Available online at www.sciencedirect.com
ScienceDirect
Computational Biology and Chemistry
Computational Biology and Chemistry 31 (2007) 143–147
www.elsevier.com/locate/compbiolchem

Database notes

A method of microarray data storage using array data type

Lam C. Tsoi [a], W. Jim Zheng [b,*]

[a] *Bioinformatics Graduate Program, College of Graduate Study, Medical University of South Carolina, 135 Cannon Street, Suite 303, Charleston, SC 29425, United States*
[b] *Department of Biostatistics, Bioinformatics and Epidemiology, and Bioinformatics Core Facility, Hollings Cancer Center, Medical University of South Carolina, 135 Cannon Street, Suite 303, Charleston, SC 29425, United States*

Received 3 November 2006; accepted 4 January 2007

Abstract

A well-designed microarray database can provide valuable information on gene expression levels. However, designing an efficient microarray database with minimum space usage is not an easy task since designers need to integrate the microarray data with the information of genes, probe annotation, and the descriptions of each microarray experiment. Developing better methods to store microarray data can greatly improve the efficiency and usefulness of such data. A new schema is proposed to store microarray data by using array data type in an object-relational database management system—PostgreSQL. The implemented database can store all the microarray data from the same chip in an array data structure. The variable-length array data type in PostgreSQL can store microarray data from same chip. The implementation of our schema can help to increase the data retrieval and space efficiency.

Keywords: Microarray; Database schema; Array data type; PostgreSQL

1. Introduction

The number of available public microarray databases is dramatically increasing as the cost of computer hardware decreases (Galperin, 2006). However, the development of a good microarray database can only be achieved through a well-designed relational schema, which helps the system to manage the data effectively, and increases the data retrieval speed. In current microarray database (Ball et al., 2005; Cheung et al., 2002; Killion et al., 2003; Sherlock et al., 2001), it is common for a relational schema of a microarray database to consist of tables to store the accession number and description of gene (*GENE* table), the description of the chip (*CHIP* table), the probe name and the expression value (*DATA* table), and the experiment information (*EXPERIMENT* table). The schemas for most of the databases would tend to store the probe's expression value of each experiment in a record (a row) in the table *DATA*. For instance, in the Stanford (Ball et al., 2005; Sherlock et al., 2001) and Longhorn microarray databases (Killion et al., 2003), each expression value identified by the experiment and the probe set (probe's name) is stored as a record (each Expression Value Per Record, EVPR). For example, if we use Y98 (Affymetrix), a chip with 9335 probe sets to study the gene expression of yeast (*Saccharomyces cerevisiae*), performing 8 experiments will yield $9335 \times 8 = 74{,}680$ records in the table *DATA*. The exponential growth of microarray data will eventually makes the table extremely large and becomes unmanageable (Sarkans et al., 2005). As a result, alternative method has been proposed in ArrayExpress to store expression values in NetCDF format and keep the whole microarray data as BLOBs in the database (Sarkans et al., 2005). In the schema like SMD, for a researcher to query all the expression values of a particular probe set such as AFFX-MurIL2 from Y98, the system would have to run through each experiment done on the Y98 to find the correct probe set. On the other hand, the query has to go through the BLOBs in NetCDF format and find the right record when query ArrayExpress database, or the NetCDF has to be pre-computed to store expression values in the data warehouse. Since the experiments performed using the same chip will have the same number of records, here we propose a new schema by using the array data type to store the expression values of microarray experiments, which will increase the efficiency of space usage and data retrieval.

* Corresponding author. Tel.: +1 843 876 1123; fax: +1 843 876 1126.
E-mail addresses: tsoi@musc.edu (L.C. Tsoi), zhengw@musc.edu (W.J. Zheng).

1476-9271/$ – see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compbiolchem.2007.01.004

2. Material and methods

Our database is implemented in PostgreSQL, an object-relational open-source database management system (DBMS) that can be freely downloaded at http://www.postgresql.org/. In order to compare the performances of EVPR and our proposed array data type schemas, we created the tables and attributes that would be necessary for each type of schema (Section 3), and uploaded all the information of chip, probset, and gene, and the experimental results to both schemas. The two databases were implemented using PostgreSQL in a PC with Pentium(R) 4 CPU (2.4 GHz) and 1.00 GB of ram. Microarray data from eight experiments of chip Y98 were used, and another seven experiments with artificially generated experiment data (by random number) were added. We also included two additional chips (called Y99 and C50), which also have 9335 probsets, with 15 (for Y99) and 20 (for C50) sets of artificially generated experiment data. The overview of the number of records needed for each table of the two schemas is shown in Table 1. We tested the performance of our schema by comparing the query times of retrieving different types of query to that of the schema storing each expression value per record (EVPR), and the core tables for this schema is shown in Fig. 1B. The SQL commands used to test the databases' performance are shown in Table 2.

