Construction and realization of the algorithm bootstrap on synchronization database

Huabei Nie

Department of Computer & Information Science,
City College of DongGuan Univercity of Technology,
Dongguan523419, Guangdong, China

Bin Yang*

School of Electronic and Information Engineering,
Shunde Polytechnic, Shunde 528300, Guangdong, China

Yang Zhao

Department of Electronic and Information Technology, Jiangmen Polytechnic,
Jiangmen 529090,
Guangdong, China

**Corresponding author is Bin Yang, email:yangbby@126.com*

Abstract
Based on real time mode, this paper has proposed a new construction and algorithm for design and application of algorithm engine which has better performance than the traditional algorithm. This article draws on the idea of industrial policy configuration software, presents a general algorithm based on real-time database engine design ideas. The algorithm engine has three main features.(1) Good openness. The algorithm engine complies with OPC specification provides an open data access interface; (2) Easy to use. Algorithm engine provides a convenient graphical strategy configuration feature, the capability of online modifying parameters and the capability of strategies debugging features, greatly reducing the workload of the operator; (3) Good system security: allows setting different permissions for different users.
Key words: ALGORITHM ENGINE, REAL TIME DATABASE, CONFIGURATION, REAL TIME DATA DIGGING

Introduction

With the increasing competition in the international market, the use of high technology to transform information, computer, automation and other traditional companies have become a trend. In the process industry, with principle of high-yield, low consumption, low cost, less pollution, and the maximum overall effectiveness, Computer

Integrated Manufacturing System (CIMS) has been increasingly adopted by enterprises [1], and has become a powerful tool modern management, scientific production and increasing market competitiveness.

Real-time database is an important part of the process industry CIMS architecture [2]. Together with relational database, they constitute the CIMS system information integration environment. As a process-oriented data platform, real-time database not only bear the process of data integration tasks, but also directly supports a number of application control layer and process monitoring layer. The success of the implementation of real-time database directly affects the quality of the CIMS system. However, China's process industry also has many deficiencies in real-time database platform/ management and control integration. Integrated management and control system based on the real-time database includes many sub-tasks, such as the cost of computing, fault detection, optimization control. These sub-tasks, in addition to differences on the realization of business functions, you must also complete some common tasks, such as data obtained from the real-time database, the result is written in real-time database, online parameters tuning and debugging strategies, etc. This makes the workload increases. It is difficult to ensure the stability of the software performance optimization and operation, thereby affecting the reliability of the entire application system. On the other hand it makes the development of software does not have the versatility and scalability, maintainability when problems emerge relatively poor.

A general algorithm about engine is proposed based on real-time database. It draws on industrial strategy configuration software design ideas, takes the data structure and design method of algorithm configuration software, solves the above problem, and has been successfully applied in practical work.

The design of algorithm about engine based on real-time data

Real-time database is the database that its data and transactions have explicit timing constraints. Correctness of the system depends not only on the logical result of the transaction, but also on the time the logical result produced [3]. The biggest difference between the real-time database and relational database is a timely, original, non-modified, data volume. Its raw data sampling interval is usually in milliseconds. But the raw data into a real-time database data interval is generally seconds. Real-time database stores a large number of real-time data. These data produce a true record of the working conditions of the system. Therefore, in addition to real-time database to guide the process control, production situation analysis, beyond the fault diagnosis, but also that the production process optimization, and enterprise information management, optimized to provide a reliable basis for decision-making data.

According to the size range of applications, real-time database can be divided into the field level real-time database and a dedicated plant-wide real-time database [4].Plant-wide real-time database is usually developed as a stand-alone system. The world's most famous plant-wide real-time database, there are two main products, one is InfoPlus of AspenPlus company; the other is the PI real-time database of OSI company. Field-level real-time database is usually present as a core part of industrial control configuration software or DCS systems.

Engine algorithms discussed in this article applies to both plant-wide database, also applies to the field level real-time database. In fact, as long as the real-time database can provide data access interface, this is transparent to the algorithm engines.

