Harmony Search for Data Mining with Big Data

Jerzy Balicki, Piotr Dryja, Waldemar Korlub

Harmony Search for Data Mining with Big Data

J. Balicki[1], P. Dryja[2], W. Korłub[2]

[1]Faculty of Mathematics and Information Science, Warsaw University of Technology, Koszykowa St. 75, 00-662 Warsaw, Poland
[2]Faculty of Telecommunications, Electronics and Informatics, Gdańsk University of Technology, Narutowicza St. 11/12, 80-233 Gdańsk, Poland
j.balicki@mini.pw.edu.pl, piotrdryja83@gmail.com, waldemar.korlub@pg.gda.pl

Abstract. In this paper, some harmony search algorithms have been proposed for data mining with big data. Three areas of big data processing have been studied to apply new metaheuristics. The first problem is related to *MapReduce* architecture that can be supported by a team of harmony search agents in grid infrastructure. The second dilemma involves development of harmony search in preprocessing of data series before data mining. Moreover, harmony search as a classification algorithm is studied as the third application. Finally, some outcomes for numerical experiments are submitted.

1 Introduction

A goal of this paper is to describe an approach based on harmony search (an acronym HS) for data mining with Big Data (BD). Although, there are several data mining algorithms, including back-propagation neural networks or locally weighted linear regression, we can extend these set to better support of parallelism in Big Data processing and to avoid some limitations of well-known machine learning procedures like k-Means, or support vector machines [5].

We assume that data is usually gathered from multiple sources, which may be heterogeneous and spread geographically across the world. Also, the collected data may be stored in distributed facilities. Data mining algorithms can be applied for large-scale multimedia applications, in massive parallel way, to ensure higher capacity of training. In fact, both logistic regression and Gaussian discriminant analysis can be used concurrently for different parts of images to make decision for the whole pattern. Similarly, naive Bayes or the independent variable analysis can be developed, simultaneously [12].

A motivation for this paper is fact that a quality of BD processing mainly depends on services provided by some public cloud computing platforms. Especially, BigQuery service that is delivered by Google Cloud can be applied to analyze data in the cloud by SQL-like queries. A query for BD is usually performed for multi-terabyte datasets in 1-2 seconds. This service is easy to use and it is scalable. In consequence, BD services provided by some public cloud computing platforms can offer real-time insights about large-scale multimedia data [11].

MapReduce is a batch-oriented parallel cloud computing model and it can be applied to some machine learning algorithms because they regularly prerequisite to probe through the training data. It needs exhaustive computing to entrance the large-scale data recurrently. It is important that MapReduce has still got a certain advantage in performance over some relational databases [16].

In this paper, related work is described in Section II. Then, MapReduce architecture is characterized in Section III. Next, Section IV presents some studies under intelligent agents in data mining. Genetic programming in data mining is discussed in Section V. Moreover, some harmony search algorithms for data series are studied in Section VI. We focus on time series during experiments. A description of parallelism in big data is included in Section VII. Furthermore, harmony search for big data preprocessing and some outcomes for numerical experiments with harmony agents are submitted in Section VIII.

2 Related work

Main characteristics of dynamic data streams are continuity and unpredictability. What is more, some crucial features are quickness and infinity, too. It is worth to underline that above features of dynamic data streams can loss some valuable information. Data streams are extensively applied in medical testing, online trading, and financial analysis. Chang, Bai, and Zhu proposed some parallel algorithms for mining large-scale rich-media data that can be treated as the massive, multisource, and dynamic Big Data [13]. Some data mining methods have been extended from the single-source knowledge discovery methods.

Moreover, Chen, Sivakumar and Kargupta studied collective mining of Bayesian networks from distributed heterogeneous data [14]. Especially, they analyzed stream data that are dynamic and the other approach for data mining should be applied. Similarly, Domingos and Hulten extended a mining method on high-speed data streams that can be applied to visual information processing on large-scale [17].

