Dynamic Obstacle Avoidance Algorithm Based on DYNAMIC Environment MR

Juan Guo1,*

1School of Software Engineering, Shandong Polytechnic College, Shandong, China

*Corresponding author e-mail: guojuan@sdpu.edu.cn

Abstract. With the rapid development of computer technology, the importance of database systems as an indispensable part of information systems is becoming more and more prominent. And nowadays, the society has been increasingly using modern means to program databases. Database is a large and complex, huge amount of data and has a certain structure and independence of the important system, its programming requires certain technical means, the author will be in the text for the database programming involved in the key technology to explain.

Keywords: Computer Software Engineering, Database, Programming Technology

1. Introduction

Software engineering is a very comprehensive discipline, which involves computer technology, network communication and other aspects, and it covers a variety of fields. With the rapid development of computer technology, software engineering profession in the domestic market share is increasingly high, it has become the most important and most active in the field of economic construction in China, so the use of software engineering-related knowledge of database programming is the most appropriate choice, the database programming is to create, read and write a database and other operations, involving the design, management and optimization of the database.

2. Basic concept of the database

Database is a "warehouse" that stores, manages and uses query data in a specific way, with the advantages of multi-user sharing (by setting users and corresponding permissions), convenient operation, stability and efficiency. Common database models can be divided into: relational database and non-relational database, at present, in most traditional industries and small and medium-sized enterprises, relational database is still the main. Relational database: that is, the form of two-dimensional tables, the complex data structure into a relatively simple binary relationship between multiple two-dimensional tables through foreign keys to establish inter-table relationships (similar to multiple sheet pages in Excel, each other through VLOOKUP correlation matching) SQL language: short for Structured Query Language, is a more widely used relational database query language, usually used to access, query, update and manage the database (is one of the basic skills that data workers need to master). Database technology has evolved as Figure 1 shows.
3. Technical method to access the database
Data access or data access is the application's access to a database (mainly relational databases) or other persistent data. Data access is the core operation of information system. The information system receives user input through the user interface, processes it in the business logic, and then stores it in the database; it retrieves user query information from the database, processes it in the business logic, and then presents it in the user interface. The data access method refers to the programming interface and related software environment for accessing the database in general. Applications can use different languages (suppose there are m), and databases can come from different vendors in different versions (suppose there are n), and if methods to access each database are built in for each language, m*n methods will be needed; if a common interface (interface) is created between the language and the database, providing a method for each language to convert requests and data into the form specified by this interface. If a common interface (interface) is established between the language and the database, providing a method for each language to convert requests and data into the form specified by this interface, and a method for each database to convert requests and data from this interface into the form supported by the database, only m+n methods are required. As shown in Figure 1, most existing data access methods fall into the latter category, providing access to one or more databases in one or more languages in the form of an interface. The data access methods are shown in Figure 2 below.
3.1. Embedded SQL
SQL is a structured language and most statements are executed independent of the statements before and after them, so it is not procedural, while most high-level programming languages are procedural based on loops, conditions and other structures. In order to provide applications with the ability to access data, Embedded SQL was created. Embedded SQL (ESQL) is a method of writing SQL statements directly into the source code of programming languages such as C, COBOL, FORTRAN, Ada, etc. The intersection of SQL and high-level languages is embedded ESQL, and the existence of ESQL gives applications the ability to access data and process data [1]. In this approach, the language in which the SQL statements are embedded in the target source code is called the host language, or simply the master language. For example, a programming language such as Pro*c provided by Oracle Corporation, which embeds SQL statements into C, provides a good way to access the database and is widely used in the banking and telecommunication industries. Embedded SQL provides the following three main types of query operations [2].

1. A single tuple can be retrieved with embedded SQL.
2. Multiple tuples can be retrieved using cursors in embedded SQL.
3. Queries can be specified at runtime using dynamic SQL.

In the Java programming language, a technique for embedding SQL commands in Java is provided SQLJ [3]. SQLJ converters are able to convert SQL statements into Java statements, which can then be executed through the JDBC interface. Therefore, when using SQLJ, a JDBC driver needs to be installed. In SQLJ, multiple tuples can be retrieved using iterators. Two kinds of iterators are provided: named iterators, which are associated with a query structure by listing the names and types of attributes that appear in the query structure, and the attribute names must correspond to the declared Java program variables, respectively; and positional iterators, which list only the types of attributes that appear in the query structure. For both of these iterators, the order of the list should be consistent with the order in which the attributes are listed in the SELECT clause of the query. An analysis of the commonly used ESQL reveals some basic features of ESQL as follows [4].

1. The embedded SQL statements appear in the middle of the host language statements. The SQL statements are not case-sensitive and it is common practice to follow the form of the host language for input.
2. Each SQL statement starts with an introduction marker to indicate that it is a SQL statement. The big number is marked by "EXEC SQL".
3. If the embedded SQL statement is extended to multiple lines, the statement continuity policy of the host language is used.
4. Each embedded SQL statement is marked with a terminator to indicate the end of the statement. The terminator varies according to the host language.

