

Research Statement

Wei Lu (welu@cs.indiana.edu) -Indiana University

My research interests lie at the intersection of *Parallel/Concurrent Programming*, *XML*, *Service Oriented Architecture*, and *Cloud Computing*. As manufacturers have encountered difficulties in continuing to exponentially increase clock speeds, they are increasingly utilizing the march of Moore's law to provide multiple cores on a single chip. While the multi-core processor is rapidly becoming the mainstream with quad-core shipping now and 16 core systems expected within two or three years, the software must be able to exploit the parallelism to take advantage of multi-core processors. **My thesis research strives to devise such approaches to help the software, particular those dealing with XML processing and Service Oriented Architecture, to utilize the multiple cores efficiently.** The overarching goal of my research is to build the appropriate programming methods which can greatly reduce the complexity of developing the large-scale system which can span computing platforms from single multi-core machine, through clusters, to Internet. In the remainder of this document, I will first describe my dissertation research, and then describe research I have one outside of my dissertation, and finally, provide my future research agenda.

Dissertation Research Recent years have witnessed the prevalence of XML and web based Service Oriented Architecture (SOA). XML processing and web services, however, are noted for the low performance and there has been an imminent need for effective solutions, particularly with the multi-core processor. My dissertation research tackles this challenge from the two pillars of SOA: XML processing and service orchestration.

Parallel XML Processing The very characteristics of XML that have led to its success, however, such as its verbose and self-descriptive nature, can incur significant performance penalties. Processing of the XML documents has been recognized as the performance bottleneck in a number of systems. Various techniques have been proposed to improve the performance of XML processing, but unfortunately few of them are able to leverage the parallelism benefits of multi-core processors.

To exploit this parallelism, I start from the most challenging task: parallel XML parsing. Since a XML document is the serialization of a tree data model, the problem can be stated as: *How do we partition the document into parse-able partitions that can be processed in parallel?* I devise a novel data-parallel XML parsing algorithm, PXP [4], which adopts a two-pass-scanning strategy. A quick sequential scanning of the document is first performed to identify the structural information of the document, and then a complete parallel parsing guided by the structural information parses the document in parallel. To our knowledge, PXP is the first parallel XML parser and experimental results show a good performance speedup on the multi-core machine.

Since more and more web-services rely on the higher level XML tasks, such as XML security, it is crucial to have a general parallel XML processing model which address the common parallel patterns. Therefore after the PXP I developed a general-purpose parallel XML processing model, ParaXML [7], which serves as the general paradigm for the potential parallelization of other XML tasks. Generally speaking, ParaXML treats the XML document as a tree structure and the XML processing as the extension of the parallel tree traversal algorithm. However the XML processing has quite distinct characteristics from the traditional discrete optimization problems, thus demanding the special fine-grained optimization. ParaXML internally adopts a lock-free work-stealing scheme to dynamically control the load balance; a novel stealing tracing approach is introduced to reduce the results. I has used ParaXML to parallelize various XML tasks, including XML searching, serialization, XML signature, parsing and DOM building. The empirical study shows that those parallel implementations present substantial speedup on the multi-core machine.

Concurrent Service Orchestration Runtime The essential goal of SOA is building large system by composing distributed services together. WS-BPEL is the *de facto* language for the workflow of web-services and it adopts the orchestration paradigm, in which a central engine orchestrates the remote services according the workflow. As WS-BPEL allows sophisticated concurrency and coordination logic in a workflow, the workflow engine, as the hot-spot of the entire system, should be able to handle the massive concurrency and the complex coordination logic as well. To implement an efficient and scalable workflow engine, the conventional thread/lock model is inappropriate either in term of the performance or in term of the expressiveness. Instead, my research strives to investigate an alternative model, namely the event-driven asynchronous programming model together with join patterns. My research has shown that most concurrency and coordination logic of WS-BPEL can be mapped to the join operators (provided in Microsoft CCR library) in a very elegant manner, meanwhile the asynchronous events makes the system extremely lightweight and scalable [8].

Besides, in pursuit of the programming productivity I further develop a service orchestration library in C#, which provides a set of higher-level type-safe API for the service orchestration. By leveraging some features in C# language this library allows users to write the efficient asynchronous concurrent workflow program without the typical “inversion of control” problem. Also thanks to the standard message-passing interface of web-services and the lightweight workflow engine, this library enables users to write a large-scale service oriented system which can efficiently scales to platforms from single multi-core CPU, to clusters of distributed computers, to the Internet.

