High Performance Computing Techniques in "Information to Knowledge" Process

Dror BEN-AMI
Zefat Academic College, Zefat, Israel;
Moldova State University
E-mail: dbenami84@gmail.com

Abstract - Variety of *Data Mining* (DM) and *Artificial Intelligence* (AI) algorithms and software techniques are considered to be advanced, progressive and sophisticated toolkits. These packages support industries and R&D institutes with solutions of „*how to transform information to knowledge*" in efficient ways. A lot of these algorithms are developed in high programming languages, such as: C++, C#, ADA and JAVA. The usage of JAVA becomes more and more popular as the „*ultimate*" programming language in the last years.

Is JAVA the ultimate software language to implement DM and *Information to Knowledge* (I2K) processes? The article examines reasons for using JAVA for DM developments and algorithms implementations for I2K process. The article mainly concentrates on „*time-consuming*" JAVA-based applications. Some of the popular paradigms and concepts are broken-down in this scientific research. The article plots perceptual reasonable conclusions, which destabilize the „*High-Performance-Computing*" (HPC) understanding on one hand, and gives better understanding on JAVA's attributes and characteristics as progressive software tool. Therefore this article is significant breakthrough in the merged and unified area of JAVA-based I2K solutions by DM implementations. The article is based on continuous and deep scientific research of DM algorithms. The article considers the parameters' set for efficient and inefficient application. Afterwards the article deploys possible reasons for the described phenomenon and finally supports the readers with some reasonable answers and conclusions.

Keywords - Artificial Intelligence (AI), Data Mining (DM), Information to Knowledge (IK), High Performance Computing (HPC), Knowledge Management (KM).

I. Introduction

High Performance Computing (HPC) paradigms emerged significantly during the last years. Various types of problems require efficient techniques and methods to solve and support organizations with Knowledge Problems and complicated algorithms in Biology, Physics, Astrophysics, Chemistry, Mathematics and on, requires efficient ways to implement these algorithms. The request for „*efficiency*" becomes much more relevant when we concentrate on *Artificial Intelligence* (AI) and *Data Mining* (DM) research fields. Wide range of problems require weeks, months and even years on mini and supercomputers till we get the „*final*" research results! So far, JAVA programming language is mostly used to implement and support practical solutions for HPC. What are the main advantages and disadvantages of JAVA as progressive programming language? Are there any ways to leverage and upgrade the JAVA utilization significantly, especially in these HPC implementations? The research shows that sometimes 25 to 33 percents of the total run-time can be saved! This article confronts with the critical success factors, and explores the reasons and ways to increase the „*efficiency*". It is based on few years of research. One main and critical aspect has to be considered: the world's population is „*bombarded*" by floods of mainly-Web information, which are changed dynamically every day, every minute, and every second. The need to transform these Web amounts of „*Information*" into „*Knowledge*" requires automatic, intelligent robust software tools and sophisticated models, concepts and apparatuses to activate the complex-process on the best efficient ways. Figure 1 [1] describes the data hierarchy and the basic relations between *data, information* & *knowledge*.

This article would not pretend to get into the wide and specific declarations of the term efficiency. On its simple and intuitive meaning, we can assume *that „Efficiency means a level of performance that describes a process that uses the lowest amount of inputs to create the greatest amount of outputs"*.

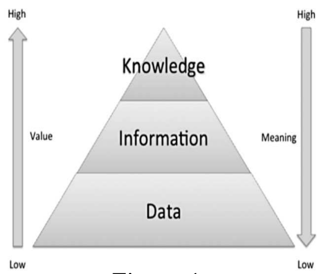

Figure 1

In most of the times we are interested and require much wide detailed way, describing these relations, as shown on Figure 2 [2].

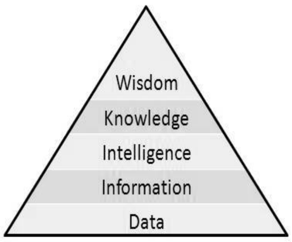

Figure 2.

This scheme is an adapted from Barabba, Vincent P. model [3]. The article, which is based on few years of research, describes, examines and concentrates on JAVA as advanced software computer language; and in its significant central place in the entire development techniques. It specially focuses on techniques, which support organizations with knowledge, on one hand.

On the second hand, as part of the research, it explores, suggests, proposes and plots better performances concepts and practical ways for using the I2K (*Information to Knowledge*) tools-set. Thus this research renews, empowers and trying to establish HPC with high level standards in this practical and empirical research field.

The article suggests new different point of view in JAVA-based I2K algorithms in their technical environments. Specifically the research deals with 2 issues:

(1) Plots advanced and progressive practical HPC methods and techniques to improve and optimize current use of the DM algorithms;
(2) Offers different and advanced standards to activate these algorithms in the wide range of technical environments.