Big Data are gathered from different sources. Due to some computer hardware improvements, some new knowledge discovery algorithms can be developed for massive data. That is why, there has been constructed a logical framework for identifying quality knowledge from different data sources. Besides, some data mining algorithms have been studied for a multisource massive data.

Some crucial differences between multisource data withdrawal and single-source knowledge mining are related to massive characteristics of multisource data. So, we can distinguish some characteristics for heterogeneous and real-time data. Some characteristics of flow and a data stream require some knowledge mining algorithms. Recently, the theory of local pattern analysis has been introduced to avoid centralized computing.

Banerjee and Agarwal studied collective behavior from blogs using swarm intelligence [7]. On the other hand, it has been proposed a paralleled big data algorithm with MapReduce framework for mining data from Twitter [10]. MapReduce is adopted to parallelize the mining algorithm as well as to ensure scalability of knowledge

discovery system. A performance can rise despite growths in data set size. A performance ratio raises as the size of the dataset growths, and as the number of data nodes rises. Several millions of active users of Twitter post over 140 million messages every day and a big data set may contains billions of records. So, a new architecture to process the data mining algorithm is required [19].

On this background, harmony search can be applied for big data mining in different areas [6]. In this evolutionary-based metaheuristic, an optimization problem is imitated as a melody performance procedure with searching for a good harmony. HS models how an orchestra conductor or a music composer optimizes a melody performance to achieve good harmony music. Similarly to jazz improvisation when musicians seek a best harmony by artistic intelligence, some outcomes from an optimization algorithm tend to a global optimum regarding objective function.

3 Hadoop and MapReduce Architecture

MapReduce has recently gained in popularity as one of the key concepts related to BD. It was introduced by Dean and Ghemawat as a distributed and scalable solution for parallel processing and large data sets on clusters [16]. Many different commercial and non-commercial products offer a variety of different implementations of MapReduce making the everyday use relatively easy. The most popular open-source implementation is Apache Hadoop, embraced by companies such as Adobe or Amazon [1].

MapReduce can be perceived as a universal model which can be used in a variety of scenarios. This is due to the fact that the model requires two custom functions to be provided by a user. These functions, known as map and reduce, are responsible for domain-specific computation not covered by the model. The map function takes the pair (key, value) as an input, then it performs computation and returns a set of other intermediate pairs (key, value). The key is not unique and it is even usually expected to be many pairs with the same key. The reduce function is executed after the map function in the logical. Usually, the role of the reduce function is to merge the input values to a smaller set. It is therefore possible to define the map and reduce functions, as follows [26]:

$$\text{map } (k1,v1) \rightarrow \text{list}(k2,v2)$$
$$\text{reduce } (k2,\text{list}(v2)) \rightarrow \text{list}(v2)$$

An execution flow in MapReduce model consists of four major phases (Fig.1). The first phase is dedicated to splitting the input required to be processed. Let the text with random words be considered. Each of the splits can then be processed by the map function in parallel. The particular word is the key and the '1' is the value. Some intermediate pairs (key, value) must be then sorted and grouped by a key. This is because the particular key and its values are passed to the reduce function in the next phase. The reduce function can be also run in parallel and after computation results are being sent to output. In this case the reduce function's responsibility is to sum the each key's values (Fig. 1) [29].

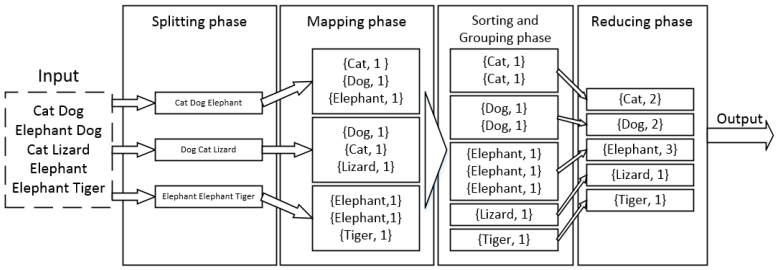

Fig. 1. MapReduce execution flow illustrated with the case of counting the number of occurrences of each word

Apache Hadoop is a widely known open-source project that offers the set of libraries with the MapReduce implementation and own implementation of the distributed file system – Hadoop Distributed File System (HDFS). These two modules can be perceived as one core. However, Hadoop with other related projects offer much more additional functionality nowadays. Some the Hadoop ecosystem solutions are worth mentioning such as Apache Pig, which offers custom scripting language and Apache HBase – a non-relational distributed database which can serve as a data input for MapReduce.