In order for data mining algorithms unified scheduling and management, you need to define a unified data structure algorithms. If the algorithm is seen as places data processing, and data processing algorithms to complete the following:

$$Y = f(X, C, L) \qquad (1)$$

where, Function f is a function of the algorithm implemented, $X = [\mathrm{x}_1, \mathrm{x}_2, ..., \mathrm{x}_N]^T$ and $Y = [\mathrm{y}_1, \mathrm{y}_2, ..., \mathrm{y}_N]^T$ are respectively the input and output of the algorithm, $C = [\mathrm{c}_1, \mathrm{c}_2, ..., \mathrm{c}_P]^T$ is adjustable parameter of algorithms, $L = [\mathrm{l}_1, \mathrm{l}_2, ..., \mathrm{l}_Q]^T$ is intermediate amount of algorithms. N, M, P and Q are respectively algorithm input, output, the number of adjustable parameters and intermediate amounts. Then an algorithm for data structure should include the following core part [5]:

Algorithm input area: Raw data processing algorithm, i.e., the above formula vector X. It should be noted that each component of X may have different data types (real, integer, boolean, etc.). And these components may also be a function of time, because the data processing algorithms may be time series.

Algorithm output area: Results processing algorithm, i.e., the vector Y of above formula.

Tunable parameters area: Arithmetic processing for a live modified or tuning coefficients, i.e., the vector C of above formula. Engine must have parameters on-line modification.

Algorithms intermediate variable area: Intermediate amounts algorithm generated in the process, i.e. the vector L of above formula. These amounts do not need to be modified online, but typically produces the current time, and at a later time using the algorithm run time.

Algorithm code area: the realization of function f in the above formula. Since these algorithms are executable reusable modules called, Thus to achieve the same algorithm although the same code, but can have different data and processing parameters, i.e., a different algorithm may have the input, and tunable intermediate amount, thus the different output of algorithms.

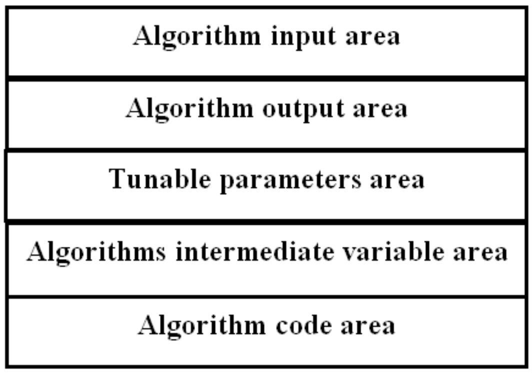

Figure 1. The general form of the algorithm data structure

Further, from the input-output characteristics, the algorithm can be viewed as a black box with *N* inputs and *M* outputs. Figure 2 shows a simple "incremental PID" algorithm diagram. The algorithm is an input and an output algorithm, completing the following operation:

$$y(k) = K_P[e(k) - e(k-1)] + K_i\, e(k) + K_d[e(k) - 2e(k-1) + e(k-2)] \quad (2)$$

where, $e(k)$ is algorithm input, $y(k)$ is algorithm output. K_P, K_i and K_d are tunable parameters of algorithm. $e(k-1)$ and $e(k-2)$ are intermediate parameter of algorithm. when the representation of strategy configuration algorithm, figure 2 is useful.

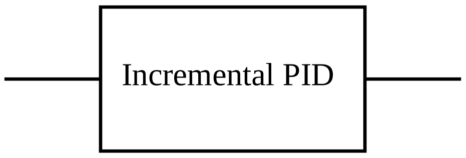

Figure 2. Incremental PID algorithm represents

For the convenience of the algorithm for adding custom algorithm engine also provides a range of API functions as a data access interface. Users can be accessed via the API function algorithm input, output, adjustable parameters and intermediate amount, enabling the user to customize the algorithm.

Database table structure design

Database system needs a lot of writing and deleting operations to files, there must therefore be a reasonable way to control the position of the write operation takes place, otherwise it will produce a large number of file fragments unavailable, thereby reducing the access speed and storage density of the database files. This paper uses a bitmap file allocation method to achieve storage space. Depending on the configuration size setting bitmap number of pages, each bitmap page is 4kB. Therefore, a total of 32k bits, each bit can be allocated 8 bytes (It could not be 8 bytes. It is based on system configuration). If this bit is 0, indicating the location of the 8 bytes corresponding bits representative has not been assigned, otherwise 8 bytes corresponding position has been allocated out. So allocated space of a bitmap page is 256k bytes. When a new space allocation request arrives, we must find a starting bitmap page and the corresponding bit position 1.

