Chi-Keung Luk

(As of January, 2000)

Computer Science Department
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
(412) 268-7673 (Voice)
(412) 268-5576 (Fax)
luk@cs.cmu.edu
http://www.cs.cmu.edu/~luk

4716 Ellsworth Ave. Apt. 714
Pittsburgh, PA 15213
(412) 687-2622

EDUCATION

University of Toronto, Ph.D. in Computer Science, June 2000. Also as visiting scholar at Computer Science Department of **Carnegie Mellon University** (11/97 - 2/00).

Chinese University of Hong Kong, Masters of Philosophy in Computer Science, July 1993.

Chinese University of Hong Kong, Bachelor of Science in Computer Science (First Class Honors), May 1991.

RESEARCH

Research Areas: Computer Architecture, Compilers, and Programming Languages.

Ph.D. Thesis: ***Optimizing the Cache Performance of Non-Numeric Applications***.

Advisor: Professor Todd C. Mowry.

<u>Summary of Contributions</u>

As the gap between processor and memory speeds continues to widen, memory access latency is becoming an increasingly important performance bottleneck. To maximize overall performance, both *data* and *instruction* access latencies must be addressed. While caches are an essential step towards addressing this problem, they are not a panacea. In particular, effective techniques for coping with the memory latency in *non-numeric applications* have so far been lacking. My thesis makes the following primary contributions to coping with the latency of accessing data and instructions in non-numeric applications:

(i) To cope with instruction cache miss latency, I have proposed a novel prefetching technique called **cooperative instruction prefetching** [5,7] whereby the hardware and the compiler cooperate to prefetch far enough in advance without polluting the cache. Experimental results demonstrate that cooperative prefetching hides 50% of more of the latency remaining with state-of-the-art instruction prefetching schemes on a modern processor.

(ii) I have designed three **compiler-based prefetching schemes for pointer-based applications** [9]. My schemes are to date the unique of their kind in the literature. I have also implemented the most widely applicable scheme in a research compiler. Experimental results demonstrate that automatic compiler-inserted prefetching significantly improves the execution speed of pointer-based codes in both uniprocessors and large-scale shared memory multiprocessors [2]. In addition, the more sophisticated schemes can improve performance further by as much as twofold.

(iii) To reduce the runtime overheads of software-based latency tolerance techniques (e.g., software prefetching and load speculation), I have proposed a new profiling technique named **correlation profiling** [1,8] that helps predict which dynamic instances of a static memory reference will hit or miss in the cache, whereby a latency tolerance technique can be applied only to those dynamic instances that miss. Experimental results show that roughly half of the 22 non-numeric applications studied can potentially enjoy significant reductions in memory stall time by exploiting correlation profiling.

(iv) I have proposed an architectural mechanism called **memory forwarding** [6] which can *always* guarantee the safety of data relocation in C programs. With the support of memory forwarding, software can perform any *dynamic data-layout optimizations* without worrying about hurting the program correctness. I have also addressed some important issues of implementing memory forwarding in modern processors. Experimental results show that the aggressive locality optimizations enabled by memory forwarding greatly reduce both memory latencies and memory bandwidth consumption in a set of non-numeric applications, and hence offer impressive speedups: in some cases by more than twofold.

M.Phil. Thesis: ***The Design and Implementation of a Multiparadigm Programming Language***.

Advisor: Professor Kam-Wing Ng.

HONORS

Canadian Commonwealth Scholarship, 1994 - Present.

IBM Centre for Advanced Studies Fellowship, 1995-1996, 1997-1998.

Croucher Foundation Studentship, 1991 - 1993 (awarded to top two graduate students in Computer Science Department of Chinese University of Hong Kong).

Certificate of Merit of Excellent Teaching at Chinese University of Hong Kong, 1993.

PUBLICATIONS

Journal Articles:

[1] T. C. Mowry and C. -K. Luk. Understanding Why Correlation Profiling Improves the Predictability of Data Cache Misses in Non-Numeric Applications. To appear in *IEEE Transactions on Computers*.

[2] C.-K. Luk and T. C. Mowry. Automatic Compiler-Inserted Prefetching for Pointer-Based Applications. In *IEEE Transactions on Computers* (Special Issue on Cache Memory), vol. 48, no. 2, 1999.

[3] K. W. Ng and C. K. Luk. I+: A Multiparadigm Language for Object-Oriented Declarative Programming. In *Computer Languages*, vol. 21, no.2, pp. 81-100, 1995.

[4] K. W. Ng and C. K. Luk. A Survey of Languages Integrating Functional, Object-Oriented and Logic Programming. In *Journal of Systems Architecture* (formerly *Microprocessing and Microprogramming*) 41:5-36, 1995.

Submitted Journal Articles:

[5] C.-K. Luk and T. C. Mowry. Architectural and Compiler Support for Effective Instruction Prefetching: A Cooperative Approach. Submitted to *ACM Transactions on Computer Systems*.

