Annual Report for Period: 09/2011 - 08/2012 **Submitted on:** 06/03/2012

Principal Investigator: Yang, Jun . **Award ID:** 0916027

Organization: Duke University

Submitted By:

Yang, Jun - Principal Investigator

Title:

III: Small: RIOT: Statistical Computing with Efficient, Transparent I/O

Project Participants

Senior Personnel

Name: Yang, Jun

Worked for more than 160 Hours: Yes

Contribution to Project:

Jun Yang has been serving as the faculty lead on this project.

Post-doc

Graduate Student

Name: Zhang, Yi

Worked for more than 160 Hours: Yes

Contribution to Project:

Since the project's conception Yi Zhang had been the main student contributor. He was involved in most of the research and development efforts, including the RIOT-DB system (CIDR 2009) and the next generation RIOT system (ICDE 2010, PVLDB 2011, PVLDB 2012). Yi graduated with PhD in spring 2012 and is now working at Google.

Name: Thonangi, Risi

Worked for more than 160 Hours: Yes

Contribution to Project:

Risi Thonangi has been researching how to leverage the recently emerging SSDs (solid state drives) to improve performance. He has developed an index for SSDs that support efficient concurrent accesses despite the need for batch index organization. He is currently focusing on how to exploit SSD in RIOT workloads, in particular permutation, which arises in array layout conversions and matrix transposition.

Name: Herodotou, Herodotos

Worked for more than 160 Hours: Yes

Contribution to Project:

Although not funded by this project, Herodotos Herodotou contributed to the development of the RIOT-DB system (CIDR 2009).

Name: Huang, Botong

Worked for more than 160 Hours: Yes

Contribution to Project:

Botong Huang is investigating how to run RIOT on data-parallel cloud computing platforms such as Hadoop MapReduce. He has experimented with various implementations of large dense matrix multiply on Hadoop, and helped to build a multi-input and dynamic load balancing extension to Hadoop. He is now working on automatic optimization and provision of

RIOT workloads on commercial clouds such as Amazon EC2.

Undergraduate Student

Name: Zhang, Weiping

Worked for more than 160 Hours: Yes

Contribution to Project:

Shortly after the conception of the project, Weiping Zhang joined the RIOT team and worked for an REU-funded research internship and an independent study course (Spring 2010). He contributed to the development of RIOT's native array storage manager and is a co-author on of ICDE 2010 demo paper.

Name: Yan, Jiaqi

Worked for more than 160 Hours: Yes

Contribution to Project:

Jiaqi Yan joined the RIOT team in the summer of 2010 on an REU-funded research internship. He studied several information retrieval applications with scalability demands to see how RIOT might help. He is currently working on extending PostgreSQL with better support for matrix operations and leverage of modern GPU hardware.

Technician, Programmer

Other Participant

Research Experience for Undergraduates

Organizational Partners

HP Labs

HP Labs provided additional funding to RIOT in Year 2 through their Innovation Research Program. The PI visited HP Labs in Beijing in July 2010 to help jump-start the collaboration. The collaboration focused on the issues of database extensibility and parallelization, at both GPU and cloud level. See the section of this report on research and education activities for details.

Other Collaborators or Contacts

Huan Feng (intern at HP Labs China), Meichun Hsu (HP Labs Palo Alto), Xing-Xing Ju (intern at HP Labs China), Keyan Liu (HP Labs China), Kamesh Munagala (Duke University), Lei Wang (HP Labs China), Min Wang (HP Labs China).

