable for metadata, and "NEW. attribute" refers to the attributes of the metadata. When a receiver receives metadata, Rule 1 stores them when source of the broadcast is "A broadcasting" or "B broadcasting" and the "GENRE" is "NEWS". Rule 2 accumulates the content on a disk, if the "ID" exists in the database when the receiver receives the content.

[Distributed processing]

Distributed processing is realized with ECA rules (Figure 3) for a main receiver and the ECA rules (Figure 4) for an additional receiver.

Rule 3 stores data in a temporary table if the broadcast source is "A broadcasting" when the receiver receives

```
[Rule 8]
E  META_RECEIVE
C  NEW.RESOURCE = B broadcasting
A  QUERY("INSERT INTO temporary table")
[Rule 9]
E  TIMER
A  QUERY("SELECT * FROM temporary table")
[Rule 10]
E  SELECT temporary table
C  NEW.GENRE = NEWS
A  SEND_FILE main receiver
```

Figure 4. ECA rules for the distributed processing for additional receiver

```
E  TIMER
C  CPU_USAGE >= 50%
A  DISABLE_RULE sequential processing
   ENABLE_RULE batch processing
```

Figure 5. ECA rule for processing method conversion

metadata. Rules 4 and 5 transfer the metadata whose "GENRE" is "NEWS" to the accumulation table from the temporary table and accumulate the content on a disk when the set up timer fires. Rules 6 and 7 accumulate those data when the receiver receives metadata and content data, respectively. On the other hand, the rule of the additional receiver means that if broadcast source is "B broadcasting" on receiving metadata, it stores data in the temporary table. When the set up timer fires, it transmits the metadata and the content whose "GENRE" is "NEWS" to the main receiver.

[Batch processing]

It can be described as Rules 3, 4, and 5. We omit a detailed explanation.

[Parallel processing]

It can be expressed by adding the ECA rules for one more filtering on the results to Rules 3-7. We omit a detailed explanation.

2.3.2 ECA rule for changing processing method

As described in Section 2.2, equivalent processing methods are clarified by the mathematical properties of the filtering function. Therefore, ECA rules for selecting the method decide the optimum one in response to the change of situation. For example, the ECA rule shown in Figure 5 checks CPU usage for every fixed time while performing the sequential processing, and if CPU usage increases more than 50 %, the system changes the processing method to the batch processing.

In this system, the processing method is flexibly convertible by such environmental parameters as receiving costs. For example, as shown in Figure 6, the system can reduce processing costs with sequential processing by changing to the batch processing when the throughput of a

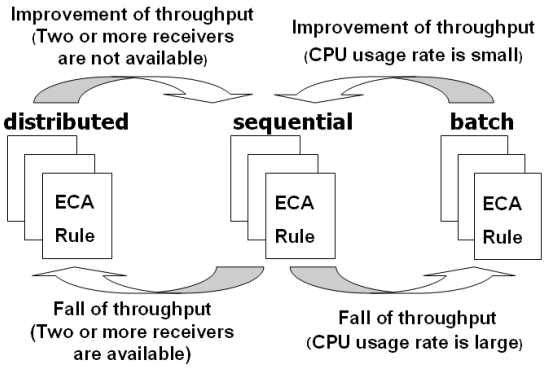

Figure 6. Change of Processing method

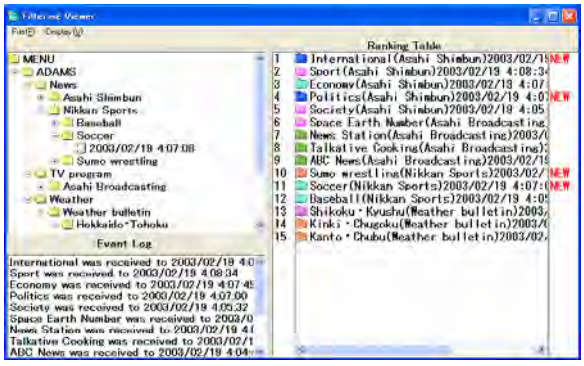

Figure 7. Viewer of stored data

receiver declines or by changing to the distributed processing when two or more receivers are available.

3 Implementation

We have implemented a prototype of a filtering system explained in the previous sections. We used Microsoft Visual Basic 6.0 and Visual C++ 6.0 on Windows XP for the implementation. The system receives and filters data items of ADAMS[1] and bitcast[4], which are data broadcast services via ground-based broadcasting.

A client system consists of four parts: a rule processing engine that processes ECA rules; a database that manages metadata and accumulated data; a viewer for referring to the stored data; and an editor that a user inputs his/her request. We used A-WEAR[6] as a rule processing engine. Since we can extend functions in A-WEAR by adding plug-ins, we implemented the function in our system as several plug-ins for A-WEAR. For example, we implemented a plug-in that analyzes the metadata and a plug-in that stores or deletes a file. Figure 7 shows a snapshot of the viewer which displays filtering results in our prototype. In the figure, the tree view on the left side displays the filtered data items in the hierarchical structure. At the same time, the system shows the ranking of stored data items and event logs, such as storing and deleting data items. This system changes the processing method and asks for the filtering processes of other receivers automatically.

4 Consideration

In this section, we mention the advantages of our system on the basis of processing costs. Next, we present related work and consider our system by comparing with them.