Other Research My earlier research work strives to tackle the performance issue of the XML and web service from various perspectives. First, I co-investigate how the higher level XML schema information can be utilized to accelerate the processing of the XML document [2]. This technology is called *schema-specific parsing* and its basic idea is that the schema information can significantly decrease the size of the searching space of parsing by generating a specific parser. Secondly, I have been interested in the alternative encoding schema of the XML model. To meet the high-performance requirements of the scientific applications where the large amount of binary scientific data dominates, I have co-designed and developed *a binary XML encoding schema* for scientific applications [1] and based on that I develop *a generic SOAP engine* which supports both textual and binary XML [3]. I also have investigated the *streaming XML processing model*, especially for the XML message security processing [6] which is a well-known bottleneck in the web service protocol stack. My research has shown that the processing of XML message signature can be embedded within the XML parsing and both can be processed by an augmented automaton in a streaming way.

Besides the aforementioned work on the performance of web service and XML, I also investigate a hybrid component scheme which combines two different component schemes: the CCA parallel component architecture and Kepler workflow system [5]. In this schema we use Kepler to arrange the temporal ordering of computations as a workflow, but CCA to organize the functional units of the parallel computation.

Future Research Goals I will continue my research by following two dimensions: multi-core programming and cloud computing.

Parallel Programming on Multi-core processors First, based on my thesis research work I plan to apply the ParaXML model to more ambitious XML processing tasks, such as XPath and XSLT. The parallelization of both XPath and XSLT may involve a complex combination of data parallelism and functional parallelism. It will be interesting to extend the ParaXML model to embrace more parallel programming patterns to solve those challenges. Generally, I am interested in multi-core parallel algorithms and parallel programming methodologies (such as language-supported concurrency, transactional memory and so on), which can fully exploit multi-core processors with the less programming complexity.

Development Models for Cloud Computing The emerging data center “cloud” model of computing can have a pivotal role in addressing the problems that are not well served by the traditional supercomputer center. There are currently three approaches, namely OS virtualization (e.g., Amazon EC2), Parallel Frameworks (e.g., Google MapReduce) and Software-As-A-Service (SaaS), to providing application support for the cloud computing. As each approach is devised from a different perspective with a different concern,

each has strengths and also limitations. However, they can be combined in several significant ways. For example, a service-oriented programming model together with parallel workflow orchestration is a promising starting point. I am very interested in a systematic way to integrate these approaches together into a coherent application development model which will greatly reduce the complexity of developing reliable, highly concurrent, large-scale applications on the cloud platform.

References

- [1] K. Chiu, T. Devadithya, W. Lu, and A. Slominski. A binary xml for scientific applications,. In *e-science'05 (1st IEEE international conference for e-science and Grid Computing)*, Melbourne, Australia, Dec 2005.
- [2] K. Chiu and W. Lu. A compiler-based approach to schema-specific xml parsing. In *The First International Workshop on High Performance XML Processing, Satellite workshop of WWW2004 International Conference*, 2004.
- [3] W. Lu, K. Chiu, and D. Gannon. Building a generic soap framework over binary xml,. In *HPDC'06 (The 15th IEEE International Symposium on High Performance Distributed Computing)*, Pairs, France, June 2006.
- [4] W. Lu, K. Chiu, and Y. Pan. A parallel approach to xml parsing. In *The 7th IEEE/ACM International Conference on Grid Computing*, Barcelona, September 2006.
- [5] W. Lu, K. Chiu, S. Shirasuna, and D. Gannon. A hybrid decomposition scheme for building scientific workflows. In *HPC'07 (Proceedings of High Performance Computing Symposium, SCS Spring Simulation Multiconference*, Norfolk, VA, March 2007.
- [6] W. Lu, K. Chiu, A. Slominski, , and D. Gannon. A streaming validation model for soap digital signature. In *14th IEEE International Symposium on High Performance Distributed Computing (HPDC-14)*, July 2005.
- [7] W. Lu and D. Gannon. Parallel xml processing by work stealing. In *Workshop on Service Oriented Computing Performance In Conjunction with HPDC*, Monterey Bay California, June 2007.
- [8] W. Lu, T. Gunarathne, and D. Gannon. Developing a concurrent service orchestration engine in ccr. In *Workshop on Multicore Software Engineering (IWMSE) in conjunction with ICSE, Leipzig, Germany*, 2008.
- [9] Y. Pan, W. Lu, Y. Zhang, and K. Chiu. A static load-balancing scheme for parallel xml parsing on multicore cpus. In *CCGrid'07 (IEEE International Symposium on Cluster Computing and the Grid)*, Rio de Janeiro - Brazil, May 2007.