Review of Literature

1. **Basem Almadani and Abdullah Rashed, (2015)**, “Enforce a Reliable Environment in Parallel Computing Applications”, a paper published in Procedia Computer Science. ISSN: 1877-0509. The authors discuss how parallel computing may be used in various applications in a faster way. In this paper, a model is submitted for Data Distribution Service (DDS) which is developed by object management group using quality of service parameters.

2. **Jan Kolek and Roman Jasek (2015)**, “A time performance evaluation of SOMA asynchronous parallel distribution in Java and C#”, a paper published in Procedia Engineering. ISSN 1877-7058. The author compares the implementation of Self-Organizing Migrating Algorithm (SOMA) for Java and C# in terms of time efficiency. SOMA is an effective tool to evaluate the performance of individual algorithm.

3. **Joel C. Adams, et al (2015)**, “TSGL: A Thread Safe Graphic Library for visualizing Parallelism”, a paper published in Procedia Computer Science. ISSN 1877-0509. This paper dwells on the need for writing multithreaded parallel programs and discusses a C++ library TSGL which allows threads to be used in shared memory space. The author has presented three applications to showcase working of TSGL and how it speeds up the program execution.


5. **Tomy Paul, and Sheena Mathews (2015)**, “Parallel for loop and Parallel Reduction: an SMP comparison of four languages”, a paper published in Procedia Computer Science.ISSN : 1877-0509. This paper discusses task parallelism by parallel for loop and parallel reduction. The author compares OpenMP, Cilk plus, UPC and chapel languages and finds that OpenMP gives the best results.

6. **Kari Kahkonen et al (2013)**, “LCT: A Parallel Distributed Testing tool for multithreaded Java programs”, a paper published in Electronic Notes in Theoretical Computer Science. ISSN: 1571-0661. This paper discusses about LIME concolic Tester (LCT) which is a testing tool for sequential and multithreaded parallel programs. The author has described architecture of LCT and other features.

7. **Herbert Kuchen and Steffen Ernsting, (2012)**, “Data Parallel Skeleton in Java”, a paper published in Procedia Computer Science. ISSN: 1877-0509. The author discusses about skeleton programs available in Java for parallelism particularly Muesli and makes a comparison of same with c++ implementation.

9. Alessandro Ricci, et al (2011), “simA: An agent-oriented approach for programming concurrent applications on the top of Java”, a paper published in Science of Computer Programming. ISSN: 0167-6423. Since concurrency and concurrent programming is becoming essential, the object oriented programming languages are being extended. In this paper the authors describe the application programming interface (API) provided to Java Programming for developing simA applications. They explain the features of this approach from software engineering point of view.

10. Rafael Durate, etal (2011), “Introducing Concurrency in sequential Java via laws”, a paper published in Information Processing Letters. ISSN: 0020-0190. The author starts with multi core processors and conveys to us about how concurrency can be introduced by using algebraic laws. They have discussed five such laws and call these laws parallelization laws.

11. Yue Qi et al (2011), “Dynamic Management of Hardware multi-threading for Network Processors”, a paper published in Procedia Engineering. ISSN: 1877-7058. This paper presents a dynamic management mechanism of hardware multithreading for pipeline architecture. It dwells upon the architecture for multithreading hardware as well as implementation of special instruction.


13. Borys J. Brandel and Tarek S Abdelrehman(2009), “A study of Potential Parallelism among traces in java Programs”, a paper published in Science of Computer Programming. ISSN: 0167-6423. This paper discusses about parallelism in traces which are paths of execution of the program. It studies the amount of trace level parallelism in Java programs. Further it counts on the effect of number of processors, trace window size and cost on performance along with analysis of dependencies and how they connect with the parallelism.

require any programme annotation. It transforms multiple, non-adjacent program statements.