The main goal is to create conceptual and practical ways for using I2K algorithms and focus these. The basic methodology, which is presented, describes first the basic facts about JAVA as advanced programming language. Afterwards it tries to cope and examines the suitability of JAVA to be HPC ultimate programming language for I2K algorithms and/or applications. Therefore this article is significant breakthrough in the merged and unified area of JAVA-based I2K solutions by practical implementations. In order to get much specific point of view about the proposed assumptions and solutions, we are required to understand well the data-mining & IK general stages. This is illustrated in the Figure 3 [4].

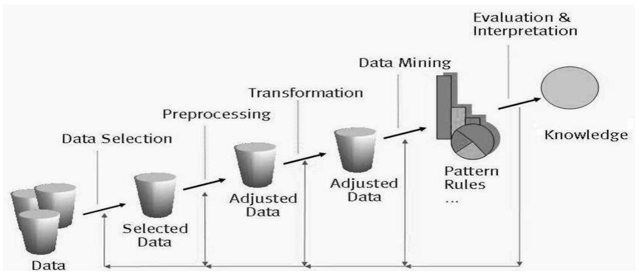

Figure 3. „*Information to Knowledge*”

II. Knowing Java

„*How many JAVA developers are there in the world?*” Oracle says its 9,000,000 [7].

Even if it less, this significant estimations raises the question „why JAVA is so attractive and popular programming language?”.

There are a lot of reasons for that. Java is high-level programming language, general purpose and wide-spectrum imperative language. It is class-based, static-typed, and provides concurrency options. Java is *object-oriented programming* (OOP) language [5], which emphasize most of its advantage [6]. In addition JAVA is reflective, generic and multi-paradigm language and as cross-platform (multi-platform) language, it is easily understood why it is so convenient to developers to implement their applications. More than that; JAVA „*admires*” would claim that java language is:

(1) Simple,
(2) Support the developers' robust and secure environment,
(3) Architecture-neutral and portable,
(4) High performance programming language and
(5) Interpreted, threaded and dynamic language.

These 5 principles summarize briefly the advantages for using JAVA.

So far, it seems that java is the ultimate programming language. If so, can we argue or assume that java is the „*ultimate*” programming language *for I2K applications, implementations and for I2K algorithms' development*, as well?

The following section faces with this question/issue and examines it, trying to summarize and understand well what is needed to plan, create an advanced JAVA-based I2K information system.

III. Discussion

There is no doubt (and can easily be proved), noticed and realized that JAVA has been used for a wide range of I2K applications indeed. In order to support us with „*real*” scientific analyzed results for this question/assumption we are required to understand and clarify the I2K role, and what are the characteristics and attributes of the I2K algorithms, specifically. Just after knowing that, we would be able to measure and test the suitability of the JAVA language for I2K algorithms' developments. Another important background, which would help us, is the need to understand and clarify the terms „*effectiveness*” and „*efficiency*”. These points are mentioned in separated detailed article.

IV. The Research

The research thesis objectives correlate two basic dimensions: theoretical and practical, as follows.

- To diagnose current status and performances of AI/DM algorithms, which were implemented in/under HPC standards.
- To propose different POV (point of view) for better AI/DM HPC solutions.
- To prove feasibility and practicality of using new improved AI/DM example algorithms; implemented by adopting better approach in HPC.
- To examine and analyze the reasons for implementing ineffective or not properly utilized programming principles in AI/DM solutions.
- Practically to develop and elaborate one/two example/s in the AI/DM algorithms.
- To propose, crystallize and establish expanded HPC standards' set.
- To assess the potential benefits of using JAVA as a language for HPC in general, and specifically in the AI/DM.

Practically, two specific JAVA–based implemented algorithms are mentioned as examples of the research.

The two code segments relate to:

(1) pre-process computations in I2K environment;
(2) finding whether network has „*open-route*” (based on I2K aggregated information).

The main and critical parameters list, which built the „*efficiency*” list is described in Table 1.

TABLE 1: THE LIST OF MAIN PARAMETERS WHICH IDENTIFY „*EFFICIENCY*”

	Parameter / Factor	Component	Comments
1	Time consuming	Processor	minutes / hours / days / months
2	Memory used	RAM	Kilo Bytes, Mega Bytes
3	# of parallel processes	CPU	
4	Using Threads	CPU / RAM / OS	Yes / No / How many
5	Operating System page faults (PF)	OS	
6	Identify pre-process procedures	Processor, I/O	When possible
7	Identifying parameters range	Input / Output	Values / Limits

It is important to emphasize that the research examines and offers software-based solutions rather than „*simple*” hardware-based solutions.

V. The Results

The entire research has examined wide range of algorithms in AI and DM.

