

Guest Editors' Introduction— Cache Memory and Related Problems: Enhancing and Exploiting the Locality

Veljko Milutinovic, *Senior Member, IEEE*, and Mateo Valero, *Member, IEEE*

THE concept of cache memory has emerged as a solution for the ever increasing time domain gap between processor technology and memory technology. Since the very early works of Wilkes [13], the concept has evolved into a sophisticated system of hardware-implemented and software-implemented solutions. Actually, the best performance/complexity ratio is obtained through a synergistic interaction of hardware-based and software-based solutions.

The efficiency of the caching system is achieved through appropriate exploitation of the principles of temporal and spatial locality. Traditionally, temporal locality means that the probability is relatively high that a data or an instruction item will be reused in the near future. Spatial locality means that the probability is relatively high that the next data or instruction item to be used is in some way neighboring the previously used data or instruction item.

In traditional systems, temporal locality is exploited by keeping some of the most recently used data/instructions in the cache memory and by incorporating the cache hierarchy. Spatial locality is exploited by using larger cache blocks and by incorporating the prefetching mechanisms into the caching system. As technology gets more and more sophisticated, it has become obvious that a much better performance can be achieved through the incorporation of more sophisticated solutions for enhancing and exploiting of the locality present in the code or data.

As microprocessors get more and more complex, cache design and performance become more and more impacted by the solutions utilized in other domains, like superpipelining, superscaling, multithreading, prediction, parallelization, etc. Implementation issues in modern microprocessor systems are getting new dimensions. The issues of most interest for cache designers are treated in "Implementation Issues in Modern Cache Memories" by Jih-Kwon Peir, Windsor Hsu, and Alan J. Smith, while the impacts of multithreading on cache performance are treated in "Effects of Multithreading on Cache Performance" by Hantak Kwak, Ben Lee, Hurson Ali, Suk-Han Yoon, and Woo-Jong Han.

- V. Milutinovic is with the Department of Computer Engineering, School of Electrical Engineering, University of Belgrade, POB 35-54, 11120 Belgrade, Serbia, Yugoslavia. E-mail: vm@etf.bg.ac.yu.
- M. Valero is with the Department of Computer Architecture, Universidad Politecnica de Catalunya, c/ Jordi Girona 1-3, Modulo D6, 08034 Barcelona, Spain. E-mail: mateo@ac.upc.es.

For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 109034.

Optimal local memory performance is investigated in "Investigating Optimal Memory Performance" by Olivier Temam. It is important for the designers to know the theoretical limits before they can concentrate on their own ideas.

As indicated above, it has become obvious that more sophisticated approaches to locality exploitation are needed. Two early attempts imply the approaches by which the temporal and the spatial localities are handled by separate cache systems [2], [5]; this is in contrast to the traditional approaches by which the temporal and the spatial localities are treated using unified resources. The so-called split temporal/spatial cache approach can be implemented predominantly in hardware domain, predominantly in software domain, or in some combination of the two. Separate cache memories are maintained for data with a predominantly spatial locality and for data with a predominantly temporal locality. In its simplest form, hardware design parameters in two subsystems are tuned to the type of locality to be exploited and compiler helps with data classification. In its more sophisticated forms, only the temporal part includes the hierarchy and only the spatial part includes forms of prefetching, with data being able to migrate between the spatial and the temporal parts, with or without the assistance of the system software. More recent approaches explore an even wider plethora of possibilities [3], [8], [10], [12].