Activities and Findings

Research and Education Activities:

Recent technological advances have enabled collection of massive amounts of data in science, commerce, and society. These large datasets have brought us closer than ever before to solving important problems such as decoding human genomes and coping with climate changes. Meanwhile, the exponential growth in the amount of data has created an urgent challenge. Today, much of advanced analysis is done with programs custom-developed by statisticians. Unfortunately, progress has been hindered by the lack of easy-to-use statistical computing environments that scale to large datasets. Many existing tools assume that datasets fit in main memory; when applied to large datasets, they are unacceptably slow because of excessive disk input/output (I/O) operations. There have been many approaches toward I/O-efficiency, but none has gained traction with the statistical computing community. Disk-based storage engines and I/O-efficient function libraries provide only a partial solution, because many sources of I/O-inefficiency in a program remain at a higher, inter-operation level: e.g., how large intermediate results are passed between operations, how much performance can be gained by deferring and reordering operations, etc. Database systems seem to be a natural solution, with I/O-efficiency and a high-level language (SQL) enabling many high-level optimizations. However, work in integrating databases and statistical computing has mostly remained database-centric, forcing statisticians to learn unfamiliar languages and deal with their impedance mismatch with the host language.

To make a practical impact on the statistical computing community, the RIOT project seeks to extend R---an open-source statistical computing environment widely used by statisticians---to transparently provide scalability over large datasets. Transparency means no SQL, or any new language to learn. Transparency means that existing code should run without modification, and automatically gain efficiency. RIOT is developing an end-to-end solution that addresses issues on all fronts: I/O-efficient and parallel algorithms, deferred evaluation, pipelined execution, cost-driven optimization, smart storage and materialization options, and seamless integration with database systems and the interpreted host language.

(In the third year of the project, we expanded into several focused problems, including building a better storage engine and designing better execution and optimization frameworks for RIOT, extending database systems to support matrix storage and computation, parallelization using GPU and cloud, and leveraging SSDs for better performance. A detailed description of our contributions can be found below in our 2010-2011 project report.)

In Year 3 of the project, we have investigated the following specific research problems.

A) RIOT Execution and Optimization Issues. Database queries exhibit a limited number of fairly simple data access patterns. However, linear algebra operations common in statistical computing give rise to a much larger space of complex access patterns typically expressed using nested loops. We have developed a framework that allows these access patterns to be captured---either through code analysis or user specification---in a form amenable to automatic optimization.

Specifically, the first problem that we tackled with this framework is exploiting I/O sharing opportunities, published in PVLDB 2012. Most analysis tasks consist of multiple steps, each making one or multiple passes over arrays to be analyzed and generating intermediate results. Existing database techniques fall short of solving the I/O sharing problem in this setting. A database-like, operator-based approach does not allow full-fledged inter-operator optimization: when putting operators together for co-optimization, they cannot be treated as black boxes but need to be 'opened up' so that the optimizer can tweak their inner workings further. Our framework captures a broad range of analysis tasks expressible in nested-loop forms, represents them in a declarative way, and optimizes their I/O by identifying sharing opportunities. Experiment results show that our optimizer is capable of finding execution plans that exploit nontrivial I/O sharing opportunities with significant savings.

The next problem we consider is the joint optimization of I/O and data layouts. Our PVLDB 2012 paper did not consider array layouts as one dimension of its optimization space; instead, array layouts were assumed to be predetermined by the programmer or some other software component. However, the choice of array layouts can dramatically affect a program's overall I/O performance, and thus needs to be determined in a cost-based manner. Furthermore, the choice of array layout may influence how to share I/Os optimally. Therefore, we consider how layout choices can be co-optimized with I/O sharing to yield further I/O savings. We focus on the commonly used block-based layout (which subsumes column- and row-major layouts). This work has been completed and we are preparing it for submission.

B) Parallelization with Cloud. In Year 2, we studied how to support statistical computing workloads in Hadoop, an open-source implementation of MapReduce popular for data-parallel cloud computing. Our approach to overcoming the limitations of Hadoop there was to extend Hadoop's programming model and execution framework to support multiple inputs and better load balancing. This work's submission has been delayed, because the lead student is no longer working on the project. We hope to wrap up and submit this work in Year 4.