3.2. Database private query language
Database specific query language, is a query language that can run on the server side without interacting with the client, and the typical specific query languages are PL / SQL (Procedural Language) [5]. Transaction-SQL [6]. PL / SQL, the procedural SQL language, is an extension of ORACLE to SQL, adding programming language features to SQL, organizing data operations and queries in procedural units of PL / SQL code, and realizing complex functions and calculations through logical judgments, loops, etc. It consists of four main modules: declaration, start of execution body, exception handling, and end of execution body. In PL / SQL, only the data manipulation language can be used and only the data definition language can be used in a dynamic way. Transaction-SQL is an ANSISQL implementation on Microsoft SQL Server and Sybase SQL Server, which has similar characteristics to PL-SQL.

The two database query languages mentioned above are representatives of widely used languages with their own specific advantages, such as improved application performance when running in a client / server environment. For client / server applications, the real performance bottleneck is the network. No matter how fast the network is, as long as a large amount of data is exchanged between
the client and the server, the efficiency of the application operation will be affected. By using a
database specific query language, the server is responsible for the processing of large amounts of data,
so the network transfer time of the data is reduced accordingly.

3.3. Use the call-layer interface
With the development of the Internet, the traditional way of data and application binding is not
suitable for the network environment, and separating the data from the application and connecting
them through an interface to achieve data exchange is the call layer interface [7]. The process of using
the call layer interface for database access is to use the call layer interface as an adapter that provides a
conversion role between the application and the database, realizing the decoupling of the application
from the database and improving the relative independence of the data. Currently there are two main
schools of thought about the call layer interface: the Microsoft series, and the ORACLE (SUN) series.

4. Query and optimization technology of the database
Since the entity data model differs from the relational data model in allowing the existence of
uncertain attribute values, some unique definitions are derived for the query optimization problem, and
I will give the precise definitions for each of these unique features below [8].

4.1. The maximum similar tuple of the table
Based on the consideration of the characteristics of the entity data model, because in the actual query
execution process of the entity-based inferior data management system, a similarity is recorded for
each tuple, and this similarity indicates the current matching degree of this tuple and the user query
statement, and for a single tuple, the corresponding similarity is not increased during another query
execution, and the similarity of all table tuples in the query The initial state of all table tuples is 1.0 at
the beginning of query execution, and the similarity value of tuples after query operation is calculated
by judging the similarity degree of the selected tuple attribute values during the query execution [9].
ed (Los Angles, LA) = 8 > 3, and the similarity calculation result after multiplying the uncertainty
attribute value cleanliness is Sim = 0.67*1 + 0.33*0 = 0.67. Then, the similarity of the tuples in Table
R is calculated after the similarity selection operation with the constraint City = 'Los Angles'. then the
similarity of the tuples in table R becomes 0.67 after the similarity selection operation with constraint
City = 'Los Angles'.

Definition 1: The maximum tuple similarity of a tuple r in table R can be expressed as:

\[
\text{Max sim}(R) = \{\text{sim}(r) | r \in R, \forall r' \in R, \text{sim}(r) \geq \text{sim}(r')\}
\]  

(1)

The above definition gives the definition of the maximum tuple similarity of a table, which is
important in the subsequent estimation of the cost of query operations.

4.2. Data fetch
Definition 2: Single table data extraction, for a given similarity threshold \( \varepsilon \), tuple r in table R, sim (r)
denotes the similarity of r. The extracted set of results can be expressed as:

\[
\xi(R) = \{r | r \in R, \text{sim}(r) \geq \varepsilon \}
\]  

(2)

When doing the threshold similarity join operation, the basis of whether the joined tuples of two
tables appear in the join result is not only based on the similarity judgment of the two tuples in the join
attribute, but also related to the similarity of the two tuples, taking table R and table S as an example,
if the similarity of a tuple in table R is less than the similarity threshold and the similarity of the largest
tuple in table S, then it is impossible for this tuple to meet the final result If the similarity of a tuple in
the R table is less than the similarity threshold and the maximum tuple similarity in the S table, then
this tuple cannot meet the similarity requirement of the final result, so there is no need for it to appear
in the join result, and it should be filtered out when making the join similarity judgment. Assuming \( \varepsilon = 0.5 \), the tuples with similarity 0.5 in the R table do not need to participate in the join operation because
they are no longer likely to appear in the final result, which leads to the definition of join result data extraction in Definition 2-3.

Definition 3: Connection result data extraction, for a given similarity threshold $\varepsilon$, a tuple $r$ in table $R$ and tuple $s$ in table $S$, $\text{sim}(r)$ and $\text{sim}(s)$ denote the similarity of $r$ and $s$ respectively, $\text{max sim}(r)$ and $\text{max sim}(s)$ denote the maximum tuple similarity in table $R$ and $S$ respectively, and the set of connection results in table $R$ and $S$ after extraction can be expressed as:

$$
\xi(R \bowtie S) = \{(r, s) | r \in R, s \in S, \text{sim}(r) \geq \varepsilon / \text{max sim}(S), \text{sim}(s) \geq \varepsilon / \text{max sim}(R)\}
$$

R and S are similar in connection attributes.

