Designing of Image Compression Memory Hierarchy for Embedded System

施嘉政、王欣平

E-mail: 9509668@mail.dyu.edu.tw

ABSTRACT

Designing of modern computers ’ micro-architecture relies on dynamic instruction traces for design optimization. However, dynamic instruction traces often generates massive data that make the traces difficult to analysis and process. This thesis propose a novel dynamic instruction traces profiling framework and a profiling algorithm that named as Melting in mining of the most frequent and longest instruction sequence. The profiling framework is exemplified by designing of memory hierarchy for JPEG image compression algorithm. The proposed profiling framework combines both the merits of traditional functional profiling and modern instruction traces schemes. The framework is divided into to two steps. The target program is first profiled using function level profiler that the most frequent function is determined. The derived function is simulated using the SimpleScalar/ARM 4.0 simulator where dynamic instruction traces is generated. As result, the amount of traces data is greatly reduced. Finally, having the traces obtained, the Melting Algorithm is applied to mine the most frequent and longest consecutive instruction sequence. The mined sequence is applied to optimized memory hierarch. The sequence can also be applied in instruction compression and other micro-architecture design issues.

Keywords : Data mining ; Cache design ; SimpleScalar ; JPEG ; ARM

Table of Contents

REFERENCES

[1] Gregory K. Wallace, “ The JPEG Still Picture Compression Standard ”, CACM, Vol.34, No.4, pp.31-44, 1991.

[2] ITU/CCITT, Recommendation T.81, Digital compression and coding of continuous-tone still images, September. 1992.

[3] K. Karuri, M. Faruque, S. Kraemer, R. Leupers, G. Ascheid, and H. Meyr, “ Fine-grained Application Source Code Profiling for ASIP Design ”, In 42nd Design Automation Conference, pp.329-334, June 2005 [4] T. Ball, “ Efficiently Counting Program Events with Support for on-line Queries ”, ACM Transactions on Programming Languages and Systems, September. 1994.

[5] T. Ball, J. R. Larus, “ Optimally Profiling and Tracing Programs ”, ACM Transactions on Programming Languages and Systems, Volume 16, Issue4, pp.1319-1360, July 1994.

[6] J. R. Larus, “ Whole Program Paths ”, Proceedings of the SIGPLAN 99 Conference on Programming Languages Design and Implementation(PLDI 99), May 1999, Atlanta Georgia.

[7] Erez Perelman, Trishul M. Chilimbi, Brad Calder, Variational Path Profiling, Proceeding of the International Conference on Parallel Architectures and Compilation Techniques(PACT), September. 2005.

[8] W.-C. Hsu, J. Lu, P.-C. Yew, D. Chen, “ Dynamic trace selection using performance monitoring hardware sampling ”, International Symposium on Code Generation and Optimization, pp.79-90, March 2003.

[9] B. Cmelik, “ SpixTools Introduction and User ’ s Manual ”, Technical Report SMLI TR-93-6, Sun Microsystems Laboratory, Mountain View, CA, February. 1993.

[10] A. Srivastava and A. Eustace, “ ATOM: A system for building customized program analysis tools ”, In ACM conference on Programming Language Design and Implementation, pp.196-205, Orlando, FL, June 1994.

[11] L. Benini, F. Menichelli, M. Olivieri, “ A class of code compression schemes for reducing power consumption in embedded microprocessor systems ”, IEEE Transactions on Computers, Volume 53, Issue 4, pp.467-482. April 2004.

[12] M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, R. B. Brown. Mibench, A free, “ commercially representative embedded benchmark suite ”, In Proceedings of the IEEE 4th Annual Workshop on Workload Characterization, 2001.

[13] Mibench Benchmark, http://www.eecs.umich.edu/mibench/.
[14] SimpleScalar Version 4.0 , http://www.simplescalar.com/ [15] T.-C. Chiueh and P. Pradhan, “ Cache memory design for network processors “ , High-Performance Computer Architecture, pp.409-418, 2000.
[16] P. Stefan, K. Dhireesha, and J. Eugene, “ Cache performance of video computation workloads ”, Digital and Computational Video, pp.169-175, 2002.
[17] Dinesh C. Suresh, Frank Vahid, Greg Stitt, Jason R. Villarreal, and Walid A. Najjar, “ Profiling tools for hardware/software partitioning of embedded applications. ” Proceedings of the 2003 ACM SIGPLAN conference on Language, compiler, and tool for embedded systems, pp.189-198, 2003.
[18] A. J. Smith, “ Cache memories ”, ACM Computing Surveys 14, No.3, pp.473-530, 1982.
[19] N. Linda and L. Jilia, “ The Essentials of Computer Organization and Architecture ”, Jones and Bartlett Publishers, Inc., 2003.
[20] D. A. Patterson and J. L. Hennessy, “ Computer Organization & Design ”, Second edition, Morgan Kaufmann Publishers, San Francisco.
[21] http://www.gnu.org/software/binutils/manual/gprof-2.9.1/html_mono/gprof.html [22] http://kprof.sourceforge.net/ [23] 楊智喬 , Xtensa可組態處理器及其應用（下）, 國家晶片系統設計中心。