15. **Eric Tanter et al (2008)**, “Flexible meta programming and AOP in Java”, a paper published in Science of Computer Programming. ISSN: 0167-6423. This paper presents a software tool, Reflex, used in aspect oriented programming (AOP) in Java and discusses modularization and adaptability. The author concludes that Reflex provides structural and behavioural facilities.


17. **Lingampalli R, et al (2007)**, “A multipurpose code coverage tool for Java”, a paper published in IEEE Xplore Digital Library. ISSN: 1530-1605. This paper presents a Java based tool, JavaCodeCoverage for test coverage reporting. Open source database support of the tool makes it very useful for software testing. It maintains the test coverage information for a set of test on individual as well as test suite.

18. **George C Wells (2006)**, “New and Improved: Linda in Java”, a paper published in Science of Computer Programming. ISSN: 0167-6423. This paper elaborates about how Linda, coordination language to achieve parallelism has been reinvented with some new features particularly for Java. It illustrates the power of new Linda extension citing various examples.

19. **Felea V and Toursel B, (2004)**, “Adaptive distributed execution of Java applications”, a paper published in IEEE Xplore Digital Library. ISSN: 1066-6192. This paper presents adaptive distributed applications in Java (ADAJ) environment with the aim to express parallelism. Using this, the developer can easily activate processing in MIMD model. At execution, ADAJ allows to gather information on processing behaviour in order to dynamically redistribute the load.

20. **Linebarger J and Kessler G, (2004)**, “Concurrency Control mechanism for closely coupled collaboration in multithreaded peer-to-peer virtual environment”, a paper published in IEEE Xplore Digital Library. ISSN: 1054-7460. This paper presents three concurrent control mechanisms. One mechanism is semi-optimistic, other two are pessimistic. They are designed for peer-to-peer virtual environment in which several threads have access to shared graph. These algorithms are straightforward and relatively simple which can be implemented in C++ or Java.

various features of array handling, stack and queues. Further it tells about how dates and Mathematics related methods can be implemented in Java applications.

22. **Wei Du and Agrawal G (2003)**, “Compiler supported coarse-grained pipelined parallelism: Why and How”, a paper published in IEEE Xplore Digital Library. ISSN: 1530-2075. This paper reports on compilation system developed to exploit coarse grained pipelined parallelism. The compiler selects set of filters, performs decomposition and generates a code in which each filter unpacks the data received from buffer.

23. **Huo Yan Chen (2002)**, “Race condition and concurrency safety of multithreaded object-oriented programming in Java”, a paper published in IEEE Xplore Digital Library. ISSN: 1062-922X. This paper primarily finds a new function to the synchronized mechanism of Java. The software safety requires the data consistence in the software. In multithreaded object-oriented programming, the coherency problem, which is also known as race condition may destroy the data consistence. Also the authors propose new approach for detecting deadlock in multithreaded programs.


25. **Soares S and Borba P (2002)**, “Concurrency control with Java and relational databases”, a paper published in IEEE Xplore Digital Library. ISSN: 0730-3157. The complexity for implementing and testing web based system is high. This paper defines the guidelines for concurrency control relational databases. It is shown where concurrency control mechanism should be used. Also the authors analyze the performance of different concurrency control approaches.

26. **Watcharawitch P and Moore S, (2002)**, “JMA: the Java multithreading architecture for embedded processors”, a paper published in IEEE Xplore Digital Library. ISSN: 1063-6404. Embedded processors are employed in applications needing high performance. Parallelism can provide this at architecture level. Alternatively, parallelism may be employed at thread level too. This paper presents an architectural study of JMA, which is high performance multithreaded architecture that supports Java.

