JOURNAL OF MEDICAL INFORMATICS & TECHNOLOGIES Vol.5/2003, ISSN 1642-6037

query language, data stream processing, database management system, fetal monitoring

Michał WIDERA[*], Janusz JEŻEWSKI[*], Ryszard WINIARCZYK[**], Janusz WRÓBEL[*], Krzysztof HOROBA[*], Adam GACEK[*]

DATA STREAM PROCESSING IN FETAL MONITORING SYSTEM: I. ALGEBRA AND QUERY LANGUAGE

The paper presents the algebra and declarative query language developed for fetal monitoring system MONAKO. At present, the system architecture is centralised. Recording, analyzing and visualization of data are carrying out in the central unit. Therefore, the system cannot scale in a simple manner. Presented solution enables to define query based on data streams that makes the updated answers currently available. Applying database management system that carries out its tasks based on presented assumptions enables the construction of monitoring system of distributed architecture

1. INTRODUCTION

Application of the n-tier architecture or client-server is taking into consideration in the classic computer system design. Separate Database Management System (DBMS) control database in these systems. The application communicates with the database by DBMS in Declarative Query Language. Biomedical monitoring computer systems are in the scope of our interest. The work that has been carried out for many years in the Institute of Medical Technology and Equipment on the fetal monitoring system MONAKO becomes the basis for the research on introducing more flexible and efficient architecture [1,2]. The system allows continuous and precise monitoring of the fetus condition. It leads to reduction of complications at childbirth. This system has introduced in many obstetric centres in Poland. At present, the system architecture is centralised. Recording, analyzing and visualization of data are carrying out in the central unit. Therefore, the system cannot scale in a simple manner.

MONAKO system records signals in rough disk files. Additional information is stored in relational database set. This is the main weakness of system construction. Stored data are under control of various processes exposing them for potential integrity loss. Our goal is to introduce the new monitoring system based on the architecture where the recorded data will be control by DBMS [3]. It should allow initial analysis and process of the recorded biophysical values. The database

[*] Institute of Medical Technology and Equipment, 118 Roosevelt St, 41-800 Zabrze, POLAND, tel./fax: (32) 2716013/2712312, e-mail: michal@itam.zabrze.pl
[**] Institute of Theoretical and Applied Informatics, 5 Bałtycka St, 44-100 Gliwice, POLAND tel./fax: (32) 2317319/2317026, e-mail: office@iitis.gliwice.pl

access will be realized by the drawn up declarative query language. The new system efficiently manages available resources and enables the construction of distributed monitoring system.

Data stream is an unbounded bag of elements (*a,t*) where the first element contains measured value and the second – time of occurrence. In the MONAKO system the biomedical data streams can be presented in the form of time series. Time series are in the form of a bag of elements ($\{a_n\}$,Δ), where the first element is data sequence and the second is a real number that determines time interval between the consecutive elements of the sequence. Every time series can be described with the help of data stream, however both definitions are not equivalent to each other.

The specified character of recorded data determines the selection of an appropriate database management system. Unfortunately, the relational database management system cannot be applied because of its limited possibilities of fast recording data in big quantities. The limitations were one of the reasons for starting the search of alternative data model adjusted to record data stream. Nowadays several projects, that in the future will probably enable analysis and recording of biomedical signals, can be found [4]. Ongoing research on data stream management system has not brought any sufficient and universal solutions so far. The goal of the world research is creating more flexible and natural way of gathering information from data streams [5,6]. In the classic relational database, the user inserts and queries data from data sets. This model of database management system is calling Human Active Database Passive. In case of data stream management system the user does not take part in the process of inserting data. This model of database management system is calling Database Active Human Passive [5]. Data are loaded into a system directly from the external devices by means of procedures linked to the database management system. Created queries return continuously updated results. This query is calls continuous query. Several systems implementing continuous query have been recently introduce: CQL, StreaQuel and Aquery. Each of the languages has an extended syntax of the SQL language at its disposal and can determine data window in the stream. The technique of sliding window is a commonly used way of presenting selected area of the data stream in time [7].

2. ALGEBRA AND DECLARATIVE QUERY LANGUAGE

After determining the character of incoming data into fetal monitoring system several requirements about database management system have been determined:

- Database management system should, in a simple way, enable implementing initial procedures of processing biomedical signals (signal filtering, statistical analysis, etc.).
- Database management system should enable separated analysis and data recording in many units presenting their resources by consistent and homogeneous data schema.
- Recorded and analysed biomedical signals should be made available for continuous query and presented in a graphic form of curves and markers.
- The system should ensure integrity and consistency of collected database.
- Besides current analysis, database management system should enable retrospective analysis of collected database.