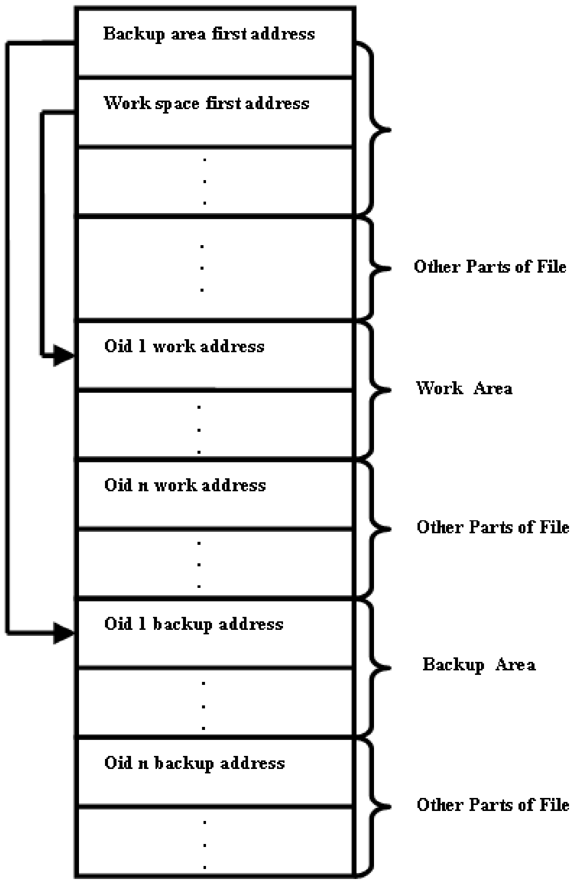

Figure 3. The initial structure of the database table

This paper uses the shadow page method for database real-time fault recovery [6]. For this reason each record, whether recorded or index entries conventional bitmap page record, the system must assign an object identifier (oid). Each record to maintain two versions, one is to restore the backup version reserved for the database fails, the other is reserved for transaction processing work release. As shown in figure 1, the initial header file header information table is a table, the table for each column of the table index information and attribute table names described. Each record header information storage table address, which has two kinds. One is a backup address, pointing to record backup version. One is the work address, point to record a working version.

There are two typical current database physical storage model: row store and column store. Update-intensive online transaction processing applications are suitable for row storage model. Online analytical processing and query-intensive applications are suitable for column storage model. Storage and real-time calculation of real-time text database is responsible for a large number of real-time data, so the use of a physical model row store, that a record of all the attributes of a column address on file is a linear continuous, as shown in Figure 3 the record store.

Of which Size represents the size of the record, Prev is the oid of the previous record on the table, Next is the oid of the next record on the table. The following is the value of each property column: (1) If the size of the property of the column is fixed (integer floating point variables, etc.), property stored in the column is the actual value; (2) If the size of the property of the column is not fixed (string array type, etc.), attribute columns stored items from offs and size composition, The actual value stored in the space of (the first address of record + offs, the first address of record + offs + size). Use this model because: (1) insert or delete a record equivalent to insert or delete records in a doubly linked list in the table, so to establish a two-way linked list of the logical structure of the database table; (2) Each record is assigned a unique object identifier (oid). According to oid informs work address or backup address of record, combined with record head size, a complete record can be read.

Transaction processing and the shadow page method

When the transaction begins, the database table is in a consistent state. Any record is only one version that the same address backup and work address. So back up the contents of an object identifier and work area, as well as the number of object identifiers are the same. When the transaction modifies non-bitmap page records, determine whether the address work records is set to modify bits. If there is modifying operation, delete working version the work address referred to. If not, then there is no extra work release, do not delete it. After treatment, the allocation of new storage space for records. The new space will be assigned to work as a recording start address and modify bit is set, the address remains the same back-up records and records referred maintain.

Any object bitmap page by writing 1 or cleared to allocate or free up space, this is bound to modify bitmap page. When modify a bitmap page records, work records to determine whether the address is set to modify bits. If set to modify bit, show a bitmap page has shadow page, without having to copy a shadow of its pages. Otherwise the bitmap page requires a shadow page, that record is assigned a bitmap page shadow page. Allocation shadow page may cause a series of copy operations of bitmap page. However, due to the principle of locality allocated, time similar dispensing operation always occurs on a continuous page bitmap, the bitmap page copy operation does not generate excessive shadow pages.

When the transaction operation to insert a record, you must assign an object identifier for the record. If freeoid single list is empty, the need to expand the work area to store more freedom oid. After the first expansion of the work area of the database table structure is shown in Figure 4.

