INFORMATION
SYSTEMS IN
MANAGEMENT

Information Systems in Management (2016) Vol. 5 (2) 294–304

SELECTION OF WORKING DATABASE FOR THE GENETIC ALGORITHM PROCESSING DATA OF EXCHANGE QUOTATIONS

GRZEGORZ WOJARNIK

Institute of Computer Science in Management
Faculty of Economics and Management, University of Szczecin

The aim of this article is to carry out a comparative analysis of the performance of databases: SQLite, MS SQL Server 2014, Firebird 2.5 Server and Firebird 2.5 Embedded with the use of the object-relational mapping library ServiceStack.Ormlite and IDBCommand interface in Visual Studio with the use of C# programming language within the framework of .Net Framework 4.5 platform. The selected database will serve in the future as a data store for the operation of a genetic algorithm, which role will be processing of stock market data. Test data is daily data of stock quotations of the stock exchange downloaded from bossa.pl on 6.06.2015.

Keywords: databases, genetic algorithm, database performance, stock exchange

1. Introduction

The issue concerning the productivity of software, which task is to analyze data that requires many repetitive operations (SELECT, UPDATE, INSERT) is extremely important, when its task is to process data collection, in which considerable amount of records was saved. Due to the fact that databases allow the easy management, connection and selection of data for its storage, we should pay special attention to their performance during the above-mentioned operations.

Methods related to the frequent performance of repetitive data undoubtedly include genetic algorithms. Their operation assumes the reiteration of action for socalled genetic operators. Simultaneously, if a genetic algorithm operates on such a large data set like, for example, data concerning stocks quoted on the stock exchange, we must also take into account the performance of such a database in the context of operations on a large amount of data. The article will describe the performance test of selected databases with the aim of using them as the data storage for the genetic algorithm data in the stock exchange. An example will be data about all daily quotations of stocks quoted on the Warsaw Stock Exchange.

The principle of the algorithm and the use of test data are described in this study [9]. It can be assumed that in case of the algorithm that processes stock market data, an fitness function will be based on the analysis of historical data containing values of technical analysis for stock market individual stocks. Due to the fact that the collection of this data is a set of significant volume, a key becomes a mechanism for reading the data in order to calculate the value of the fitness function on statistics that show how did the formation of stock’s prices look like in previous periods after the occurrence of signals, which are a combination of technical analysis indicators. These combinations will be generated during the operation of the genetic algorithm.

It should also be noted that the choice of a base in the context of genetic algorithms is significant in the case when historical data of significant volume will be used to calculate a value of the fitness function. The speed of this algorithm will be determined by the speed of the fitness function, which must be actuated for each generated solution. However, if an fitness function in its action is based on a calculation of the statistics on the basis of a large set of historical data, a key becomes the selection of such a database, in which data will be stored and shared so as to provide the fastest time of this process.

2. Assumptions and conditions of the research problem

2.1. Technical analysis of stock market indicators

Technical analysis is one of the methods to identify possible or probable trends in the behavior of rates for stocks. One definition of the technical analysis says that it is “the use of regularities in the formation of share prices to predict changes in price trends before these trends will be reversed” [6]. To this definition, we should add that in addition to anticipate changes in trends, it is also important to predict the time horizon for a continuation of already observed trends.

Technical analysis is based on the assumption saying that there are regularities in price movements of financial instruments. However, it should be noted that there is no single universal method that fully describes these regularities. This is connected with a very large number of factors affecting the game of supply and demand observed in the financial markets. The stock exchange is included to these markets. Therefore, technical analysis focuses on the study of variability of different indicators derived on the basis of basic parameters of stock market quotations. These quotations include the opening price, closing prices, maximum price, minimum price and the value of trading in a specific period of time. The sheer number of technical analysis indicators, which are widely described in the literature of the subject, oscillates around thousands of points. It presents the complexity of a matter for analysis of rules governing the formation of a listed companies' rate.