Unfortunately none of the systems meets all the listed requirements [5,6]. Moreover, most of research being carried out is still in the project phase. The application of data stream management system in monitoring of biomedical processes has not been considered in the so far published articles [4,6]. Therefore we started working on creating our own data stream management system to be applied in the fetal monitoring system MONAKO. It is necessary to define some notions in order to present our solution. By data stream schema A we will understand the list of attributes of individual elements (tuples) of sequence $\{a_n\}$. The schema is written in the following way $A(A_1,A_2,A_3)$, where A_1,A_2,A_3 represent areas where data are stored. It has to be note that the relation of order within the list is compulsory. The order presented in the schema is binding.

Considering data stream as a bag of elements $(\{a_n\},\Delta)$ and assuming that all the operations realized in database management system will refer to the set of data streams we determined the following set of operations necessary to implement the analytical procedures in MONAKO system: Interlace Deinterlace, Sum, Difference, Aggregation/Serialisation, Projection and Selection. Some of them are analogous to operators presented in Aurora query algebra [6].

Interlace: By the operation of interlace of data streams $(\{a_n\},\Delta_a)$ and $(\{b_n\},\Delta_b)$ we will understand the operation marked by operator # creating a new data stream $(\{c_n\},\Delta_c)$. As a result of applying the interlace operation of two data streams of different intervals Δ one data stream is created. Its interval is smaller then two primary values. The resulting sequence $\{c_n\}$ and the interval Δ can be calculated with the help of the following equations:

$$c_n = \begin{cases} b_{n-\lfloor (n+1)\Delta \rfloor} & \text{for } \lfloor n\Delta \rfloor = \lfloor (n+1)\Delta \rfloor \\ a_{\lfloor n\Delta \rfloor} & \text{for } \lfloor n\Delta \rfloor \neq \lfloor (n+1)\Delta \rfloor \end{cases} \quad \textit{where } \Delta = \frac{\Delta_b}{\Delta_b + \Delta_a} \textit{ and } \Delta_c = \frac{\Delta_a \Delta_b}{\Delta_a + \Delta_b} \qquad (1)$$

This operation can be explained in the two exampled streams. The first A=[1,2,3,...],1 includes sequence of natural numbers added once in every second. The second B=[a,b,c,d,...],2 includes letters of Latin alphabet added once in every two seconds. Operation of interlaces A#B creates the sequence C=[a,1,2,b,3,4,c,5,6,...],0.66 whose elements are added once in 0.66 second. The streams created as a result of interlacing streams A and B for different coefficients are presented in the Table 1. In stream A the elements are added once in every second and in the stream C – elements are added once in 0.5 second. In stream D elements are added once in every two seconds.

Tab.1. Operation of interlace between two data streams C:=A#B

C=A#B	A,0.5	A,1	A,2
B,0.5	1 a 2 b 3 c 4 d 5 e *C,0.25*	a b 1 c d 2 e f 3 g *C,0.33(3)*	a b c d 1 e f g h 2 *C,0.4*
B,1	a 1 2 b 3 4 c 5 6 d *C,0,33(3)*	1 a 2 b 3 c 4 d 5 e *C,0.5*	a b 1 c d 2 e f 3 g *C,0.66(6)*
B,2	a 1 2 3 4 b 5 6 7 8 *C,0.4*	a 1 2 b 3 4 c 5 6 d *C=0.66(6)*	1 a 2 b 3 c 4 d 5 e *C,1*

Assuming that there is stream A with the schema A(a) and stream B with the schema B(b), the example record of presented operation in declarative query language looks as following:

SELECT a,b **AS** C **FROM** A#B

Deinterlace: Assuming that sequence C was created as a result of operation C:=A#B, deinterlacing operation will be marked by A:=C&Δ_b;B or B:=C&Δ_a;A. As a result of this operation two data streams are created – first including elements of stream A and second including elements of stream B. The proposed operation enables recreating primary data streams based on the stream that was created as a result of interlacing of these streams. The function selecting elements of streams A and B from stream C may be presented in the following way :

$$b_n = c_{n+\left\lfloor \frac{n\Delta_b}{\Delta_a} \right\rfloor} \; \textit{and} \; a_n = c_{n+\left\lceil \frac{(n+1)\Delta_a}{\Delta_b} \right\rceil} \; , \Delta_a = \frac{\Delta_b \Delta_c}{\left|\Delta_b - \Delta_c\right|} \; \textit{and} \; \Delta_b = \frac{\Delta_a \Delta_c}{\left|\Delta_a - \Delta_c\right|} \tag{2}$$