Table 1
Simulated data in the PostgreSQL for two different schemas

	Table	Number of rows
Expression per row	CHIP	3
	PROBE	28,005
	EXPERIMENT	50
	GENE	6,776
	DATA	466,750
Array type	CHIP	3
	EXPERIMENT	50
	GENE	6,776
	MICROARRAY	28,005

The core tables in each of the schema and the number of records needed to store the data are shown.

3. Results

3.1. Microarray data storing in array data type schema

PostgreSQL supports most SQL commands and different data types, including array data type. In PostgreSQL, array data type can have a variable length and allow index access, so it can be used efficiently to store data. Fig. 1A shows the relational diagram of the schema, and it consists of four tables: *MICROARRAY*, *CHIP*, *GENE* and *EXPERIMENT*. In our schema, the table

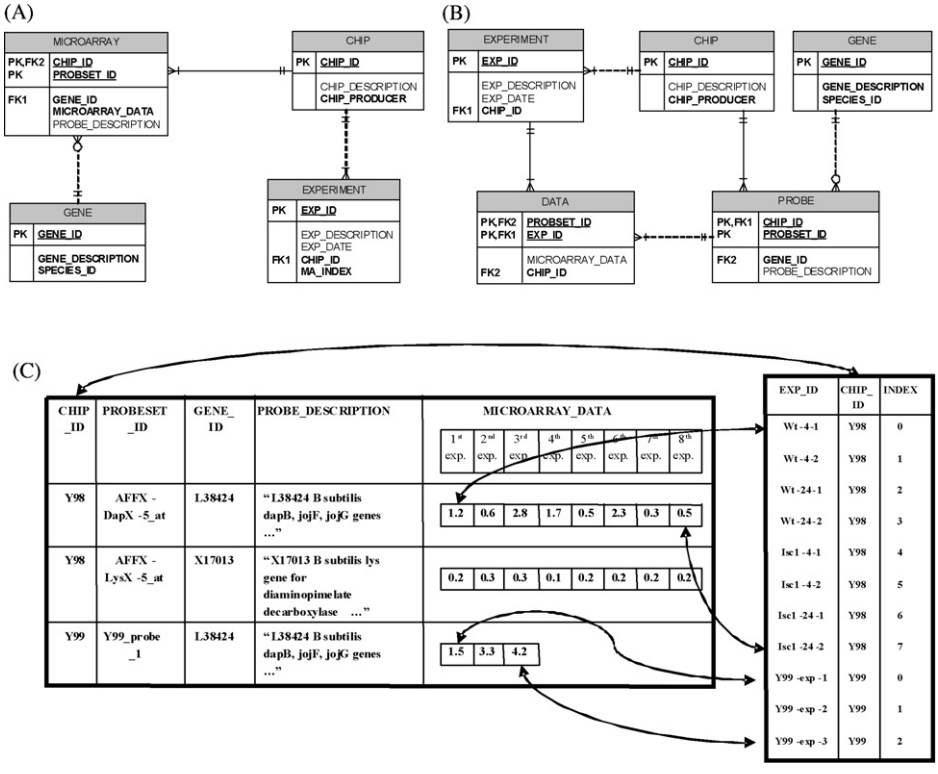

Fig. 1. Using array data type to store microarray data. The top shows the schemas of the microarray databases: (A) is the schema that uses array data type; (B) is the schema that stores expression value per row (EVPR). (C) A diagram to illustrate how microarray data are stored in the array data type schema. The left table is the table *MICROARRAY*, and the table *EXPERIMENT* is on the right. In the table *MICROARRAY*, two example probes from chip Y98 and one example probe from chip Y99 are shown. The attribute MICROARRAY_DATA is an array data type, with variable array length. The table *EXPERIMENT* shows the eight experiments done on Y98 and three experiments done on Y99. The attributes EXP_DESCRIPTION and EXP_DATE are not shown here. The arrows show the relationship between the two tables. For the probes from chip Y98, the first entries in MICROARRAY_DATA correspond to the experiment Wt-4-1; and for Y99 the third entries of MICROARRAY_DATA are the expression data from experiment Y99-exp-3.