MapReduce is applied to solve several problems like large-scale machine learning or clustering problems for the Google News. Moreover, an extraction of data is used to produce reports of popular queries and extraction of geographical locations from a large corpus of web pages for localized search. In 2004, Google changed an indexing system that produces data used for web search service to system that used MapReduce. The new indexing system takes input documents that have been retrieved from a crawling system store as a set of files, and then they are processed by from five to ten MapReduce operations. That system gives them many benefits like simpler, smaller, easier to understand code, simplicity to change entire indexing process. Moreover, it is easier to operate because of automatic resolving problems like machine failures, slow machines and networking hiccups [1].

In 2008, New Your Times released service web-based archive of scanned issues from 1851 till 1980. To create that archive company had to convert 405,000 large TIFF images, 3.3 million SGML files, and 405,000 XML files. Output consisted on 810,000 PNG images and 405,000 JavaScript files. To speed up this process they applied Amazon Elastic Compute Cloud and Hadoop. Regarding to hundreds of machines, the process took less than 36 hours [16].

4 Intelligent Agents and Genetic Programming in Data Mining

We consider agents that are based on harmony search, generally. There are several traits of multiagent systems that make them suitable for BD acquisition and pro-

cessing. One of them is mobility of agents, which translates into an ability to move between different data facilities and processing nodes constituting the system that deals with BD. Software agent is not bound to any particular machine or execution container [24]. By migrating to different machines, agents can get closer to the source of an online data stream or closer to the data storage they are about to process. Instead of having to provide the data to the system, data administrators can rely on agents that acquire it on their own. It reduces bandwidth requirements and communication delays [22]. Another important factor is the ability of agents to react upon sudden changes of the environment in which they reside. It provides means for handling changes in the availability of data sources. An agent can make decision about moving to another set of data.

Agents are designed with heterogeneous environments in mind [25]. It translates to capability of integrating with different data sources. Agents can integrate with multiple different data stores and handle online streams of information. Acquired data can be later delivered to the processing algorithm in a unified form without the need to use internal storage mechanisms of any particular software framework, e.g. distributed file system of the aforementioned Hadoop platform. One trait of agents that aids the integration with online data streams is their reactive nature. Each new information fragment delivered in the stream becomes an event that agent should react upon.

However, agents can not only react upon external events but also take actions on their own, which is an effect of their pro-activeness. Knowledge extraction from BD is one of the fields, where this trait is particularly important. As there are often no clues in advance about possible interrelationships in BD, the final result is heavily influenced by the decisions made during the actual analysis. Pro-active architectures provide foundation for agents that need to work under uncertainty and act without complete knowledge about their environment.

As was previously mentioned, agents can easily handle acquisition of data from both offline datasets and online streams. Agents can be also deployed inside the lambda architecture to aid actual analysis of data coming from those two distinctive types of sources. The lambda architecture itself is built of three layers: the batch layer, the serving layer and the speed layer. The batch layer is responsible for offline data processing and uses solutions like the MapReduce architecture. Batch view produces by this layer are exposed to external clients by the serving layer. The speed layer is responsible for real-time processing of data streams. It analyses data that was not yet processed by the batch layer. Speed layer produces real-time views that can be merged with batch views to create complete representation of the extracted knowledge.

The main motivation for employing agents is the fact that implementation of a system based on lambda architecture requires integration of several heterogeneous components: one for batch processing, another one for serving views, a different solution for real-time stream analysis, and a component that merges real-time views with batch views. The use of multiagent environment provides a common ground for information exchange between different component and a common execution model.

Agents can improve efficiency of data mining compared to centralized approaches. Multiagent systems have been applied in different domains showing promising

results for further research, e.g. banking and finance domain or resource allocation in distributed environments [4, 22].