This operation of deinterlace may be presented by following stream example C=[a,1,2,b,3,4,c,5,6,...],0.66 whose elements are added once in 0.66 second. The deinterlace operation A=C&2 creates the stream A=[1,2,3,…],1 including the natural number sequence added once in one second. As a result of this operation A:=C&2 the stream in 'the reminder' form is created B=[a,b,c,d,…],2 including the letters of alphabet added once in every two seconds. Assuming that there is stream C with the schema C(a) that was created as a result of operation A#B an example record of presented deinterlace operation in declarative query language looks as follows:

SELECT a **AS** A **FROM** C&2

Sum: By sum operation of given streams ($\{a_n\}$,Δ_a) and ($\{b_n\}$,Δ_b) we understand the operation marked by operator + and creating a new data stream ($\{c_n\}$,Δ_c). The operation enables linking the two data streams of different intervals Δ in one data stream whose interval will be the minimal value of primary data streams. The sequence $\{c_n\}$ can be presented with the help of the formula:

$$c_n = \begin{cases} a_n \mid b_{\left\lfloor \frac{n\Delta_a}{\Delta_b} \right\rfloor} \cdots \text{for } \Delta_c = \Delta_a \\ a_{\left\lfloor \frac{n\Delta_b}{\Delta_a} \right\rfloor} \mid b_n \quad \text{for } \Delta_c = \Delta_b \end{cases} , \textit{where } \Delta_c = \min(\Delta_a, \Delta_b) \tag{3}$$

Where by operation $a_n|b_n$ we are understand schema union of streams A and B. The proposed operation enables the link of data existing in both sequences where every element is linked and duplicated when needed in accordance with Δ of every stream. Assuming that data streams presented in interlace operation exist, as a result of operation A+B stream C=[1a,2a,3b,4b,5c,6c,7d,8d,9e,...],1 will be created. The streams A and B summed up for different coefficients Δ are present in the Table II. Every tuple of stream C is represent in the table by two values placed horizontally. Some of the elements duplicates in the initial streams.

Tab.2. Operation of sum of two data streams C := A+B

C=A+B	A,0.5	A,1	A,2
B,0.5	1 2 3 4 5 6 7 8 9 0	1 1 2 2 3 3 4 4 5 5	1 1 1 1 2 2 2 2 3 3
	a b c d e f g h i j	a b c d e f g h i j	a b c d e f g h i j
	C,0.5	*C,0.5*	*C,0.5*
B,1	1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0	1 1 2 2 3 3 4 4 5 5
	a a b b c c d d e e	a b c d e f g h i j	a b c d e f g h i j
	C,0.5	*C,1*	*C,1*
B,2	1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0
	a a a a b b b b c c	a a b b c c d d e e	a b c d e f g h i j
	C,0.5	*C,1*	*C,2*

Assuming that there is stream A with the schema A(a) and stream B with the schema B(b) an example record of presented operation in declarative query language looks as follows:

SELECT a,b **AS** C **FROM** A+B

Difference: Assuming that stream **C** was created as a result of operation of sum C:=A+B, the difference operation is determined as A:=C–(a,b). As a result of this operation the elements of data stream belonging to stream B will be separated and, in case, when tuples of stream A were duplicated in the process of summing – the duplicated elements will be deleted. The proposed operation enables determining primary data streams based on the stream that was created as a result of summing up of these streams. This operation presents in the following way:

$$a_n = \begin{cases} c_n - B(B_1, B_2, B_3, ...) & \text{for } \Delta_b \geq \Delta_a \\ c_{\left\lceil \frac{n\Delta_a}{\Delta_b} \right\rceil} - B(B_1, B_2, B_3, ..) & \text{for } \Delta_b < \Delta_a \end{cases} \qquad (4)$$

Where by operation c_n – B(B1,B2,B3,...) we understand deleting from stream schema C the elements belonging to stream B. Assuming that there is stream with the schema C(NUMBER n1, CHAR c1) containing values: C=[1a,2a,3b,4b,5c,6c,7d,8d,9e,...], after carrying out the operation A=C–(1,2) we receive the stream A=[1,2,3,4,5...],1. Assuming that there is stream C with the schema C(a,b) that was created as a result of operation A,1+B,2 an example record of the presented operation in declarative query language looks as following:

SELECT a **AS** A **FROM** C-(1,2)