Recognizing that a modified Hadoop would be difficult to gain adoption, in Year 3, we have focused on the alternative of working within the current Hadoop, and automatically optimizing and provisioning for RIOT workloads on pay-as-you-call cloud services. The cost of acquiring and maintaining a dedicated high-end cluster is prohibitive for most organizations. Fortunately, cloud computing has made it possible to rent a commodity cluster on demand. However, optimizing and provisioning for RIOT workloads is a daunting task for most users, as the space of choices is intricate and overwhelming---given a cloud, one must choose from multiple machine types (each with a different price) and any number of machines; given a cluster, one must select the right job plan and configuration for each MapReduce (e.g., the partitioning factor for each matrix and the number of concurrent slots to run in each machine). To make cloud easier to use, we study automatic optimization and provisioning techniques that can suggest good execution and provisioning plans to users together with the estimated completion time and cost. With this optimizer, we can answer questions like: 'what is the minimum budget required to execute the workflow by a given deadline?' and 'what cluster should be used at different stages of the workflow?' This work is under way and we are close to finishing a submission.

C) Extending Database Systems (DBMS) and Parallelization with GPU. In collaboration with HP Labs, we have made significant progress towards demonstrating that, with careful design and engineering, we can overcome some purported disadvantages of DBMS in storing and computing with matrices. Specifically, we reduce the overhead of element-oriented storage and iterator-based execution by storing and processing matrices in the unit of 'chunks' (submatrices). We work around the inefficiency of DBMS execution engine for numerical computation by invoking highly optimized numerical libraries (e.g., BLAS) on a per-chunk basis. We avoid the awkwardness of SQL in expressing linear algebra operations with user-defined functions that operate on entire matrices. In Year 2, we developed and evaluated alternative matrix linearization and chunking schemes, and showed superior flexibility and performance than popular approaches. We also considered different strategies for implementing matrix operations within DBMS (specifically, PostgreSQL), and studied their trade-offs. In Year 3, we completed the study of a more complex operation, QR factorization. Overall, we have demonstrated that the combination of chunking, user-defined functions, and use of optimized libraries makes highly efficient matrix computation possible within a DBMS. We also showed GPU acceleration to be effective within DBMS. On the other hand, the reduction in computation cost now makes the overhead of getting data through the database significant, and we are revisiting the approach of exporting data out of the database before analysis. Our initial findings were summarized in a paper titled 'MaSSA (Massive-Scale Statistical Analysis) DBMS,' presented at HP Tech Con 2011, and a provisional patent has been filed. We are working on polishing results and preparing a submission for publication in a database research conference.

D) Working with Solid-State Drives (SSDs). Solid-state drives are becoming a viable alternative to magnetic disks for many workloads. We have been studying how to use them effectively for various database and linear algebra workloads. The first problem, which we began to investigate in Year 1 and completed in Year 2, is indexing. We designed, implemented, and evaluated an index structure called FD+tree and an associated concurrency control scheme called FD+FC. A paper describing our results is under submission.

In Year 3, we have focused on two other problems---sorting and permuting on SSDs. A main application of these problems in RIOT is data layout conversion. Conventional I/O-efficient algorithms for these problems assume that disk reads and writes have equal costs, and that random accesses cost a lot more than sequential ones. However, these assumptions are not valid with SSDs. We found that the trade-off between reads and writes and fast random accesses enable interesting interesting new algorithms for permuting and sorting on SSDs. This work has almost been completed and we are preparing a submission.

Summary) The progress we have made thus far, as summarized above, are in line with our original proposal and the directions laid out in our last annual report. Because of their complexity, several research problems that we are working on currently need additional time to finish. Therefore, we have requested a grantee-approved no-cost extension of the project, with the new expiration date of August 31, 2013. In the final year of the project, we plan to complete our investigation of the problems described above, and make a public code-release for RIOT.

In terms of educational activities in Year 3, the PI has continued to closely supervise and train graduate and undergraduate researchers. Yi Zhang graduated with a PhD in spring 2012. Risi Thonangi and Botong Huang continue to make solid progress in their PhD study. Jiaqi Yan, an undergraduate researcher, graduated in spring 2012 but is still working on the project in summer 2012. The PI taught an undergraduate database course in fall 2011; in spring 2012, he introduced a new graduate course, 'Projects in Computational Journalism,' devoted to investigating the nascent research area of computational journalism, which leverages computing to help advance public-interest journalism.