2.2. Genetic algorithms

Genetic algorithms are the field of artificial intelligence, which was taken assumptions drawn from the theory of evolution as a paradigm. It was observed that the evolution is based on a variation in the population of individuals, which ensures a variety of characteristics for a set of individuals. This, in turn, enables (saying in the greatest generalization) the survival and transfer of feature for these individuals that are best suited to the environment surrounding them. The feature of adaptation determined the extent, in which particular individuals are able to survive in the environment surrounding them [7, 8]. It can be assumed that genetic algorithms should be used wherever there is a well-known role that gives a chance to solve the problem in detail. Referring to Łęski, it can be assumed to algorithms or evolutionary methods include such methods that mimic the process of natural evolution, which involves making changes in populations of living organisms, which direction is the adaptation in order to improve the chances of survival and reproduction of organisms [5].

The most popular types of evolutionary methods are genetic algorithms. The forerunner of this approach is J. H. Holland, who published in 1962 the work under the title "Outline for a logical theory of adaptive systems". Holland presented in this work the bases for adaptive systems, which can vary in response to the environment, in which they function [1]. Thanks to this J. H. Holland's research work, it can be seen that the operation of genetic algorithms is largely based on a process controlled by the fitness function that largely uses the probability calculus. As a result, solutions more and more consistent with the stated objective are generated.

Operation of the genetic algorithm can be presented by the following steps [2]:

1. Initialization of the population
2. Calculation of the value for fitness function of each individual in a population
3. Reproduction of selected individuals in order to create a new population
4. Performance of crossing and mutation in the new population
5. Realization of the step 2, unless there will be a conditions of ending the processing.

It can be assumed that in the genetic algorithm each representative of the population is a presentation of one solution in the tested problem. The quality of such a solution is determined on the basis of its matching to the criteria that are taken into account during the assessment, which is a function of the assessment of a certain individual to a given problem. Therefore, it refers to the value represented by a chromosome (of the individual) when calculating the adaptation [4]. Speed of the fitness function is crucial for the functioning of the whole algorithm, because due to the fact that it must be done for each individual constituting the solution of the problem, it is actually the most frequently performed operation, which additionally requires an access to substantial data subsets necessary to set up the procedure of its calculation in many applications.

In one course of the genetic algorithm, a new population constituting a set of individuals is generated. This population includes individuals that are best suited to create conditions, which are represented and determined by the fitness function. Assurance of variation elements is realized at the stage of genetic operations, which through the use of genetic operators such as crossing or mutations make changes to the genotype of individuals, so they are changes that affect the particular characteristics of an individual or individuals selected for changes.

Working principle for the classical genetic algorithm is the simplest approach to genetic algorithms. Its operation can be controlled by a number of parameters describing the functioning of the algorithm, as well as parameters of individual genetic operators and boundary conditions of generated solutions. The generation of initial population in this algorithm randomly creates an initial set of individuals that will undergo further changes in order to achieve a solution, which will be as close as possible to the optimal solution. The crossing operator allows random matching of individuals in order to generate descendants, whose genes will be mixed from genomes (i.e. genetic materials). By contrast, the mutation operator is used to introduce random changes on a smaller scale, because it randomly inserts small changes in an individual’s genome.

Why, despite the fact that that GA does not guarantee the emergency of the optimal solution, do generated results usually turn out to be useful in solving a given problem? First of all, classic and individualized approaches normally will not be possible to carry out, and GA paradigms will be useful in many different situations. Another reason is that the real power of GA paradigms shows that they are generally quite durable. The durability means that the algorithm can be used to solve many problems and even many types of problems and it requires a minimal amount of specific corrections considering the specific characteristics of a particular problem. Typically, the evolutionary algorithm requires the determination of the length of vectors for solving the problem, details of their coding and fitness function – the rest of the program implementing this algorithm does not need to be changed.

Therefore, it can be assumed that if a solution or solutions (generated through the genetic algorithm) are good enough, they are provided quickly enough and they are cheap enough – they are appropriate. Almost all applications of the real world search sufficient solutions and they are usually considered as satisfactory solutions. Of course, it should be noted that the "good enough" solution means that the solution meets the requirements of the specification.