Systems with unified treatment of different locality types still prevail and can be classified into a number of correlated categories. Some of them focus on the final goal through appropriate cache architecture and design innovations, with no or no major compiler modifications (trace caching, victim caching, and randomized caching represent important new contributions). Examples include, but are not limited to, "Trace Cache: A Low Latency Approach to High Bandwidth Instruction Fetching" by Eric Rotenberg, Steve Bennett, and James E. Smith, "Evaluation of Design Options for the Trace Cache Fetch Mechanism" by Sanjay Jeram Patel, Daniel Holmes Friendly, and Yale N. Patt, and "Randomized Cache Placement for Eliminating Conflicts" by Nigel Topham and Antonio Gonzalez. Others imply more or less traditional cache architectures combined with relatively sophisticated compiler-based analysis (improving the cache locality by loop transformations, data transformations, or a combination of the two; improving cache performance by loop tiling, data alignment, or a combination of the two; and analysis/synthesis of temporal-based program behavior,

spatial-based program behavior, or a combination of the two). Examples include, but are not limited to, "Improving Cache Locality by a Combination of Loop and Data Transformations" by M. Kandemir, J. Ramanujam, and A. Choudhary, "Augmenting Loop Tiling with Data Alignment for Improved Cache Performance" by Preeti Ranjan Panda, Hiroshi Nakamura, Nikil D. Dutt, and Alexandru Nicolau, and "Analysis of Temporal-Based Program Behavior for Improved Cache Performance" by J. Kalamattianos, A. Khalafi, David Kaeli, and W. Meleis. Finally, the approaches which combine elements of both hardware and software support can be classified into hardware-mostly and software-mostly (oftentimes, the approaches fully oriented to compiler domain do include, sometimes hidden, hardware support and vice versa). Examples include, but are not limited to, "Prefetching with Markov Predictors" by Doug Joseph and Dirk Grunwald and "Compiler-Based Prefetching for Recursive Data Structures" by Chi-Keung Luk and Todd C. Mowry.

In multiprocessor systems of the SMP (shared-memory multiprocessors or symmetric multiprocessors) and the DSM (distributed shared memory) types, in addition to the traditionally defined locality types (temporal and spatial), a number of additional locality types are present and can/should be exploited (processor locality, cache consistency maintenance locality types, memory consistency modeling locality types, etc.). Appropriate mechanisms are incorporated in order to maintain the cache and memory consistency. More background information on issues of importance can be found in [11], [7]. All of these mechanisms represent a necessary system overhead, but also a source of system level information which can be utilized for potential performance improvements. In SMP and DSM systems, exploitation of sophisticated locality types is more implicit than explicit.

State-of-the-art research in scalable shared memory multiprocessor systems concentrates on two major research avenues:

- 1) performance evaluation and sophisticated verification aimed at better understanding of the potentials and ways in which different multiprocessor level localities can be exploited, and
- 2) architecture innovations aimed at better exploitation of different multiprocessor level localities.

Early multiprocessing-oriented research tried to exploit traditional locality types in the new multiprocessor context. For example, it has been found that entry memory consistency models may work better where the temporal locality prevails in the code, while the lazy release memory consistency models may work better where the spatial locality prevails [1]. Various locality types are also treated in [6].

More recent multiprocessing-oriented research tries to exploit the locality types inherent in the multiprocessing environments more directly (and nonexistent in the uniprocessor environments). For example, data-forwarding, remote-write, or cache-injection try to place data local to what is expected to be the next data processing site; for this purpose, appropriate hardware, software, or combined techniques can be used [4], [9].

As already indicated, evaluation of performance potentials is an important on-going research avenue. Popular protocols are analyzed in various environments and for various applications in "A Quantitative Analysis of the Performance and Scalability of Cache Coherence Protocols" by Mark Heinrich, Vijayaraghavan Soundararajan, Anoop Gupta, and John Hennessy. It has been found that the achieved performance and the optimal protocol change for different applications; protocol and architecture tuning to specific locality types, typical of the application, is absolutely necessary. Popular models are analyzed in the context which permits additional optimizations possible in the ILP (Instruction Level Parallelism) systems in "The Impact of Instruction-Level Parallelism on the Memory System Performance" by Vijay S. Pai, Parthasarathy Ranganathan, Hazim Abdel-Shafi, and Sarita V. Adve. It has been shown that additional optimizations lower the performance difference of various memory consistency models; this is because ILP optimizations equalize the locality patterns in typical code. An important prerequisite for further research is the existence of formal verification tools such as the one in "An Executable Specification and Verifier for Relaxed Memory Order" by Seungjoon Park and David L. Dill. Having such tools in hand, one can experiment with different models and how they behave in specific applications characterized by specific locality types.