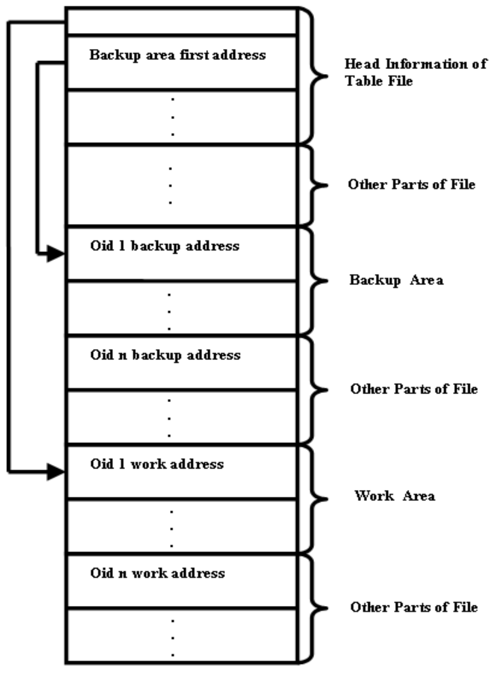

Figure 4. The database table structure diagram after the expansion of workspace

When the transaction delete records, records and backup address backup versions remain unchanged. Therefore, when the transaction delete records, and there is no real delete records backup version, it will only set record work address invalid. But after the record is deleted, need to be recovered object identifier of deleted records in the workspace. freeoid single chain store just recovered identifier. When oid recovered, it again assigned to another record at some point. Then backup versions and versions of the same work oid record may not be relevant, but it does not affect the correctness of the program. Because: (1) the transaction is successfully submitted, the oid leaving only a working version of its backup versions are deleted. Because the original purpose of the transaction is to delete the original record. (2) When the transaction commits fail, and fall back to a backup version, the record also keep only one backup version, delete records equal to the transaction operation is withdrawn.

Transaction Commit and Table's Malfunction Recovery

After the end of the transaction, need to submit the modify it made to the table. As a result of the shadow page method [7], delete, and modify records do not delete the backup version of the operating record, it must be eliminated before the transaction commits backup version. Workspace object identifier number is greater than or equal to the backup area only object identifier number two cases, each object identifier must have a working backup area address. As shown in Figure 6, the backup address and work address equal, indicating that the transaction process does not allocate additional space, without deleting the backup version. Backup address is invalid show before the transaction began, recording no backup versions, there is no need to delete. Backup address is valid, indicating that a backup version must be removed. After the above treatment process, the space occupied by the backup version is recovered. When space reclamation, ensuring the same backup and work area size, and the same address and work address of each backup object identifiers to commit the transaction is completed. Now the database tables and restore consistency to a new state. After the mutual exchange backup role and work area, you can start a new transaction.

When a transaction failure or accident database and restart, the system needs to restore the database tables. Recovery equal undoing changes made to the database transaction. Ideological shadow page method is a backup version remains the same, always modify the working version. So back to work just the start address of a backup copy of the contents of the area back to the work area. Then the contents of work area and back up area are exact same. Compared to log-based recovery techniques failure [8], the shadow page method eliminates the overhead of logging and processing, recovery speed, suitable for higher satellite ground equipment monitoring real-time requirements of such applications.

Algorithm engine test

A simple arithmetic algorithm can test the transaction engine. An accumulation two operations, the two constant parameters initially set to 1, the addition and subtraction are set for computation, three input ports of the weights is set to 1. Each algorithm module contains some basic information, including the input value, the output value, the algorithm name, algorithm parameters, algorithms entry function name, algorithm ID, algorithmic scanning cycle. Runtime engine algorithm through an algorithm for each scan, the entire strategy document will be scanned once, and then call the appropriate algorithm library files, obtain final output policy. The output is displayed on the LAN monitoring software, real-time transmission shown in Figure 5.

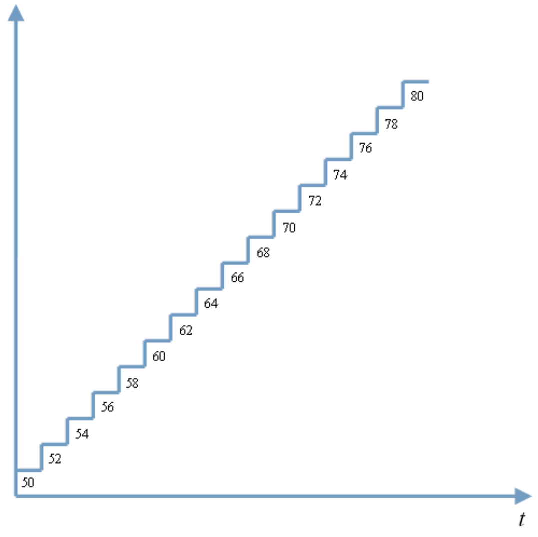

Figure 5. Output curve of the practical operation example (Y is Real Value)

By monitoring software, you can view real-time algorithm for each module's output and modify the parameters of each algorithm module. The third right input port for adding or subtracting the value is set to -1, the resulting increase running curve 2 shown in Figure 6.