Some special sort of intelligent agents are based on genetic algorithms that have several advantages in data mining. One of them is its feature of automated searching for optimized solutions without a need for prior knowledge of data. Evolutionary computing also constantly evaluates created models as a whole, which is not a case with most traditional machine learning algorithms. It should be mentioned that this approach is easily adapted to suit researcher needs by changing it evaluation method or tree representation [32]. Genetic programming is being used in wide range of data mining techniques like classification [23].

Although traditional solutions like C4.5 are usually faster than genetic programming, its execution time can be improved by removing parts of decision tree that do not contribute to overall result of evaluation. These parts are named introns [30].

Genetic programming performance can be improved by using some optimization techniques. One of them is search space refinement by dividing existing search space into smaller groups. Researchers may also extend tree representation by using fuzzy logic to improve created models [18]. When compared genetic programming solutions are often better than traditional data mining tools including simple genetic algorithm.

5 Harmony Search for Data Series

Big data requires some database capacities from terabytes to even zetabytes. Capacity of big data depends on the sort of application, e.g. capacity of five terabytes is a great enough for a banking transaction system, but too small to test a web search engine. BG are uncooperative with using some relational database management systems RDBMS, too [8].

Parallelism for big data can be supported by using open grids instead of clouds. In the grids like BOINC, Comcute@home, or SETI@Home data sets are transformed into several millions of subsets that are executed parallel by thousands of computers. In grid, we can use a massively parallel system with lots of CPUs, GPUs, RAM and disks to obtain a high performance by data-based parallelism. So, maximal performance can be estimated at PFLOPS [9].

That is why, we suggest using grid as virtual supercomputer to data mining of big data. In an experimental grid called Comcute, a dilemma with capacity of BD appears if we study some simulations of fire spread to find some strategies. So, we design some intelligent agents for support parallelism in BD queries [15].

Two kinds of intelligent agents act in a middleware layer. Managing agents send data series from distributed sources to distribution agents that cooperate with web computers to calculate outcomes and return them to managing agents, and then to users. Both types of agents can autonomously move from one host to another to improve quality of grid resource using [2].

Moreover, two groups of the other agents based on harmony search as well as genetic programming, optimize a resource usage. A set of agents designed for local optimization consists of some harmony search schedulers. They cooperate with dis-

tributors and managers to give them information about optimal workload in some parts of grid. Finally, genetic programming has been applied for finding the compromise configurations of the whole grid. These agents cooperate with harmony search schedulers to correct some local timetables.

Crucial difficulties in parallelism of Comcute are data capture, regarding different sources: sensors, smartphones, cameras, tablets, microphones, computer simulations, satellites, and social networks via some wireless sensor networks. Moreover, data storage, their visualization, analysis and search are still some open problems.

BD is not convenient to the most RDBMS. Advantage of grid development for data basis is related to a fact that an intensive progress in data communication capacity is observed. For instance, the dilemma of finding the optimal strategy against a fire spread can be identified as the 4Vs model descripted by: high volume, extraordinary velocity, great data variety, and veracity.

In data mining for BD, harmony search can be used to find predictions for data series. Most of methods in data mining for time series consider some discrete time series. On the other hand, some analog time series are discretized with floating-point values. Moreover, time series discretization for knowledge discovery method like artificial neural network can be optimized [27].

Time series values can be represented by a multi connected graph. Under this representation, similar time series can be grouped into a graphical model. However, this approach works with one time series at a time, only. In result, it is inconvenient for parallel processing.

A symbolic representation of time series called SAX allows on dimensionality reduction, and it also allows distance measures to be defined on the symbolic approach where the lower bound corresponding distance measures is defined on the original series. Moreover, Genetic Entropy Based Linear Approximation GENEBLA can be used for time series discretization. For this approach, the EBLA3 algorithm is based on the simulated annealing and it permits on finding an alphabet size and word size to maximize accuracy in classification [20].