Recent architectural research is characterized by numerous ideas; this fact indicates the prosperity of the field. The approaches which deserve special attention are given in "Excel-NUMA: Toward Programmability, Simplicity, and High Performance" by Zheng Zhang, Marcelo Cintra, and Josep Torrellas, "Coherence Controller Architectures for Scalable Shared Memory Multiprocessors" by Maged Michael, Ashwini K. Nanda, and Beng-Hong Lim, and "Exploiting the Benefits of Multiple-Path Network in DSM Systems: Architectural Alternatives and Performance Evaluation" by Donglai Dai and Dhableswar K. Panda. The paper by Zhang et al. introduces the Excel-NUMA approach, which enhances programmability by utilizing the fact that, after a local memory line is written by a processor, the memory location containing the line remains unused and can be used for temporary storage of remote data displaced from local caches. The paper by Michael et al. analyzes various coherence controller architectures and suggests solutions based on the proper characterization of communications patterns and statistics. The paper by Dai and Panda tries to exploit the benefits of the multiple-path networks for the best performance and performance/complexity in DSM systems, and proposes the novel block correlated FIFO channels approach to detect and prevent all potential coherence-sensitive race conditions.

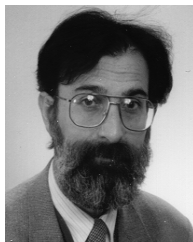
In conclusion, this overview effort tries to shed more light on the ongoing cache research in both the uniprocessing and multiprocessing arenas by pointing to a common new thread which is aimed at intensifying and exploiting different locality types present explicitly or implicitly in the application code or data [2], [5]. It is strongly believed that efficient treatment of locality issues can help achieve a significant improvement in performance and performance/complexity-ratio domains [5].

ACKNOWLEDGMENTS

The special issue editors are thankful to Professor Roger Espasa of the Universidad Politecnica de Catalunya for processing the volume of papers submitted to this special issue and for handling most e-mail correspondence with the special issue authors, and to Dr. Aleksandar Milenkovic of ETF for his helpful suggestions. The total number of papers received was 63 and the total number of evaluations generated by 160 reviewers was 233. In such conditions, due to the limited space, a number of excellent papers had to be rejected.

