An Approach for Parallel Realization of a Class of Financial Systems

Ivaylo Penev[1] and Anatoliy Antonov[2]

***Abstract* – The work presents an approach for parallel realization of a class of financial systems in a distributed computing environment. The main purpose is overcoming the delay, caused by the concurrent access of simultaneously started parallel jobs to a common data source. Mathematical model for prediction of the jobs' execution times is defined. On the basis of these times a delay start of the parallel jobs is suggested, which aim is to avoid the concurrent access. Preliminary experimental results are applied and discussed.**

***Keywords* – Parallel calculations, Performance model, Distributed computing.**

I. Introduction

The work is concerned with portfolio management systems, used by financial institutions. These systems perform estimations of funds, containing sets of financial portfolios. The estimations are used for making investment decisions. They are obtained by executing various simulation analyses over the positions of each portfolio. Each position is described by historical data for a specific financial instrument (deposit, credit, fund, option, bond, etc.). Furthermore the simulated portfolios are independent each other. As data dependencies between the portfolios do not exist, the simulations could be performed in parallel.

The distributed computing technologies are widely applied in the area of financial calculations [1, 2, 8]. A lot of realizations of financial calculations in distributed computing environment are known. The authors have also published a realization of Monte Carlo simulation of a simple financial instrument in a high-throughput [4] and in a grid computing environment [5]. Many researches on the design and programming of portfolio management systems are known, for example [9]. However no works, referred to the adaptation of existing systems to parallel execution in distributed computing environments are reported.

There are portfolio management systems, originally designed for performance by single processor machines. The source code is either unavailable, or the modification for parallel execution is a complex and an expensive process. The popular parallel techniques (for example MPI) could not be used in this case.

[1]PhD student Ivaylo Penev is with the Faculty of Computer Engineering, Studentska Str. 1, 9010 Varna, Bulgaria, E-mail: ivailopenev@yahoo.com.

[2]Assoc. Prof. PhD Anatoliy Antonov is with the Faculty of Computer Engineering, Studentska Str. 1, 9010 Varna, Bulgaria, E-mail: antonov@eurorisksystems.com.

The estimation of each portfolio is realized as a batch job, requiring the name of the portfolio simulated as a parameter. Due to the natural existing parallelism, these systems could be adopted for realization in distributed computing environments.

The historical data about the past periods of a financial object are stored in a data source (typically data base). The simulations of multiple objects, performing in parallel, cause concurrent access to the data source, which is a limiting condition about the efficiency of the parallel realization [6]. As the source code is assumed to be unavailable, the traditional synchronization primitives and constructs are inapplicable for solving the problem with concurrent access.

A key issue of the realization in a distributed environment is scheduling the parallel jobs' execution [2]. The problem is concerned with deciding in which moment and in which computing node a job is to be started.

A lot of methods and approaches, referring to this problem, are proposed, for example in [2, 8]. They are based on constructing suitable performance model. The model is used for predicting the performance of an application under various conditions in a distributed computing environment. Most often the conditions present the available computing resources and the input data. The target of the model is obtaining different a priori estimations about an application execution in the environment. The most often estimation used is the execution time.

The most cases of performance models' construction, described in the literature, are based on dynamically changing conditions (i.e. run-time conditions), for example input data and different number of computing resources [2, 8]. The definition of such a model is a complex problem. The conditions, under which the financial systems investigated here are typically executed, are defined in advance. The number of jobs performed, the computing resources and the input data are a priori known. Therefore the performance model could be easily constructed, using the advance known conditions. The purpose of the model is predicting the execution time of each job and avoiding the concurrent access of the jobs to the common data source.

In the following part a sequence of steps for definition of mathematical model is proposed. The model is used for predicting the execution times of jobs, estimating financial objects. A formal description of the problem with parallel performance of jobs with known execution times is done. An approach with delay start of the parallel jobs in the distributed environment is proposed, aiming to avoid the concurrent access to a common data source. Finally preliminary experimental results, obtained by the execution of parallel jobs using the proposed approach are discussed.

II. MODEL FOR PREDICTION OF JOBS' EXECUTION TIMES

A. Definition of regression model, presenting the dependence between an object estimation time and the count and type of the consisting instruments.

Real financial objects, consisting of two types of instruments – accounts and shares are studied. The objects have different number of instruments. For each object the most common simulation analyses are performed: back testing store, time series, static performance, historic simulation.

The estimation times for seven different objects for a specific computing architecture are measured.

TABLE I. MEASURED DATA FOR DIFFERENT OBJECTS

Number of accounts	Number of shares	Estimation time (sec)
3	1	32
32	20	118
25	23	91
25	19	110
28	19	113
4	28	110
2	51	231

The modeling purposes to formulate a dependence $y = f(x_1, x_2)$, in which the factors x_1 and x_2 present the number of accounts and shares, building an object, and y is the time for the object estimation (completing all the simulation analyses) in seconds.

A non linear model is examined:

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_{12} x_1 x_2 + b_{11} x_1^2 + b_{22} x_2^2 \quad (1)$$

The values of the coefficients b_0, b_1, b_2, b_{12}, b_{11} and b_{22} are determined by regression analysis using the method of least squares [3]. The steps of the classic decision are presented below:

- Finding the middle of the change intervals – main factors' levels.

$$x_{i0} = \frac{x_{i\max} + x_{i\min}}{2} \quad (2)$$

- Determining the varying intervals.

$$\lambda_i = \frac{x_{i\max} - x_{i\min}}{2} \quad (3)$$

- Coding the natural values of the factors t_i.

$$x_i = \frac{t_i - x_{i0}}{\lambda_i} \quad (4)$$

- Constructing an extended matrix with coded factors F using the results (2), (3) and (4).
- Constructing a transposed matrix F^T.
- Constructing an information matrix $F^T F$.
- Constructing a covariance matrix $(F^T F)^{-1}$.
- Calculating the regression coefficients.

The vector with the regression coefficients estimations is obtained by matrix multiplication of a linear equations system:

$$b = (F^T F)^{-1} F^T y \quad (5)$$

After solving the system (5) the following estimations of the coefficients are found:

$b_0 = 84.72$, $b_1 = -14.79$, $b_2 = -29.34$, $b_{12} = -132.27$, $b_{11} = 7.38$ and $b_{22} = 21.18$.

Finally the functional dependence is:

$$y = 84.72 - 14.79 x_1 - 29.34 x_2 - 132.27 x_1 x_2 + 7.38 x_1^2 + 21.18 x_2^2 \quad (6)$$

B. Estimating the adequacy of the model [3]

The prognoses $\tilde{y}$ and their declinations ε from the regression model are determined.

The sum of the declination squares in this case is $Q_{oct} = 6.12$.

Ten additional measurements about $x_1 = 32$, $x_2 = 20$ are done:

y_{c1}	y_{c2}	y_{c3}	y_{c4}	y_{c5}	y_{c6}	y_{c7}	y_{c8}	y_{c9}	y_{c10}
118	120	119	117	120	116	118	117	115	115

Statistic estimations of the additional data are calculated: average value – 117.5, dispersion s_{yc}^2 - 3.39.

The degrees of freedom about the additional experiment are determined:

$\nu_c = N_c - 1 = 9$, where $N_c = 10$ - number of additional measurements.

The degrees of freedom about the model are determined:

$N = 7$ – number of measurements, $k = 6$ – number of coefficients, $\nu_{mod} = N - k = 1$ – degrees of freedom about the model.

The dispersion about the proposed model is $s_{mod}^2 = \frac{Q_{oct}}{\nu_{mod}} = 6.12$.

The model is tested using the Fisher criteria. Considering significance level $\alpha = 0.05$ the critical values is $F_{critical}(\alpha, \nu_{mod}, \nu_c) = 5.12$.

The number of Fisher about the model is $F = \frac{s_{mod}^2}{s_{yc}^2} = 1.81$.

The comparison of the number of Fisher with the critical value shows that $F < F_{critical}$. The conclusion is that the proposed model (6) is adequate.

The described approach could be used for defining model, predicting the times for reading data, performing simulation calculations and storing data into the common source about each parallel job.