A. M. Kwiatkowski writes that an objective of the genetic algorithm can also be the acquisition of knowledge about the real, new, undiscovered and yet unknown laws and rules that are relevant for the project and govern the studied phenomena [3]. The only premises of examined processed available to researchers are often observations and their awareness of the regularities that stand behind these observations. Therefore, genetic algorithms (due to their characteristics) enable to reach and discover these rules. Genetic algorithms (or more broadly – evolutionary methods) are widely used in the analysis of stock market data.

2.3. Hypothesis and purpose of the study

The adopted research hypothesis says that Sqlite database is an appropriate to use as a working database in the processing of market data for the purpose of genetic algorithm.

The premise to consider the Sqlite database is its popularity resulting from its use is systems such as: web browsers (Google Chrome, Opera, Safari), operating systems (Blackberry, Android, NetBSD, OpenBSD, Solaris, Windows and others). At the same time, we should pay attention to a very small size of the library that enables its operation (approx. MB). In combination with the support of the SQL language, it suggests that it will be the right solution for storage and access to the data needed for operation of genetic algorithms.

Whereas the aim of this study was to perform a comparative analysis of performance for Sqlite database in combination with MS SQL Server 2014, Firebird 2.5 and Firebird Embedded 2.5 with the use of object-relational mapping library ServiceStack.Ormlite and IDBCommand class in Visual Studio using C# language within the framework of .NET Framework 4.5 platform.

3. Methodology of the experiment

3.1. Testing and software environment

The experiment was carried out by writing a computer program in MS Visual Studio 2015 with the use of C# language for .Net Framework 4.5 platform. ServiceStack.Ormlite[1] was selected as a framework ORM (object relational

[1] https://github.com/ServiceStack/ServiceStack.OrmLite

mapping). It is one of the most popular object-relational mapping tools operating in line with the POCO principle, which assumes that one business class corresponds to a single table in the data structure [11].

The aim of POCO approach is to create a ORM mechanism that will easily reconstruct the business logic on the physical structures of database unlike all complex solutions introduced in such ORM libraries like Entity Framework from Microsoft. Additionally, one of the performed tasks was made without Ormlite, but with the use of a standard interface IDBCommand that enables the performance of SQL language queries directly to the database.

The computer Intel NUC 5i5RYH was used for these tests. It was equipped with the following components:

- RAM memory: 8 GB RAM
- Hard disk: SSD 240GB
- Processor: Intel Core i5-5250U
- Operating system: Windows 8.1.

3.2. Selection of databases

The following databases were selected for testing:

SQLite: a library written in C language that implements mechanisms for managing the database. It is widely used as a database of a handy popular consumer software (e.g. Mozlilla Firefox, Adobe AIR) and systemic software (e.g. Android, Mac OS X). This database is available under public domain license.

MS SQL Server 2014: a database management system working in client-server architecture. The producer of this system is Microsoft. It is one of the most popular databases on the market [11]. MS SQL Server is a database available on a commercial basis. This system is available in free version, but with some limitations, which include mainly the maximum capacity - 10 GB of disk space.

Firebird 2.5: an open-source server for relational database management derived from the commercial database Interbase. This database was selected for the study because of two available versions: server and embedded. It allows a simple scaling of the solution from single-user applications to multi-user network applications.

3.3. Types of examined operations in database

Data processing within the framework of action in the genetic algorithm, for which large data sets are training data, requires the performance of various operations in these databases: these operations include repeatedly performed SELECT operations needed for the calculation of the fitness function for specific (both small and large sets) records in the training base, as well as update operations for already entered data (UPDATE) and INSERT operations required when saving the generated solutions during operation of this algorithm. Because of the above mentioned conditions within the framework of the testing procedure, the following operations were carried out:

1. Data loading (database filling) – download of approx. 1 800 000 records for quotations of all stocks quoted of the Warsaw Stock Exchange saved in CSV files downloaded from the brokerage office bossa.pl
2. Data loading (database filling) – IDBCommand – the same operation as in point 1 with the exception that it is carried out via INSERT command made through IDBCommand interface, not ServiceStack.Ormlite library
3. Update of all data – change in the value of all quotations
4. Readout of all data – download of all quotations
5. Partially readout of quotations for all stocks – download of quotations in parts for all stocks separately
6. Random readout of 10 000 single quotations – random download of data for randomly selected quotations.