27. **Watanbe K, et al (2001)**, “Exploiting Java instruction/thread level parallelism with horizontal multithreading”, a paper published in IEEE Xplore Digital library. ISSN: 1530-0927. The paper discusses Java processor like picoJaval and picoJavall which are pipelined processors with no instruction level parallelism (ILP) and thread level parallelism (TLP). It further points the ways to counter these limitations by using microprocessor architecture for Java computing. In this paper, the author demonstrates how this can be achieved.
28. **Xu J. and Sha E.H.M. (2001)**, “Implementing Parallelism and Scheduling data flow graphs on Java Virtual Machine”, a paper published in IEEE Xplore Digital Library. ISSN: 1520-6149. This paper presents a scheme for exploring parallelism on JVM. It is an algorithm called dynamic-duplication scheduling and code generation for data flow on parallel JVM. The algorithm improves the scheduling in comparison with traditional JVM.


30. **Rosenstiel W (2000)**, “Embedded Java”, a paper published in IEEE Xplore Digital Library. ISSN: 1080-1820. Since the complexity of embedded systems increased, it results in the growth of software also particularly in object oriented programming languages like Java and C++. Java plays an important role in context of embedded systems specially platform independence and multithreading parallelism, concurrency etc are important advantages. This paper presents various approaches of Java as well as their roles, contributions and impacts.

31. **Radhakrishnan R, et al (2000)**, “Allowing for ILP in embedded Java processor”, a paper published in IEEE Xplore Digital Library. ISSN: 1063-6897. In this paper, the authors investigate micro-architectural techniques to exploit parallelism in Java byte code streams. The authors propose the use of fill unit that stores decoded byte code into a decoded byte code cache. Secondly a stack disambiguation technique is proposed that allows elimination of false dependencies.

32. **Lan Jin et al (1999)**, “Introducing Java into the teaching of distributed systems”, a paper published in IEEE Xplore Digital Library. ISSN: 0190-5848. The authors introduce Java language, object oriented paradigm, clean and strong support and features like multithreading, concurrent programming etc. They have demonstrated this by developing a banking software on multiprocessor based system.

33. **Nugroho L E and Sajeev A S M (1999)**, “Java4P: Java with high-level concurrency constructs”, a paper published in IEEE Xplore Digital Library. ISSN: 1087-4089. Though Java has a mechanism for concurrent programming, it is complex and has limitations. The authors implement Java4P which is an extension of Java language that offers simple concurrency model and overcomes Java limitations. Hera threads creation is made implicit and they are not associated with thread objects.

34. **Peng Wu and Pauda D (1999)**, “Containers on parallelism of general purpose Java Programs”, a paper published in IEEE Xplore Digital Library. ISSN: 1089-796X. This paper presents a strategy for automatic parallelization of general purpose program
which is called tread-level data speculation (TLDS). It should be combined with strong compiler analysis. The authors propose compiler techniques to implement this approach.


36. **Chen MK and Olukotum K(1998)**, “Exploring method-level parallelism in single threaded Java Programs”, a paper published in IEEE Xplore Digital Library. ISSN: 1089-795X. This paper shows why JVM is an effective tool for exploiting method level parallelism with demonstration of speedup single threaded general purpose Java programs. Results show that speedup can be achieved with minimal programmer and compiler efforts.

37. **Chun-Mok Chung and Shin-Dug Kim, (1998)**, “A dual threaded Java processor for Java multithreading”, a paper published in IEEE Xplore Digital Library. ISSN: 1521-9097. This paper introduces a new Java processor called simultaneous multithreaded (SMT) Java chip to enhance hardware support of Java multithreading. The performance of Java chip is evaluated using a Java simulator JavaSim. Performance results show that SMT Java chip can speed up the execution of Java programs significantly.

38. **Nisholas Carriers and David Gelernter(1992)**, “How to write parallel programs: A first course”, a book published by Massachusetts Institute of Technology (MIT). ISBN: 0-262-03171-X. This book is an interesting reading. I have read chapter 1, Introduction and chapter 2, Basic Paradigms of parallelism. It gives insight on parallelism concepts, the languages used i.e computing and coordination languages and they get glued. In chapter 2, the authors discuss about paradigm and methods associated with the parallelism. All in all it is a wonderful book for those who wish to write programs using parallelism.