Some data mining algorithms for discrete time series require a word size to separate the length of time series, and also an alphabet size to reduce the series. Therefore, a harmony search algorithm can be used to find an optimal word size and alphabet size. Then, they can be used by SAX process to discrete time series for big data sets. So, the question is how to maximize both word size and alphabet size for given data sets. It permits to compress data sets and improve classification accuracy. In this approach, we protect some essential data and hidden patterns.

6 Parallelism in Big Data

Big data applications can develop Hadoop or NoSQL cluster like MongoDB with different nodes to deal with online transaction processing OLTP. Additionally, low response time for decision-support queries is related to online analytical processing OLAP to answering multi-dimensional analytical queries. A required reliability can be obtained through data replication [21].

Figure 2a shows a case for concurrent and different queries that operate on the same data.

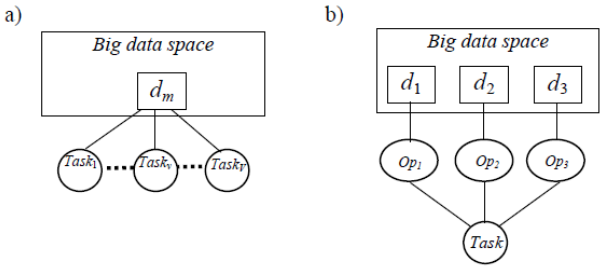

Fig. 2. Two cases of data-based parallelism

The second case (Fig. 2b) is related to complex query that is divided on some parallel operations acting on diverse data. Architecture with shared-disk cluster is much more prepared for big data processing than NUMA Non-Uniform Memory Access architecture [31]. A crucial feature of BD is related to intensive reading from hard disks and then processing, instead of processing and then intensive writing. If we consider no sharing of memory or disks across nodes (Fig. 3), this system requires data partitioning of database like in server DB2.

Big data is spread over some partitions that run on some separate servers with own table spaces, logs, and configurations. A query is performed in parallel on all partitions. Such architecture can support Google search engine.

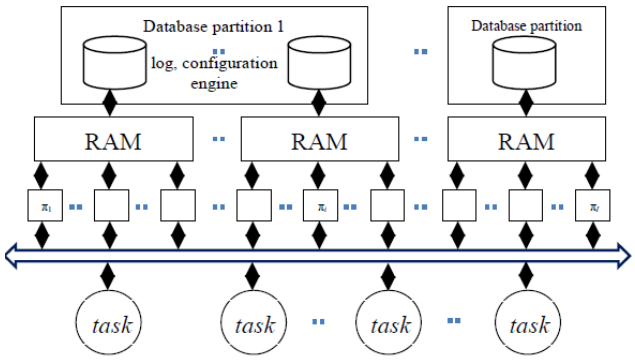

Fig. 3. Shared-nothing cluster architecture for big data [1]

7 HS for BD Preprocessing and Task Optimization

To avoid the predefinition of alphabet size and word size the entropy measure can be used. Entropy minimization heuristic EMH divides continuous value and minimum description length criteria to control the number of interval produced on continuous space. The stopping criterion for this technique is minimum description length principle MDLP.

We have M time series of data. If the length of time series is equal to N, then a word size α_m is a decision variable that cannot exceed N. We can vary a word size from 1 to N. So, we can introduce the inequality constraint, as below:

$$\alpha_m \le N, \ m = \overline{1, M}. \tag{1}$$

The maximum entropy is based on information gain and the objective function can be defined, as follows:

$$f(x) = \frac{\sum_{m=1}^{M} g_m(x)}{M}, \tag{2}$$

where
M – the time series length;
$g_m(x)$ – the gain between time series value TSV and the mth class value CV_m;
$x = (\alpha_1, \beta_1, \ldots, \alpha_m, \beta_m, \ldots, \alpha_M, \beta_M)$;
α_m – a word size for the mth series;
β_m – an alphabet size for the mth series.

The characteristics of time series value TSV is given:

$$Ent(TSV) = p(S \mid v)_m \log_2 p(S \mid v)_m. \tag{3}$$

The time series value TSV can be characterized by tsv - the number of time series with value in S. The characteristics of time series class TSC is given, as follows:

$$Ent(TSC) = -\sum_{j=1}^{J} p_j \log_2 p_j, \tag{4}$$

where
J - the number of classes;
p_j - the probability of class j in TSC.

It is important to remark that the entry time series is considered as one attribute and that the discretization scheme is considered for the whole dataset. It allows harmony search to find a good global solution that maximizes the entropy on data.

For the mth time series HS algorithm is applied to obtain the word size α_m. This the mth time series is formulated regarding new word size αm. Next, against harmony search is used to obtain the alphabet size β_m. Afterwards, the SAX procedure is developed to convert the values of time series to symbolic representation using new alphabets size β_m. Finally, any classification algorithm is applied to evaluate the performance of new data representation.

Figure 4 shows architecture of the grid Comcute with two agents AHS1 and AHS2 that are based on harmony search. One AHS can find suboptimal configuration for at most 15 nodes, 50 tasks and 15 alternatives of resource sets ARS. It represents a volunteer computer.

Figure 5 shows an example of finding the compromise configuration by *AHS1*. The first criterion is the CPU workload of the bottleneck computer ($\hat{Z}_{max}$), and the second one is the communication workload of the bottleneck server ($\tilde{Z}_{max}$) [2]. This problem is the constrained bi-criteria optimization dilemma with integer decision variables

[2]. There are presented Pareto-optimal evaluations obtained by harmony search during 10^6 iterations (Fig. 5).

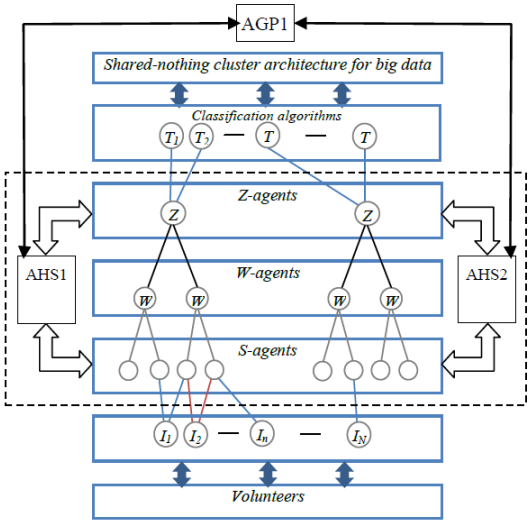

Fig. 4. Harmony search agents HAS in the grid for big data processing

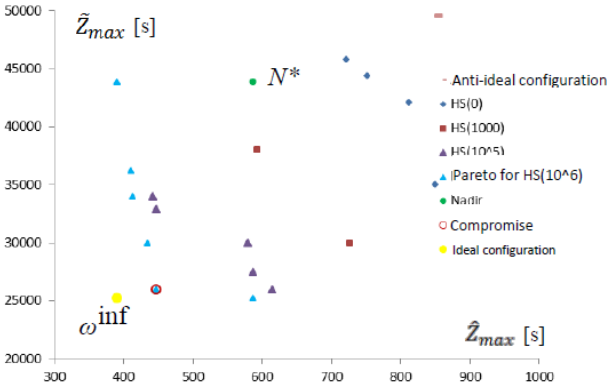

Fig. 5. Finding a compromise configuration by AHS1 in the Comcute grid

The harmony search algorithm has been used with some tuned parameters: *HMCR*=0.9, *PAR=0.5*, *BW1=3*, *BW2=4*, *HMS*=30 and *HMS2*=20. The run time for 1 000 000 updating of the harmony memory HM was 5 hours on PC *Windows 7/Intel Core i7-2670 QM 2,2 GHz.* An algorithm during one run found an ideal point ω^{inf}=(411; 25 221) [s], the nadir point N^*=(587; 43 863) [s] and the anti-ideal point ω^{sup}=(850; 49 577) [s]. Then, the criterion space is normalized that permits on finding the compromise solution. The distance $\varsigma_2(\overline{\omega}^{Sal}) = \inf_{\omega \in \boldsymbol{\Omega}} \xi_2(\overline{\omega})$ is equal to 0,127. In the normalized criterion space $\overline{\boldsymbol{\Omega}}$, there is $\overline{\omega}^{Sal}$ =(0,124; 0,030), and the compromise evaluation ω^{Sal} is (448; 25 952) [s] in the criterion space $\boldsymbol{\Omega}$. A new solu-