REFERENCES

- [1] S. Adve, A.L. Cox, S. Dwarkadas, R. Rajamony, and W. Zwaenopel, "A Comparison of Entry Consistency and Lazy Consistency Implementations," *Proc. Second HPCA Symp.*, pp. 26-37, Feb. 1996.
- [2] A. Gonzalez, C. Aliagas, and M. Valero, "A Data Cache with Multiple Strategies Tuned to Different Types of Locality," *Proc. ISC-95*, pp. 338-347, Barcelona, Spain, July 1995.
- [3] T.L. Johnson, M.C. Merten, and W.-M. Hwu, "Run-Time Spatial Locality Detection and Optimization," *Proc. Micro-30*, Dec. 1997.
- [4] D.A. Koufaty, X. Chen, D.K. Poulsen, and J. Torrellas, "Data Forwarding in Scalable Shared Memory Multiprocessors," *IEEE Trans. Parallel and Distributed Systems*, vol. 7, no. 12, pp. 1,250-1,264, Dec. 1996.
- [5] V. Milutinovic, B. Markovic, M. Tomasevic, and M. Tremblay, "The Split Temporal/Spatial Cache," *Proc. SCIZZL-5*, pp. 63-69, Santa Clara, Calif., Mar. 1996.
- [6] A. Prete, M. Graziano, and F. Lazzarini, "The ChARM Tool for Tuning Embedded Systems," *IEEE Micro*, vol. 17, no. 4, pp. 67-75, July/Aug. 1997.
- [7] J. Protic et al., *Distributed Shared Memory: Concepts and Systems*. Los Alamitos, Calif.: IEEE CS Press, 1998.
- [8] J.A. Rivers and E.S. Davidson, "Reducing Conflicts in Direct-Mapped Caches with a Temporality Based Design," *Proc. Int'l Conf. Parallel Processing*, 1996.
- [9] H.A. Shafi, J. Hall, S. Adve, and V. Adve, "An Evaluation of Fine-Grain Producer Initiated Communication in Cache-Coherent Multiprocessors," *Proc. Third HPCA Symp.*, pp. 204-215, Feb. 1997.
- [10] J. Sahuquillo and A. Pont, "The Split Data Cache in Multiprocessor Systems: An Initial Hit Ratio Analysis," *Proc. Seventh Euromicro Workshop Parallel and Distributed Processing*, Madeira, Portugal, Feb. 1999.
- [11] M. Tomasevic et al., *The Cache Coherence Problem in Shared Memory Multiprocessors*. Los Alamitos, Calif.: IEEE CS Press, 1993.
- [12] M. Tomasko, S. Hadjiyiannis, and W.A. Najjar, "Experimental Evaluation of Array Caches," *IEEE TCCA Newsletter*, pp. 11-16, Mar. 1997.
- [13] M. Wilkes, "The First Cache Memory Proposal," *Early Works*, 1951.



Veljko Milutinovic received his PhD from the University of Belgrade, Serbia, Yugoslavia, in 1982. He has been with the Department of Computer Engineering, School of Electrical Engineering, University of Belgrade since 1990. Prior to that, he was on the faculty of Purdue University, West Lafayette, Indiana. Dr. Milutinovic's research interests are in computer architecture/design, as well as in system support for electronic business on the Internet. He has contributed more than 50 papers to IEEE journals on

computer architecture/design and technology-aware system support for mission-critical applications. He has consulted for leading industry in the U.S. and Europe (IBM, RCA, NCR, AT&T, Virtual, eT, Zycad, Aerospace Corporation, ElectroSpace Corporation, Intel, Fairchild, Honeywell, Encore, Phillips, etc.). He was the principal designer or project leader on a number of market successful industrial efforts. He is the author of several books and was a co-editor for a number of IEEE tutorial books and conference proceedings. He has served as a guest editor for special issues of *Computer* and *Proceedings of the IEEE*. He has presented more than 200 invited talks around the world. Dr. Milutinovic is a senior member of the IEEE and a fellow of the Serbian Scientific Society of Engineers. For more information about Professor Milutinovic's current activities, please refer to <http://galeb.etf.bg.ac.yu/~vml/>.



Mateo Valero obtained his telecommunication engineering degree from the Polytechnic University of Madrid in 1974 and his PhD from the Polytechnic University of Catalonia (UPC) in 1980. He is a professor in the Computer Architecture Department at UPC. His current research interests are in the field of high performance architectures, with special interest in the following topics: processor organization, memory hierarchy, interconnection networks, compilation techniques, and computer benchmarking. He has

published approximately 200 papers on these topics. He served as the general chair for several conferences, including ISCA-98 and ICS-95, and has been an associate editor for *IEEE Transactions on Parallel and Distributed Systems* for three years. He is a member of the subcommittee for the Ecker-Mauchly Award. Dr. Valero has been honored with several awards, including the Narcis Monturiol presented by the Catalan government, the Salva i Campillo presented by the Telecommunications Engineer Association, and the King Jaime I by the Generalitat Valenciana. He is a member of the IEEE and director of the C4 (Catalan Center for Computation and Communications). Since 1994, he has been a member of the Spanish Engineering Academy. For more information about Professor Valero's current activities, please refer to <http://www.ac.upc.es/hpc>.