4.1 Processing Costs

In the sequential processing, since the receiver has to sequentially process a huge amount of received data, the processing costs of such filtering as calculation capability and calculation time become high. Therefore, when the throughput of a receiver falls on the sequential processing, it can reduce processing costs by changing it to the batch processing, which does not continuously need to filter received data. Furthermore, in a type of filtering that considers the correlation with data filtered together such as “a new weather report is more important than an older weather report”, although the sequential processing must calculate the degree of correlation for all relevant data whenever it receives data, the batch processing only needs to calculate after collecting some data. The distributed and the parallel processing can reduce receiving costs and network load. Thus, if two or more receivers are available, the system should change to the distributed/parallel processing method for the efficiency of the receiving cost. On the contrary, the use of the sequential processing enables users to view the latest data immediately because the system does not wait for a certain period of time. Therefore, if a user wants to get filtered data as soon as possible, the system should select the sequential processing even though it requires higher processing costs. Thus, this system chooses the optimum processing method according to the environment.

4.2 Related Work

In FBDA (Filtering mechanism Based on Distance Approximation)[5], by employing triangle inequality, each receiver lays received data in a metric space in compliance with their content, carries out a logical operation on a particular threshold value and the distance between the received data and the data the user is interested in, and then stores them only if the distance is less than the threshold. FBDA reduces processing costs for the choice determination of data by performing distance calculation based on an approximation algorithm. On the other hand, our system reduces processing costs not from the selection algorithm of data but on the basis of the processing method. In addition, our system can further reduce processing costs by considering the selection algorithm of filtering.

SIFT[13] and XFilter[2] are filtering servers located in the middle of the broadcast source with user filters. SIFT extract data if the vector product of the vectors representing the data and the user’s preference exceeds a particu-

lar threshold, and it improves performance and availability by developing efficient indexing mechanisms for profiles. XFilter provides highly efficient matching of XML documents to large numbers of user profiles. In XFilter, user interests are represented as queries using XPath language. XFilter can quickly locate and examine relevant profiles by using a sophisticated index structure and by converting XPath queries into a Finite State Machine (FSM) representation.

Since these related works do not consider changes of the processing method dynamically, it is difficult to carry out efficient processing anytime in response to the environment. Our system can change the processing method dynamically and keep processing costs low. Furthermore, our system is superior to previous research because it assues the consistency of filtering results by using the mathematical properties.

5 Conclusion and Future Work

In this research, we have designed and implemented an information filtering system that changes its processing method dynamically in response to the environment. It assures the consistency of filtering results among multiple processing methods by using the mathematical properties of filtering function. This system can reduce processing costs by converting the processing method dynamically as changing it to the batch processing when the throughput of a receiver declines.

In the future, we plan to evaluate our system on the basis of the calculation time and capability. Moreover, our system currently uses an empirical value as environmental parameters for the threshold of conversion in processing methods. Therefore, to realize the optimum conversion, we plan to calculate the optimum value of environmental parameters mathematically.

Acknowledgments

This research was supported in part by “The 21st Century Center of Excellence Program”, Special Coordination Funds for promoting Science and Technology, and Grant-in-Aid for Scientific Research (B) numbered 15300033 of the Ministry of Education, Culture, Sports, Science and Technology, Japan.

References

[1] ADAMS Homepage: http://www.tv-asahidata.com/.

[2] M. Altinel and M. J. Franklin, Efficient Filtering of XML Documents for Selective Dissemination of Information, *Proc. VLDB2000*, Cairo, Egypt, 2000, 53–64.

[3] N. J. Belkin and W. B. Croft, Information filtering and information retrieval: two sides of the same coin?, *Communications of the ACM*, 35(12), 1992, 29–38.

[4] Bitcast Homepage: http://www.bitcast.ne.jp/.

[5] X. Kan, T. Ohwada, K. Asada, A. Iizawa, and K. Furuse, FBDA: A Filtering Mechanism Based on Distance Approximation, *Proc. DASFAA 2001*, Hong Kong, China, 2001, 162–163.

[6] M. Miyamae, T. Terada, M. Tsukamoto, and S. Nishio, Design and Implementation of an Extensible Rule Processing System for Wearable Computing, *Proc. MoviQuitous 2004*, Boston, USA, 2004, 392–400.

[7] R. Sawai, T. Terada, M. Tsukamoto, and S. Nishio, FilteringSQL: A User Request Description Language for Filtering, *Workshop DEWS2000*, Shiga, Japan, 2000, CD-ROM, in Japanese.

[8] R. Sawai, M. Tsukamoto, Y. H. Loh, T. Terada, and S. Nishio, Functional properties of information filtering, *Proc. VLDB2001*, Roma, Italy, 2001, 511–520.

[9] R. Sawai, M. Tsukamoto, T. Terada, and S. Nishio, On Selection and Ranking in Filtering Function, *IPSJ Transactions on Databases*, 43(SIG12), 2002, 80–91, in Japanese.

[10] Satellite Magazine: http://www.satemaga.co.jp.

[11] A. H. Timothy and M. Alistair, The design of a high performance information filtering system, *Proc. SIGIR1996*, Zurich, Switzerland, 1996, 12–20.

[12] J. Widom and S. Ceri, *Active Database Systems*, (San Fransisco: Morgan Kaufmann Publishers Inc., 1996).

[13] T. W. Yan and H. Garcia-Molina, The SIFT Information Dissemination System, *ACM Transactions on Database Systems*, 24(4), 1999, 529–565.