3.4. Testing procedure

The purpose of the testing procedure was to eliminate interferences caused by the operating system. Therefore, each operation for each database was performed in the following sequence of steps:

- Computer restart
- 5-times start of a certain operation for a given database
- Performance of a measurement, which was the result. The number of seconds is the average time for activation of each operation.

3. Results of the experiment

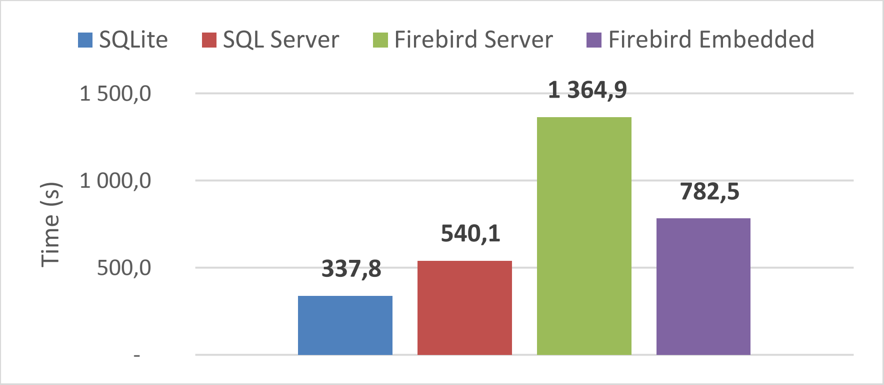

Figure 1. Data loading

As shown in Fig. 1, the shortest time for the realization of data loading test was demonstrated by SQLite database, but the time achieved by SQL Server database was not much worse. Firebird databases were significantly slower – this operation lasted especially long for Firebird Server database.

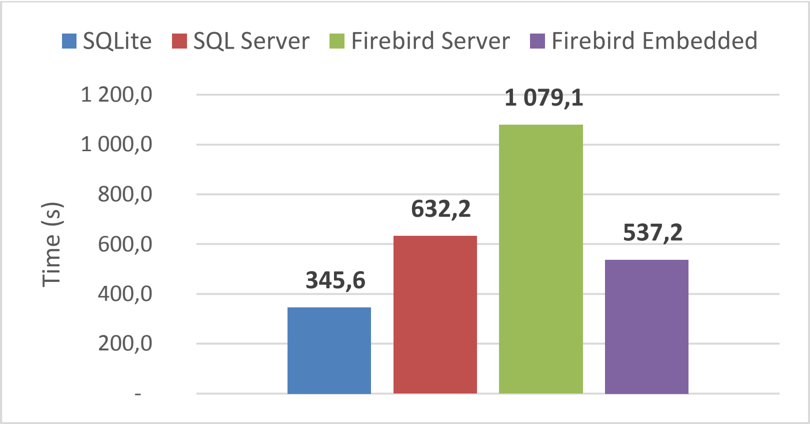

Figure 2. Data loading by using IDBCommand interface

In comparison with the first operation, there are minimal changes in the data loading operation time via IDBCommand interface (Fig. 2) – a significant increase was recorded for Firebird database in versions Server and Embedded, while this operation was slightly slower for databases SQLite and SQL Server. It proves the high optimization of action for ServiceStack.Ormlite library.

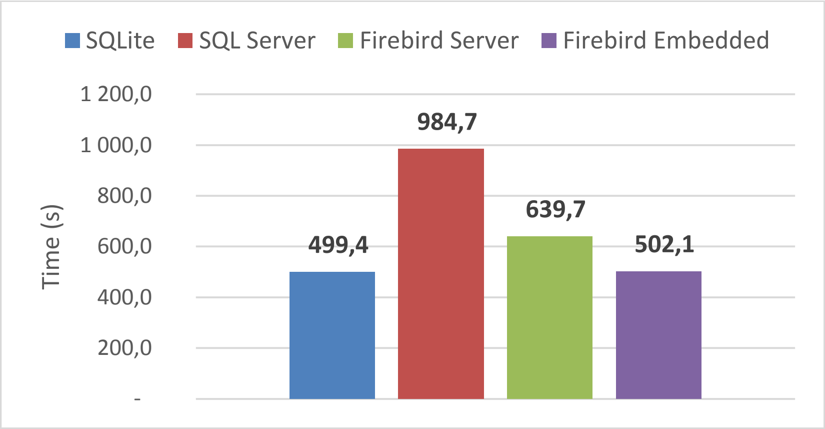

Figure 3. Update of all data of quotations

The results of update operations for all quotations in a single pass (Fig. 3) are different from previous operations. Although the shortest time is presented by SQLite, the slightly worse result was calculated for Firebird Embedded database. The slowest result was obtained by SQL Server database.

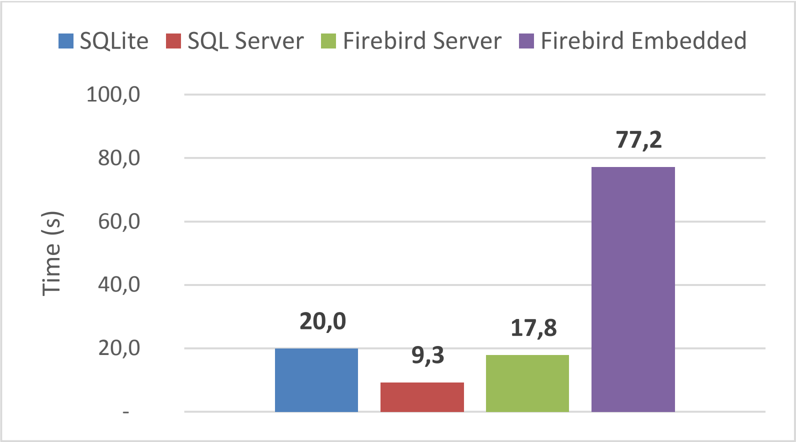

Figure 4. Readout operation for all quotations

In “competition” of readout for all quotations (Fig. 4), definitely the best result was achieved by SQL Server database. Firebird Server database and SQLite database were worse than the above-mentioned database. Definitely the slowest readout was presented by Firebird Embedded. Attention should be drawn to the fact that server solutions were characterized by the shortest readout times.

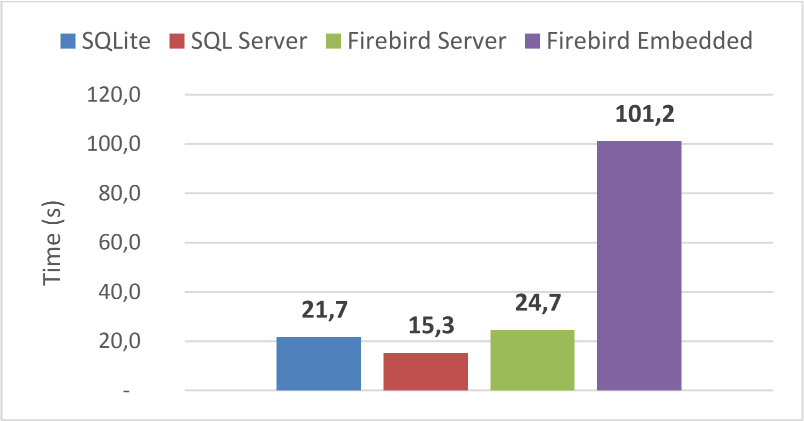

Figure 5. Download of all records, but for each stock in a separate pass

In the case of readout operations of all quotations for individual stocks (Fig. 5 – more than 1200 exchange stocks), there are similar results as in the case of readout of all quotations in one pass (operation 4), but with longer time for each database.