Aggregation/Serialization: Despite the fact that this operation was described with two notions there is only one operation. This operation is feasible only in a given data schema. Assuming that the stream A includes tuples of the schema $A_1,A_2,A_3,...,A_n$ and the stream B includes tuples of the schema $B_1,B_2,B_3,...,B_m$ and the attribute domains A_i and B_i are common by aggregation/serialization we will understand the operation of creating stream B on the basis of stream A. If n is smaller than m there will be aggregation. If m is smaller than n there will be serialization. This operation will be marked by

$$B := AGSE(A, (A1,A2,..) , step) \qquad (5)$$

The argument of this operation is a new schema of a data stream and a step of the operation. This operation enables generalization and implementation of technique of sliding window with the applied algebra. An example data stream whose schema can be presented in the following way will be considered: InStream (NUMBER n1, NUMBER n2, NUMBER n3). The stream includes the tuples filled with natural numbers, for example S=[(1,2,3),(4,5,6),(7,8,9),...],1. As a result of operation A=AGSE(S,(NUMBER),1) we receive data stream A with the schema A(NUMBER n) and form: **A**=[1,2,3,4,5,6,7,...],0.33(3). Typical process of serialization is present in Fig.1.

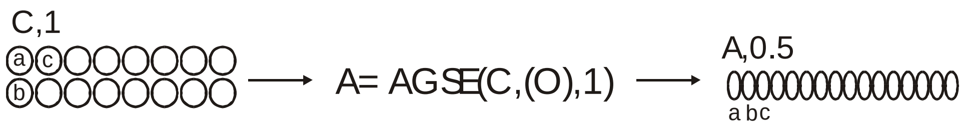

Fig.1 Serialization of data stream

As mentioned before this operation enables aggregation as well (Fig.2). AGSE operation enables applying sliding window technique in data streams. Assuming that the number of elements in schema B is bigger than the given step, presenting operation that creates data stream is possible. The elements of this stream include consecutive 'pictures' of sliding window (Fig.3).

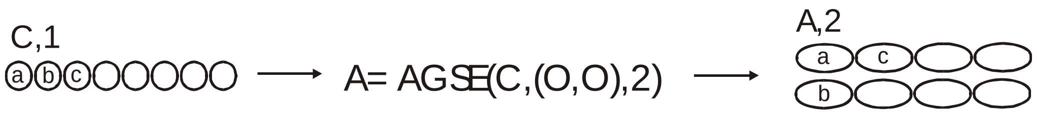

Fig.2 Aggregation of data stream

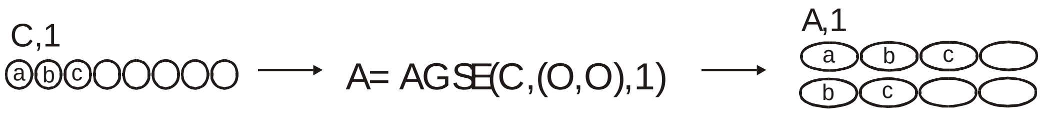

Fig.3 Aggregation of sliding window

Assuming that stream C with schema C(NUMBER n1, NUMBER n2) exists, an example operation of aggregation can be recorded in declarative query language in the following way:

SELECT AGSE(C, **NUMBER**<10> , 1) **FROM** C

Definitions of projection and selection are analogous as in case of relation algebra. The differences are minor and they mainly refer to interval Δ definition.

Projection: Assuming that stream A includes tuples with schema: $A(A_1,A_2,\dots,A_n)$, by projection of stream A into attributes set $(A_1, A_2,\dots,A_m)$ we will understand operation resulting in data stream with schema $A'(A_1,A_2,\dots,A_m)$ where n>m. This operation will be record as:

$$A' := A\ (A_1,A_2,\dots,A_n) => (A_1,A_2,\dots,A_m) \qquad (6)$$

Assuming that we want to receive the stream with schema OutStream (NUMBER n1, NUMBER n2) from the stream with schema InStream (NUMBER n1, NUMBER n2, CHAR c1), we have to carry out the projection operation. We record this operation by:

OutStream :=   InStream (NUMBER n1 , NUMBER n2, CHAR c1) (7)

=> (NUMBER n1, NUMBER n2)

As a result of this operation, the interval Δ remains unchanged, retaining its value that it had in a primary stream. Assuming that the stream C with schema C(a,b) exists, an example of projection can be recorded in the declarative query language in the following way:

SELECT a **FROM** C

Selection: By selection operation or Θ-restriction, where Θ – means comparison operator (=, >, etc.,) we will understand operation resulting in stream with identical data schema. However, the determined condition XΘY will be fulfilling for all the elements of the resulting stream. This operation will be marked with:

$$C := A(A_1,A_2,\dots,A_n),\ X\Theta Y \qquad (8)$$

Assuming that in the stream with schema InStream (CHAR C1, NUMBER N1) in the field N1 there are natural numbers, carrying out selection operation in the stream:

OuStream := InStream (CHAR C1, NUMBER N1), N1>1 (9)

Creates data stream whose tuples in the field NUMBER N1 will contain the value bigger than 10. The tuples that values are smaller than 10 will be delete from the stream. After the operation has been carrying out, the interval Δ cannot be determined based on intervals Δ of the primary stream. Therefore, we assume that the system determines it during working time based on the moment of data occurrence in the resulting stream. This way of coefficient determining will be call dynamic. Assuming that the stream C with schema C(a) exists, the example selection operation can be recorded in declarative query language in the following way:

SELECT a **FROM** C **FILTER** C **BY** a > 10

It is worth to notice that in the presented declarative query language there is no WHERE clause. FILTER BY clause intended exclusively to determine the conditions of deleting data plays similar, but very much limited function.

3. CONCLUSIONS

The presented algebra and declarative query languages are the basis of the proposed stream management system. The presented query language will realize access to the recorded biomedical signals. The presented example of application has shown that the implementation of a query plan of a declarative query language is possible. Database management systems drawn up so far are not efficient enough in biomedical applications. Ongoing research on query languages based on data streams resulted in creating some new concepts - including ours. Our solution enables to define query based on data streams that makes the updated answers currently available. Differently than in the presented so far solutions, we do not enable – at present – linking of recorded in data stream information with the information stored in relational database. It is also worth to note that in the presented declarative query language the WHERE clause does not occur. The FILTER BY clause intended exclusively to determine the conditions of data filtering plays similar but very limited function. The presented declarative query language required drawing up and presenting assumptions of data algebra intended to analyse and record of biomedical signals. The ongoing researches are carried out in the framework of MONAKO system project. At present, its architecture is centralised. Applying database management system that carries out its tasks based on presented assumptions enables the construction of monitoring system of distributed architecture.

BIBLIOGRAPHY

[1] JEŻEWSKI J. WRÓBEL J. HOROBA K. GRACZYK S. GACEK A. SIKORA J. Computerized perinatal database for retrospective qualitative assessment of cardiotocographic traces. In B. Richards (Ed.) Current Perspectives in Healthcare Computing, BJHC Ltd., pp.187-196. Weybridge, 1996.

[2] JEŻEWSKI J. WRÓBEL J. HOROBA K. Monitorowanie zagrożeń płodu wspomagane komputerem. M. Nałęcz (Red.) Biocybernetyka i Inżynieria Biomedyczna 2000 – Tom 7. Systemy komputerowe i teleinformatyczne w służbie zdrowia, Akademicka Oficyna Wydawnicza EXIT, pp. 97-119, Warszawa, 2001.

[3] WIDERA M. WINIARCZYK R. JEŻEWSKI J. GACEK A. WRÓBEL J. K. HOROBA: Strumienie danych w systemie monitorowania, analizy i klasyfikacji sygnałów biomedycznych, STUDIA INFORMATICA Volume 24, Number 3 (55), pp. 35-45, Gliwice, 2003.

[4] BABCOCK B., BABU S., DATAR M., MOTWANI R. WIDOM J. Models and Issues in Data Stream Systems, Invited paper in Proc.of the 2002 ACM Symp. on Principles of Database Systems (PODS 2002), pp. 1-16, June 2002.

[5] ARASU A. BABU S. WIDOM J. An Abstract Semantics and Concrete Language for Continuous Queries over Streams and Relations, Proc. 9th Int. Conf. Data Base Programming Languages. pp. 1-11, Potsdam, Germany, September 2003.

[6] ABADI D., CARNEY D. CETINTEMEL U. CHERNIACK M. CONVEY C. LEE S. STONEBRAKER M. TATBUL N. ZDONIK S. Aurora: A New Model and Architecture for Data Stream Management. In VLDB Journal, August 2003.

[7] MOTWANI R. WIDOM J. ARASU A. BABCOCK B. BABU S. DATAR M. MANKU G. OLSTON C., ROSENSTEIN J. VARMA R.. Query Processing, Resource Management, and Approximation in a Data Stream Management System In Proc. of the 2003 Conference on Innovative Data Systems Research (CIDR), pp.245-256, Pacific Grove, California, January 2003.

[8] GOLAB L., OZSU M.T.: Issues in data stream management, ACM SIGMOD Record Volume 32, Issue 2, pp.5-14, June 2003.

This study was supported by the State Committee for Scientific Research, Warsaw, Poland (KBN Grant No. 4 T11E 006 22).