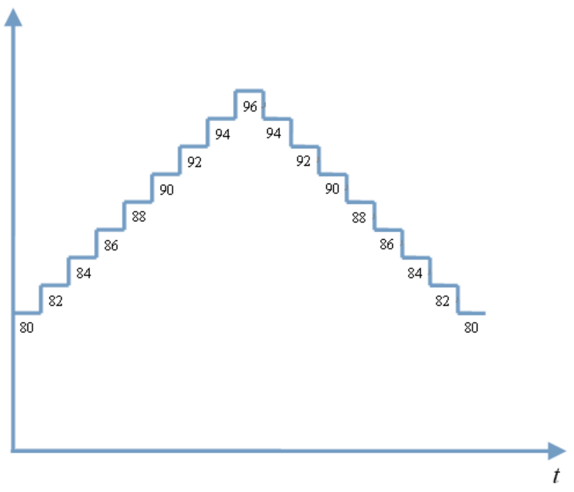

Figure 6. Operation curve after parameters modified (Y is Real Value)

The above example demonstrates that strategy configuration software to run the feasibility of the proposed method in embedded systems. For user-defined algorithm, after compiling generate the corresponding link library file that can be added to the configuration algorithm library. On the basis of interpretation based on the operation principle, through the LAN will be downloaded to the user library functions embedded systems, in order to invoke the algorithm engine load. The design of the core code was written in C, to ensure that the design of portability, which is independent of the operating system. On the hardware side, the use of dynamically loaded dynamically call I/O drivers and other modules, support for hardware extensions.

Conclusion

This article draws on the idea of industrial policy configuration software, presents a general algorithm based on real-time database engine design ideas. The main features of the algorithm engine are:

(1)The algorithm engine to comply with OPC specification provides an open data access interface; in compliance with the algorithm based on the specification, also allows users to add custom algorithms;

(2)Algorithm engine provides a convenient graphical strategy configuration features, online capabilities and strategies to modify the parameters debugging features, greatly reducing the workload of the operator;

(3) allows you to set different permissions for different users can only use the operator operating functions required to produce only an administrator can control parameters, the system parameters are set, an exit program operations and other functions.

The algorithm engine can be further improved functions are:

(1)In the policy configurator, a plurality of simple algorithm configured as a complex algorithm, and the algorithm is added to the library;

(2)In strategy runner optimization of algorithm executed order. We use right directed acyclic graph to decide the order of execution. But when the policy loop occurs, usually determines the execution order can only be based on the order of registration algorithm. This approach is also to be further improved.

References

1. Srankovic, J. A., Sang Hyuk Son and Hansson, J. (2009) Misconceptions about real-time databases. *Computer Volume*, 35(7), p.p.29-36.

2. Pattle, R. and Ramisch, J. OPC (2007) The defacto standard for real time communication parallel and distributed real-time systems. *Proc. Conf. on the Joint Workshop*, p.p.744-752..

3. Ming Syan Chen, Jiawei Han and Yu, P. S (2006) Data mining: an overview from a database perspective. *Knowledge and Data Engineering*, 39(6), p.p.76-82.

4. Jim Gy Andreus R. (1993) Transaction Processing: concepts and techniques. *Transactions Processing: Concepts and Techniques*, 14(3), p.p.724-732.

5. Kao B, Garcia-Molina H. (1995) An Overview of Real-Time Database Systems. *Upper Saddle River*, NJ, USA: Prentice-Hall, Inc., p.p.122-134.

6. Krol V., Pokorny J. (2006) Design of V4DB-Experimental real-time database system. *Proc. Conf. on Industrial Electronics*, p.p.126-131.

7. Xuetao Wu, Hongxing Li (2009) Design of Client Program Based on OPC for Remote Monitoring System, *Computational Intelligence and Design*, 15(3), p.p. 420-423.

8. Xing Jing, Jing Zhang, Yang Zhao (2015) An efficient complex event processing system having the ability of parallel processing and multievent pattern sharing. *Journal of Intelligent and Fuzzy Systems*, 28(2), p.p. 162-174.