tion x^{Sal} has the following characteristics: Z_{suma}=318 001 [s], Δ_{max} = 13 200 [s] for $\rho_1 = 0{,}5$ $\rho_2 = 0{,}5$, Θ=453 450 in units of the CPU *Mark*, Ξ =325 875 [$], υ=856 and E=19 240 [W].

Moreover, there are satisfied constraints on RAM and HDD. An encoded configuration is as follows:
X^{α}=[10,7,12,4,1,11,12,10,5,8,3,14,7,13,11,9,8,5,6,6,1,3,14,10,11,6,13,2,4,5,9,13,9,7,9, 4,6,2,2,5,3,1,4,12,2,]^T, X^{β}=[3,8,3,9,8,9,8,3,9,3,3,3,3,3,0]^T. If X^{α}[1]=10, the module with the index 1 is assigned to the node number 10. So, some clusters of software modules in nodes can be obtained from X^{α}. If X^{β}[1]=3, the computer number 3 is assigned to the node number 1. In a compromise solution, there are considered three from twelve possible computers that are assigned to fourteen nodes. The new grid consists on 8 servers *BizServer* E5-2660v2, 3 servers HP *ProLiant* E5-2695, and 3 servers HP *ProLiant* E5-2697.

8 Concluding Remarks and Future Work

Intelligent agents based on harmony search in the middleware of grid can significantly support efficiency of the proposed approach. Multi-objective harmony search can be used for the self-reconfiguration of the grid. Agents based on harmony search can optimize a problem of grid resource using.

Our future work will focus on testing the harmony search to find the compromise configurations for different criteria and constraints. Moreover, quantum-inspired algorithm can be analyzed due to supporting big data, too [3].

References

1. Apache Hadoop, http://hadoop.apache.org/, Accessed 8 March 2016.
2. J. Balicki, "Negative selection with ranking procedure in tabu-based multi-criterion evolutionary algorithm for task assignment," In: Alexandrov V.N., Van Albada G.D., Sloot P.M.A. et al.: Proc. the 6th Int. Conf. on Computational Science, Reading, England, Lecture Notes in Computer Science, Vol. 3993, 2006, pp. 863-870
3. J. Balicki, "An adaptive quantum-based multiobjective evolutionary algorithm for efficient task assignment in distributed systems," In: Mastorakis N. et al. (Eds.): Recent Advances in Computer Engineering. Proc. of the 13th WSEAS Int. Conf. on Computers, Rhodes, Greece, 2009, pp. 417-422
4. J. Balicki, Z. Kitowski, "Multicriteria evolutionary algorithm with tabu search for task assignment," In: Zitzler E, Deb K, Thiele L, et al.: Proc. the 1st Int. Conf. on Evolutionary Multi-Criterion Optimization, Zurich, Switzerland, Lecture Notes in Computer Science, 2001, Vol. 1993, pp. 373-384
5. J. Balicki, W. Korłub, J. Szymański, M. Zakidalski, "Big data paradigm developed in volunteer grid system with genetic programming scheduler," In: L. Rutkowski et al. (Eds.): Artificial Intelligence and Soft Computing. Lecture Notes in Computer Science, 8467, Proc. of the 13th Int. Conf. on Artificial Intelligence and Soft Computing ICAISC, Part II, Zakopane, Poland, June 1-5, 2014, pp. 771-782
6. J. Balicki, W. Korlub, H. Krawczyk, et al., "Genetic programming with negative selection for volunteer computing system optimization," In: Paja W.A., Wilamowski B.M.: Proc. the 6th Int. Conference on Human System Interactions, 2013, Gdańsk, Poland, pp. 271-278