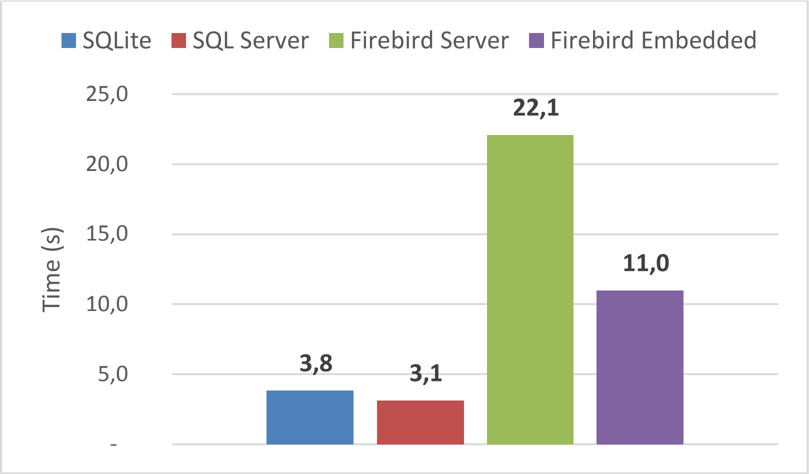

Figure 6. Random readout operation for 10 000 records of quotations

Random readout operation for 10 000 records of quotations (Fig. 6) was the quickest operation in comparison with other operations. This is caused by a small number of readouts in relation to all test base. Here you can also observe a much lower performance of Firebird databases. Simultaneously, it should also be noted that the lowest execution time was achieved by SQL Server database.

4. Conclusions

SQLite turned out to be a database, which as totals in individual tests showed the highest performance. However, we can say that this will not be the appropriate solution for network applications, but it can be recommended as a working database in all cases, where researches are carried out on a single computer station. It can be assumed that the adopted research hypothesis (with the above reservations) was made accurately, but this is not the only base that can be taken into account when constructing the operating environment of genetic algorithms.

SQL Server may be taken into consideration due to the very low data readout times, but it is necessary to consider the cost of full version, which does not have limits of the Express version (maximum use of 1 GB RAM and maximum database size of 10 GB).

Firebird has the lowest performance, but it can be used in server applications, where you cannot afford to buy MS SQL Server database (or other commercial database), and at the same time it enables to scale applications from a simple solution (Embedded) to a full-fledged database server and what is important – server, which unlike SQL Server can be installed not only in Windows operating system, but also on the different versions of Linux.

Extension of the use of proposed procedure for other databases with particular emphasis on No SQL databases, which are becoming more and more applications (including analytical applications), can be assumed as a future direction of researches.

REFERENCES

[1] De Jong K., Fogel D. B., Schwefel H. P. (1997) *A history of evolutionary computation w Handbook of Evolutionary Computation*, Oxford University Press, Oxford

[2] Kennedy J., Eberhart R. C., Shi Y. (2001) *Swarm intelligence,* Morgan Kaufman Publishers, San Francisco

[3] Kwiatkowska A. M. (2007) *Systemy wspomagania decyzji w praktyce*, Wydawnictwo Naukowe PWN SA, Warszawa

[4] Larose D. T. (2008) *Metody i modele eksploracji danych,* Wydawnictwo Naukowe PWN, Warszawa

[5] Łęski J. (2008) *Systemy neuronowo-rozmyte* Wydawnictwa Naukowo-Techniczne, Warszawa

[6] Tarczyński W. (2001) *Rynki kapitałowe, metody ilościowe*, Place Agencja Wydawnicza, Warszawa

[7] Wojarnik G. (2009) *Metody oceny jakości algorytmów genetycznych* w Technologie informacyjne dla społeczeństwa, pr. zb. pod red. W. Chmielarza i T. Parysa, Wyższa Szkoła Ekonomiczno-Informatyczna w Warszawie, Warszawa

[8] Wojarnik G. (2007) *Wykorzystanie metod sztucznej inteligencji w zastosowaniach internetowych*, Społeczeństwo informacyjne – problemy rozwoju, praca zb. pod red. A. Szewczyk, Difin, Warszawa

[9] Wojarnik G. (2013) *Ewolucyjny system analizy danych w warunkach adaptacyjnego środowiska zastosowań informatyki,* volumina.pl, Szczecin

[10] *DB-Engines Ranking* http://db-engines.com/en/ranking (2015-10-30)

[11] *Working with Plain Old CLR Objects (POCO) Classes* http://www.dotnetcurry.com/entityframework/725/plain-old-clr-objects-poco-entity-framework (2015-10-30)