III. Approach for Avoiding the Concurrent Access of Parallel Jobs to Common Data Source

A. Formal description of the problem

Using the regression model, defined in the previous part, the following formal description of the problem about execution of parallel jobs in a distributed computing environment could be done:

A set of n parallel jobs is executed in a distributed environment, consisted of n computing nodes (computers with processors). Each job performs a set of simulation analyses over a financial object. The execution time of each job consists of time for reading data t_{READ}, time for calculating operations t_{CALC} and time for storing results t_{STORE} in a common data source. The times are functionally dependent from the number of instruments, constructing the current job. The function is defined (part II).

The total time of the parallel execution of all jobs is:

$t = max(t_{READ} + t_{CALC} + t_{STORE}) + t_{DELAY}$,

where t_{DELAY} is delay time, caused by the concurrent access of the jobs to the common source. The final target is minimizing the time t_{DELAY} .

B. An approach for solving the problem

The execution time of each job is divided into time intervals, for example each second from the execution time presents an interval. In the beginning of each job's execution a new state ***wait*** is introduced. The job *i* remains in this state until next one *i+1* accesses the common data source (i.e. performs data reading or writing). This state presents the delay starting of the job. After the job is shifted along the time axis, the number of intervals, in which two or more jobs access the common resource is counted. Multiple scenarios with jobs' delay starting are simulated. The final purpose is finding a scenario, in which the delay starting minimizes the concurrent access and decreases the t_{DELAY} time.

This approach is demonstrated for two parallel jobs with known times t_{READ}, t_{CALC}, t_{STORE} .

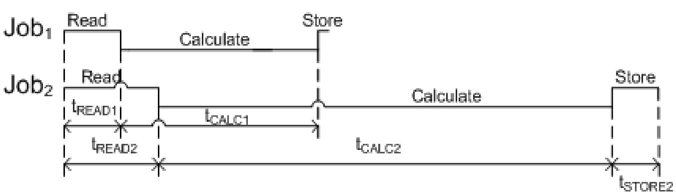
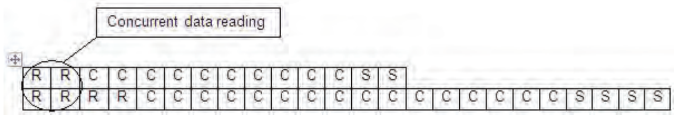


Fig.1. Simultaneous starting of two parallel jobs

In the current case of simultaneous starting a concurrent access in two time intervals of reading is observed. The separate stages of a job's execution are marked as follows: W-waiting (delay start), R – reading data, C – calculations, S-storing results.



Two scenarios with delay starting of job 1 and job 2 are simulated (Fig. 2, Fig. 3).

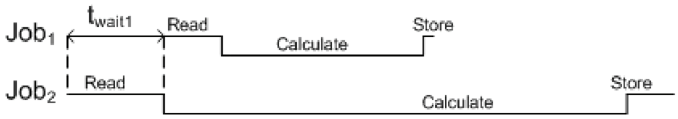


Fig.2. Delay starting of job 1

Fig. 3. Delay starting of job 2

Various algorithms for realizing this approach are possible. In [7] an iterative algorithm is proposed. Genetic algorithms could also be applied. The purpose is diminishing the necessary time for modeling the execution of multiple jobs.

IV. Experiments and Results

Experiments with estimation of two real portfolios are carried out. Each one consists of instruments (accounts and shares) with different number of positions. The distributed environment includes two computing nodes, working under the control of the high-throughput computing system Condor [10] and an Oracle data base management server. In the following tables the parameters of the nodes and the measured execution times of the jobs are shown. The times t_{READ}, t_{CALC}, t_{STORE} about each job are calculated by the mathematical model defined.