4.3. Optimizing strategy

After the logical query plan, each possible physical plan is given an estimated cost, and our aim is to select the physical query plan with the minimum cost estimate. For searching the space of possible physical query plans there are two main strategies: top-down and bottom-up. Top-down: This starts from the root of the logical query plan tree and searches downward, calculating each possible method for each physical implementation of the operation, calculating the cost of the combination and then taking the best one as the final physical query plan. Bottom-up: For each child node of the logical query plan tree, the physical implementation of the possible cost of each child node is calculated and recorded as the basis for the calculation of the upper nodes [10]. There is no big difference between the two strategies, but in the bottom-up strategy we only consider the best plan for each sub-expression when calculating the plan for larger sub-expressions, which is the optimization strategy for dynamic programming. In practical optimization work, we usually have to simplify the search space of physical query plan by some heuristic optimization rules, and we also give several common heuristic optimization rules here:

1. If projection and selection operations are involved on table $R$ in the logical query plan tree, projection and selection are pushed down to the bottom of the tree as much as possible, i.e., let the projection and selection operations finish first.

2. If there are indexed attributes on the selection and join attributes, try to process the indexed attributes first.

3. If one of the two tables joined has an index on the join attribute, place the table in the inner loop of the nested loop join.

4. If two tables are joined with the same parameters on the join attribute, then the table with the smaller number of tuples is placed in the inner loop.

The COST and RESULT parameters in the query operation cost estimation model are related to the size of the table in the estimation computation simulating the actual query execution, so minimizing the amount of data involved in the computation during the query execution can almost certainly optimize the query, based on the consideration that, as in the case of optimization on relational databases, prioritizing the execution of the selection on all tables before they are joined can be determined it must be possible to reduce this cost, and the rest of the work is the selection of the order of table joins, and we start an optimization study on the selection of the order of this table join. The dynamic programming algorithm is generally used to select the join order for relational database query optimization because the join order selection based on this cost model satisfies properties such as optimal sub-structure under the assumption that the cost estimation model is unbiased, so the dynamic programming algorithm can find the optimal query plan, which has been proved in a lot of previous research work. However, the dynamic programming algorithm has poor time efficiency and is affected by the number of join operations and the amount of data. Although the greedy selection strategy does not guarantee to find the optimal query plan, its excellent execution time efficiency can be applied in some specific cases, such as the worst query time can be predicted to be small before query execution. Therefore, we use two optimization strategies, greedy selection and dynamic programming, to accomplish the work of query optimization on an entity-based inferior data management system.
5. Conclusion
In summary, the author explores the key techniques of database programming with the knowledge of software engineering. Among them, data access technology and database query and optimization are one of the frontier research hotspots in the field of computer, and are also important technologies for database programming. The main purpose of these two is to provide better, more complete and more reliable information services to users. Database programming can extract useful information from a large amount of fuzzy and random information through the knowledge of software engineering and convert it into what users need or are interested in, and make these records with new features (i.e., diversity) and high efficiency, which is of great practical importance.

References
[1] Li Yanqing, Yang Gui, Chen Hongsheng, etc. Research of Dynamic SQL Technology in Embedded SQL [J]. Computer Science, 2006, (5).
[2] Elmasri R, Navathe S B. Database system base: Primary article [M]. 2007, 199-204.
[3] Melton J, Eisenberg A. Understanding SQL and Java together: a guide to SQLJ, JDBC, and related technologies [M]. Morgan Kaufmann, 2000.
[4] Pan L, Yao T, Yao C, et al. The Design and Realization of Human Body Database Management System Based on Non-Contact 3D Body Measurement [J]. Advanced Materials Research, 2014, 1048: 545-549.
[5] Xu Wanjin. Design and Realization of Administrative Boundary Management System Based on ArcGIS [J]. Surveying and Mapping, 2011.
[6] Yang R. Exploration of Advanced Application Development Technology Based on LabWindows / CVI [J]. China Computer & Communication, 2019.
[7] Whang S, Menestrina D, Koutrika G, et al. Entity resolution with iterative blocking // Proceedings of the 35th SIGMOD International Conference on Management of Data [J]. New York, USA, 2009: 219-232.
[8] Benjelloun O, Garcia-Molina H, Menestrina D, Whang S E, Su Q, Widom J, Swoosh. A generic approach to entity resolution [J]. The International Journal on Very Large Data Bases, 2008, 18 (1): 255-276.
[9] Hernandez, M. A., Stolfo, S. J. Real-World data is dirty: data cleansing and the merge / purge problem [J]. Data Mining and Knowledge Discovery, 1998, 2 (1): 9-37.
[10] Monge A E. Matching Algorithms within a Duplicate Detection System [J]. IEEE trans data eng, 2000.