Findings:

In Year 1, we invested most of our effort in prototyping and development activities, which provided us with much experience and insights. In Year 2, we expanded into several focused problems, including building a better storage engine and designing a better execution and optimization frameworks for RIOT, extending database systems to support matrix storage and computation, parallelization using GPU and cloud, and leveraging SSDs for better performance. In Year 3, we continued to make progress on these problems, and further investigated the problem of optimizing and provisioning for RIOT workloads in cloud and that of permuting and sorting data on SSDs. Published results so far include: 1) the RIOT 'vision' paper in CIDR 2009, which also covers RIOT-DB; 2) a demonstration of a prototype for the next generation of RIOT in ICDE 2010; 3) a paper on the RIOT storage engine in PVLDB 2011; and 4) a paper on optimizing I/O sharing in RIOT in PVLDB 2012. A number of other manuscripts are under either submission or preparation at this point; see activities above for details. We have released the code for RIOT-DB on the project website, and plan to make additional releases in the last year of the project.

Training and Development:

Ph.D. students: Yi Zhang, Risi Thonangi, Botong Huang, Herodotos Herodotou.

Undergraduate students: Weiping Zhang, Jiaqi Yan

These students have gained research experience in algorithms, compilers, databases, high-performance computing, and programming languages.

Outreach Activities:

Journal Publications

Yi Zhang, Kamesh Munagala, and Jun Yang, "Storing Matrices on Disk: Theory and Practice Revisited", Proceedings of the VLDB Endowment, p. 1075, vol. 4(11), (2011). Published,

Yi Zhang and Jun Yang, "Optimizing I/O for Big Array Analytics", Proceedings of the VLDB Endowment, p. 764, vol. 5(8), (2012). Published,

Books or Other One-time Publications

Yi Zhang, Herodotos Herodotou, and Jun Yang, "RIOT: I/O-Efficient Numerical Computing without SQL", (2009). Conference Paper, Published
Collection: Proceedings of the 4th Biennial Conference on Innovative Data Systems Research (CIDR '09)
Bibliography: Asilomar, California, USA, January 2009

Yi Zhang, Weiping Zhang, and Jun Yang, "I/O-Efficient Statistical Computing with RIOT", (2010). Conference Demonstration Description, Published
Collection: Proceedings of the 26th International Conference on Data Engineering (ICDE '10)
Bibliography: Los Angeles, California, USA, March 2010

Web/Internet Site

URL(s):
http://www.cs.duke.edu/dbgroup/Main/RIOT
Description:

Other Specific Products

Contributions

Contributions within Discipline:

At the end of Year 3 of the project, we have made a series of solid contributions toward enabling efficient statistical analysis over massive datasets. We have built two functional prototypes, RIOT-DB and RIOT, and published in CIDR 2009, ICDE 2010, PVLDB 2011, and PVLDB 2012. A number of other papers are currently in preparation or under submission. For detailed descriptions of these contributions, please refer to the section of this report on research and education activities.

Contributions to Other Disciplines:

The PI has been part of two other interdisciplinary projects---one funded by NSF, which studies how to collect and analyze ecological data from a sensor network, and another funded by NIH, which develops analytical and modeling tools for immunology. The PI is also starting a new project on computational journalism, in collaboration with journalists and social scientists. Some of the work in this project is motivated by the problems faced in the other projects, which will in turn benefit from our research results. As RIOT matures, it will be made available to the general statistical computing community for broader impact.

Contributions to Human Resource Development:

Ph.D. students: Yi Zhang, Risi Thonangi, Botong Huang.

Undergraduate students: Weiping Zhang, Jiaqi Yan.

Contributions to Resources for Research and Education:

The PI has integrated the research being carried out under this project into his education activities (see the section of this report on research and education activities for details). The RIOT tools are also publicly available in open source to other researcher and educators.

Contributions Beyond Science and Engineering:

Conference Proceedings

Special Requirements

Special reporting requirements: None
Change in Objectives or Scope: None
Animal, Human Subjects, Biohazards: None

Categories for which nothing is reported:

Activities and Findings: Any Outreach Activities
Any Product
Contributions: To Any Beyond Science and Engineering
Any Conference