TABLE II. Data about the jobs, performed in the distributed environment

Job	Accounts	Shares	t_{READ} (sec.)	t_{CALC} (sec.)	t_{STORE} (sec.)	Total (sec.)
1	4	28	12	106	1	119
2	2	51	30	243	1	274

TABLE III. Parameters of the computing nodes, executing the jobs

Host name	Processor	Operating memory	Executing job
Host1	Intel 1,6 GHz	1GB	Job 1
Host2	AMD 3200+ 1,79 GHz	2GB	Job 2

The jobs are executed in sequence, in parallel with no delay start and with delay start in different order. The results are summarized in the next tables:

TABLE IV. SEQUENTIAL EXECUTION

Job	Execution time (sec.)
Job 1	119
Job 2	274
Total time	**393**

TABLE V. PARALLEL EXECUTION WITH NO DELAY START

Job	Execution time (sec.)
Job 2	304
Job 1	149
Total time	**304**

TABLE VI. DELAY START OF JOB 1

Job	Execution time (sec.)
Job 2	290
Job 1	120
Total time	**290**

TABLE VII. DELAY START OF JOB 2

Job	Execution time (sec.)
Job 1	190
Job 2	289
Total time	**289**

The results present the time benefit, obtained by the parallel execution of the jobs. The increased execution time in the case of parallel execution with no delay start is caused by the concurrent access to the common resource (Table V). The results from Table VI and Table VII show that the presented approach improves the time parameters of the parallel execution.

The last tables also show that the order of starting influences the execution time of job 1. The starting of the job with more positions leads to decreased time for execution of the other job with almost 60 seconds for the given experimental environment (Table VI). This result is explained with the data buffering, realized by the Oracle system. The jobs use common historical data. As the job with more instruments (in this case job 2) loads the necessary data, they could be used by the other job (job 1) directly from the cache memory. This means, that the usage of such starting order guarantees the earlier freeing of the second computing node, which could be used for starting of another job.

V. CONCLUSIONS AND FUTURE WORK

The presented work proposes an approach for constructing performance model for parallel realization of a class of financial systems. The model includes two parts. First functional dependence of the execution times for each step of a job from the number and type of instruments is defined. Afterwards scenarios with delay starting of parallel jobs are simulated. The aim of the delay starting is preventing the concurrent access to the common data source. Preliminary results from the execution of parallel jobs for estimation of two portfolios in a distributed computing environment are discussed. The total execution times are compared with the same times, obtained by sequential execution of the jobs, and by parallel execution with no delay starting.

As the study of this approach is still in initial stage, the results, presented above, are obtained in the simplified case with two parallel jobs. The future authors' work is concerned with summarizing and researching the proposed approach for parallel execution of *n* jobs from the same class in distributed environment, consisting of *m* computing nodes.

ACKNOWLEDGEMENT

The work presented in this paper was supported within the project BG051PO001-3.3.04/13 of the HR Development OP of the European Social Fund 2007-2013.

REFERENCES

[1] Asanovic K., Bodik R., et., The Landscape of Parallel Computing Research: A View from Berkeley, Technical Report No. UCB/EECS-2006-183, December 18, 2006.

[2] Kerbyson D. J., Papaefstathiou E., Nudd G. R., Application execution steering using on-the-fly performance prediction, High-Performance Computing and Networking, Lecture Notes in Computer Science, vol. 1401, Springer, Amsterdam, 1988, pp. 718-727.

[3] Miser H. J., Handbook of Operations Research. Foundations and Fundamentals, Mir, Moskva, 1981 (in russian language).

[4] Penev I., Implementation of Monte Carlo simulation in a distributed computing environment, International Conference on Computer Systems and Technologies - CompSysTech'08, Gabrovo, 2008.

[5] Penev I., A. Antonov, Possibilities of Grid computing for realization of simulation problems, Conference on Computer Systems and Technologies - CompSysTech'09, Rousse, 2009.

[6] Penev I., A. Antonov, Realization of Portfolio Management System in a distributed computing environment, International Scientific Conference Computer Science, Sofia, 2009.

[7] Penev I., A strategy for management of parallel jobs with concurrent access to a common resource in a distributed computing environment, Conference Informatics in Scientific Knowledge, Varna Free University, 2010.

[8] Perry S. C., R. H. Grimwood, D. J. Kerbyson, E. Papaefstathiou, G. R. Nudd, Performance optimization of financial option calculations, Parallel Computing 26, 2000, pp. 623-639.

[9] Trobec R., M. Vajtersic, P. Zinterhof, Parallel Computing, Numerics, Applications and Trends, ISBN 978-1-84882-408-9, Springer, London, 2009.

[10] http://www.cs.wisc.edu/condor/