Conference Articles:

[6] C.-K. Luk and T. C. Mowry. Memory Forwarding: Enabling Aggressive Layout Optimizations by Guaranteeing the Safety of Data Relocation. In *Proceedings of 26th International Symposium on Computer Architecture* (ISCA'99), May 1999.

[7] C.-K. Luk and T. C. Mowry. Cooperative Prefetching: Compiler and Hardware Support for Effective Instruction Prefetching in Modern Processors. In *Proceedings of 31st ACM/IEEE International Symposium on Microarchitecture* (MICRO-31), December 1998.

[8] T. C. Mowry and C.-K. Luk. Predicting Data Cache Misses in Non-Numeric Applications Through Correlation Profiling. In *Proceedings of 30th ACM/IEEE International Symposium on Microarchitecture* (MICRO-30), pages 314-320, December 1997.

[9] C.-K. Luk and T. C. Mowry. Compiler-Based Prefetching for Recursive Data Structures. In *Proceedings of the Seventh ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (ASPLOS-VII), pages 222-233, October 1996.

[10] C.-K. Luk. Memory Disambiguation for General-Purpose Applications. In *Proceedings of CASCON'95*, November 1995.

[11] K. W. Ng and C. K. Luk. An Integrated Computation Model for Parallel Programming. In *Proceedings of IEEE TENCON'93/Beijing*, October 1993.

[12] K. W. Ng and C. K. Luk. The Design of a Multiparadigm Programming Language I. In *Proceedings of EuroMicro'93* (published as a special issue in the *Journal of Systems Architecture* 37:171-174, 1993).

Technical Reports and Dissertations:

[13] C.-K. Luk. *Optimizing the Cache Performance of Non-Numeric Applications*. Ph.D. Thesis, Department of Computer Science, University of Toronto, January 2000.

[14] C.-K. Luk and T. C. Mowry. Compiler and Hardware Support for Automatic Instruction Prefetching: A Cooperative Approach. Technical Report CMU-CS-98-140, Carnegie Mellon University, June 1998.

[15] T. C. Mowry and C. -K. Luk. Predicting Data Cache Misses in Non-Numeric Applications Through Correlation Profiling. Technical Report CMU-CS-97-175, Carnegie Mellon University, September 1997.

[16] C. K. Luk. *The Design and Implementation of a Multiparadigm Programming Language*. Masters Thesis, Department of Computer Science, The Chinese University of Hong Kong, July 1993.

PRESENTATIONS

Memory Forwarding: Enabling Aggressive Layout Optimizations by Guaranteeing the Safety of Data Relocation. In ISCA’99, Atlanta, Georgia, May 1999.

Cooperative Prefetching: Compiler and Hardware Support for Effective Instruction Prefetching in Modern Processors. In MICRO-31, Dallas, Texas, December 1998.

Predicting Data Cache Misses in Non-Numeric Applications Through Correlation Profiling. In MICRO-30, Research Triangle Park, North Carolina, December 1997.

Compiler-Based Prefetching for Recursive Data Structures. In ASPLOS-VII, Cambridge, Massachusetts, October 1996 (a long version was given as an IBM/ITRC/UofT compiler seminar).

Memory Disambiguation for General-Purpose Applications. In CASCON’95, Toronto, Ontario, November 1995 (a long version was given as an IBM/ITRC/UofT compiler seminar).

PROFESSIONAL EXPERIENCE

5/95-9/95 **Summer Intern**, TOBEY Compiler Group in IBM Toronto Laboratory.
- Analyzed the performance of a data-prefetching algorithm implemented in IBM RS6000 compilers.
- Proposed new architectural support for more effective prefetching on the PowerPC.

DEVELOPMENT EXPERIENCE

MIPS R10000 Simulator: Co-developed a cycle-by-cycle simulator of the MIPS R10000 superscalar processor written in C++. In particular, I implemented a sophisticated instruction fetching unit supporting state-of-the-arts instruction fetching and prefetching techniques.

Prefetching Compiler Pass: Designed and implemented in the SUIF Compiler using C++ a fully automatic compiler algorithm for prefetching pointer-based applications.

Prototype Implementation of a New Language: Designed a new multiparadigm language called I+ and implemented a prototype of it on UNIX using C, Prolog and ML.

Programming Languages: C++, Java, Perl, Prolog, Lisp, ML, Fortran, Pascal.

TEACHING EXPERIENCE

Teaching Assistant for three semesters at the Department of Computer Science, University of Toronto.

Teaching Assistant for 1991-1993 at the Department of Computer Science, Chinese University of Hong Kong.

PROFESSIONAL ACTIVITIES

Reviewer for IEEE Transactions on Computers, Journal of Parallel and Distributed Computing, ISCA’00, HPCA’00, HPCA’99, ASPLOS’98, ISCA’98, ISCA’96, PACT’96, CASCON’96.

Association for Computing Machinery (Student Member).

REFERENCES

Available upon request.