7. S. Banerjee and N. Agarwal, “Analyzing collective behavior from blogs using swarm intelligence,” Knowledge and Information Systems, vol. 33, no. 3, Dec. 2012, pp. 523-547
8. E. Birney, “The making of ENCODE: Lessons for Big-Data projects,” Nature, vol. 489, 2012, 49-51
9. BOINC, http://boinc.berkeley.edu/. Accessed 25 February 2015
10. J. Bollen, H. Mao and X. Zeng, “Twitter mood predicts the stock market. Journal of Computational Science”, vol. 2 (1), 2010, pp. 1-8
11. J. Bughin, M. Chui, J. Manyika, “Clouds, big data, and smart assets: ten tech-enabled business trends to watch,” McKinSey Quarterly, 2010
12. L. Cao, V. Gorodetsky, P.A. Mitkas, “Agent mining: the synergy of agents and data mining,” IEEE Intelligent Systems, 2009, Vol. 24, pp. 64-72
13. E.Y. Chang, H. Bai, K. Zhu, “Parallel algorithms for mining large-scale rich-media data,” Proc. ACM Int. Conf. Multimedia, 2009, pp. 917-918
14. R. Chen, K. Sivakumar, H. Kargupta, “Collective mining of Bayesian networks from distributed heterogeneous data,” Knowledge and Information Systems, vol. 6, no. 2, 2004, pp. 164-187
15. Comcute. http://comcute.eti.pg.gda.pl/. Accessed 25 January 2016
16. J. Dean, S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” Communications of the ACM, 2008, Vol. 51, pp. 1–13
17. P. Domingos, G. Hulten, “Mining high-speed data streams,” Proc. Sixth ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2000, pp. 71-80
18. J. Eggermont, “Data mining using genetic programming: classification and symbolic regression,” PhD Thesis, 2005
19. D. Gillick, A. Faria, and J. DeNero, “MapReduce: distributed computing for machine learning,” Berkley, Dec. 2006
20. T. Gunarathne et al, “Cloud computing paradigms for pleasingly parallel biomedical applications,” In: Proc. of the 19th ACM International Symposium on High Performance Distributed Computing, Chicago, Illinois, 2010, pp. 460-469
21. L. Guojun, Z. Ming, Y. Fei, “Large-scale social network analysis based on MapReduce,” In: Proc. Int. Conf. on Computational Aspects of Social Networks, 2010, pp.487-490
22. N.R. Jennings, M. Wooldridge, “Applications of intelligent agents,” In: Jennings NR, Wooldridge M: Intelligent agents, Springer-Verlag, New York, 1998, pp. 3-28
23. J.R. Koza et al, “Genetic programming IV. Routine human-competitive machine intelligence,” Kluwer Academic Publishers, New York 2003
24. K. Leyton-Brown, Y. Shoham, “Multiagent systems: algorithmic, game-theoretic, and logical foundations,” Cambridge University Press 2008
25. H.X. Li, R. Chosler, “Application of multilayered multi-agent data mining architecture to bank domain,” In: Proc. the Int. Conference on Wireless Communications, Networking and Mobile Computing, 2007 pp. 6721 -6724
26. S. Mardani, M.K. Akbari, S. Sharifian, “Fraud detection in process aware information systems using MapReduce,” In: Proc. on Information and Knowledge Technology, 2014, pp.88-91
27. N. Marz, J. Warren J., “Big data - Principles and best practices of scalable realtime data systems,” 2014
28. D.E. O'Leary, “Artificial intelligence and big data,” IEEE Intelligent Systems, 2013, Vol. 28, 96-99
29. D.A. Ostrowski, “MapReduce design patterns for social networking analysis,” In: Proc. Int. Conf. on Semantic Computing, 2014, pp.316-319
30. M.L. Raymer, W.F. Punch, E.D. Goodman, L.A. Kuhn, “Genetic programming for improved data mining: application to the biochemistry of protein interactions,” In: Proc. of the 1st Conf. on Genetic Programming. MIT Press, Cambridge, MA, USA, 1996, 375-380