It proves practically that *up to 33% from the total run-time can be saved* after performing and changing critical sub-sections and interventions. As we notice above, there are 4-main stages/points for intervention: beginning with the:

(1) ***Data-selection***, through
(2) ***Preprocessing*** and
(3) ***Transformation***, completed by
(4) ***DM specific algorithm***.
(5) There is no doubt that this time saves a lot of money, and the phrase „*Time is Money*” becomes very clear and obvious. Wide range of parameters was considered in this research. All of them together build the *weighted efficiency and utilization result* of the examined process.

My basic assumption argues that a lot of I2K algorithms, which have been developed by JAVA programming language are NOT so effective. Most of these JAVA-based I2K-algorithms stand in the criteria set of „*feasibility*”, „*practicality*”, „*possible solution*” standards, BUT some of them are far enough from being „*effective*”, optimized solutions or „*best-fits*” for their purposes. And here comes the BIG question – WHY?

Why a lot of I2K JAVA-based solutions are NOT so efficient?

In other „*soft*” words: are there any simple stages or sub-procedures which can raise significantly the performances' level? The readers might say „*so-what*?”. Isn't it clear that NOT ALL the algorithms are „*optimized or best-fits*” for their purposes? In other words, why is it so important to specify or explore that?

The answer for that is quiet simple! I2K-apps are mostly time-consuming applications. There are a lot of applications from different types of industries and research fields, which require a lot of the computer/s resources and inspire to have effective and optimized solution. It is critical in a lot of the cases and the best executions are required. These get twice as much validity and legacy in *very large* data-bases systems, which holds, aggregate and examine millions of millions information and data-fields figures, images and/or textual elements, i.e.: biological systems, astrophysics apps, wide medical research fields, mathematics-based apps and etc.

VI. Conclusions

Over the proven fact that I2K algorithms can be significantly improved by using and implementing specific acts, empirical data of the people' involved in the implementation process has been the DM or AI solutions for the I2K solutions. The FIRST is „*mathematicians*" that are familiar with the DM/AI algorithms and know well the usage and the suitability of each algorithms and „*when*" to use specific algorithm rather than different one. The SECOND professional staff is base on „*programmers*". More specifically – „*JAVA Programmers*". After the research it is quite clear that in order to generate optimized efficient solutions, you need to know the DM/AI algorithms very well, BUT to understand deeply the JAVA specs and usages, in order to write efficient code. The difference between „*good*" and „*best*" take place here, and may affect the final results significantly. The integration between the two areas requires YEARS of spatiality, expertise, delving into and exploration the „*two sides of the coin*". Long and continuous training and preparations are required to procurement the wide necessary knowledge. „*Thinking like a computer scientist means more than being able to program a computer.* ***It requires thinking at multiple levels of abstraction***" [8].

This phrase emphasizes the difference between being able to plan, design, program, debug and all the other activities, which are required from „*conventional*" programmer. Being able to program heavy duty sophisticated and efficient program means you need to be more than „*conventional*" programmer. Thus, a lot of knowledge, experience and very high level of making integrations between multi-disciplinary subjects and/or R&D applications are required. All these have to be integrated with the abilities of being able to activate high levels of abstraction, besides the pure analytical programming. This combination would probably ensure HPC, not just for DM and AI applications.

Thus, as a summation I would say that as far as separate professional people would develop these I2K systems, it may affect negatively the performance level of the entire process. As far as one integrated expertise will cope with the two areas and „*build*" him to work on optimal solution, we can positively assume that much higher efficient results should be accepted.

Practically I would propose that specific training routes would be opened in the universities, R&D institutes and colleges, to collaborate the knowledge between „*math*" and „*pure programming*". The implications of that conclusion would be discussed separately in the next article. These routes require years of experience and has its own wide meanings.

References

[1] D. Cheffey, S. Wood. Business & Economics, Financial Times Prentice Hall (2005).

[2] S. H. Haeckel, Presentation of the information steering group. Cambridge, MA. Marketing science institute (1987).

[3] U. Lammel, Artificial Neural Networks and Data Mining, Hochschule Wismar www.wi.hs-wismar.de/~lammel Wismar Business School.

[4] http://en.wikipedia.org/wiki/Object-oriented programming

[5] http://codebetter.com/raymondlewallen/2005/07/19/4-major-principles-of-object-oriented-programming/

[6] Potter, Priit, How many Java developers are there in the world? July 18, 2012, http://plumbr.eu/blog/how-many-java-developers-in-the-world

[7] "1.2 Design Goals of the Java^{TM} Programming Language",http://java.sun.com/docs/white/langenv/Intro.doc2.html(1999).

[8] Jeannette M. Wing, Communications of the ACM. March, 2006/Vol. 49, No. 3. https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf