Usui, Takayuki; **Behrends, Reimer**; **Evans, Jacob**; **Smaragdakis, Yannis**
Adaptive locks: combining transactions and locks for efficient concurrency. (English)
Zbl 1233.68172
J. Parallel Distrib. Comput. 70, No. 10, 1009-1023 (2010).

Summary: Transactional memory is being advanced as an alternative to traditional lock-based synchronization for concurrent programming. Transactional memory simplifies the programming model and maximizes concurrency. At the same time, transactions can suffer from interference that causes them to often abort, from heavy overheads for memory accesses, and from expressiveness limitations (e.g., for I/O operations). In this paper we propose an adaptive locking technique that dynamically observes whether a critical section would be best executed transactionally or while holding a mutex lock. The critical new elements of our approach include the adaptivity logic and cost-benefit analysis, a low-overhead implementation of statistics collection and adaptive locking in a full C compiler, and an exposition of the effects on the programming model. In experiments with both micro and macrobenchmarks we found adaptive locks to consistently match or outperform the better of the two component mechanisms (mutexes or transactions). Compared to either mechanism alone, adaptive locks often provide 3-to-10x speedups. Additionally, adaptive locks simplify the programming model by reducing the need for fine-grained locking: with adaptive locks, the programmer can specify coarse-grained locking annotations and often achieve fine-grained locking performance due to the transactional memory mechanisms.

MSC:

68Q85 Models and methods for concurrent and distributed computing (process algebras, bisimulation, transition nets, etc.)
68M07 Mathematical problems of computer architecture

Keywords:

transactional memory; nested transactions; hybrid locks; adaptive locks

Software:

Atomizer; Autolocker; CIL; Eraser; Erlang; Feather-Trace; PhTM; SNZI; STAMP

Full Text: DOI

References:

[1] M. Abadi, A. Birrell, T. Harris, J. Hsieh, M. Isard, Implementation and use of transactional memory with dynamic separation, in: Compiler Construction, 2009.

[2] Abadi, M.; Birrell, A.; Harris, T.; Isard, M.: Semantics of transactional memory and automatic mutual exclusion, Popl, 63-74 (2008) · Zbl 1295.68149

[3] Armstrong, J.; Virding, R.; Wikström, C.; Williams, M.: Concurrent programming in Erlang, (1996)

[4] Bacon, D. F.; Konuru, R.; Murthy, C.; Serrano, M.: Thin locks: featherweight synchronization for Java, , 258-268 (1998)

[5] Blundell, C.; Devietti, J.; Lewis, E. C.; Martin, M. M. K.: Making the fast case common and the uncommon case simple in unbounded transactional memory, , 24-34 (2007)

[6] C. Blundell, E.C. Lewis, M.M.K. Martin, Unrestricted transactional memory: Supporting I/O and system calls within transactions, Technical Report CIS-06-09, Department of Computer and Information Science, University of Pennsylvania, April 2006.

[7] Blundell, C.; Lewis, E. C.; Martin, M. M.: Subtleties of transactional memory atomicity semantics, IEEE comput. Archit. lett. 5, No. 2, 17 (2006)

[8] Boyapati, C.; Rinard, M.: A parameterized type system for race-free Java programs, , 56-69 (2001)

[9] B.B. Brandenburg, J.H. Anderson, Feather-trace: A light-weight event tracing toolkit, in: Proceedings of the Third International Workshop on Operating Systems Platforms for Embedded Real-Time Applications, 2007, pp. 20–27.

[10] B.D. Carlstrom, J. Chung, H. Chafi, A. McDonald, C. Cao Minh, L. Hammond, C. Kozyrakis, K. Olukotun, Transactional execution of java programs, in: OOPSLA 2005 Workshop on Synchronization and Concurrency in Object-Oriented Languages,

SCOOL, 2005. · Zbl 1119.68041

[11] Carlstrom, B. D.; Chung, J.; Chafi, H.; Mcdonald, A.; Minh, C. C.; Hammond, L.; Kozyrakis, C.; Olukotun, K.: Executing Java programs with transactional memory, Sci. comput. Programming 63, 111-129 (2006) · Zbl 1119.68041 · doi:10.1016/j.scico.2006.05.006

[12] Cascaval, C.; Blundell, C.; Michael, M.; Cain, H. W.; Wu, P.; Chiras, S.; Chatterjee, S.: Software transactional memory: why is it only a research toy?, Commun. ACM 51, No. 11, 40-46 (2008)

[13] Cherem, S.; Chilimbi, T.; Gulwani, S.: Inferring locks for atomic sections, , 304-315 (2008)

[14] Chuang, W.; Narayanasamy, S.; Venkatesh, G.; Sampson, J.; Biesbrouck, M. V.; Pokam, G.; Calder, B.; Colavin, O.: Unbounded page-based transactional memory, , 347-358 (2006)

[15] D. Dice, Y. Lev, M. Moir, D. Nussbaum, Early experience with a commercial hardware transactional memory implementation, 2009, pp. 157–168.

[16] Dice, D.; Shalev, O.; Shavit, N.: Transactional locking II, Lecture notes in computer science 4167 (2006)

[17] P.C. Diniz, M.C. Rinard, Dynamic feedback: An effective technique for adaptive computing, in: SIGPLAN Conference on Programming Language Design and Implementation, PLDI, 1997, pp. 71–84,.

[18] Ellen, F.; Lev, Y.; Luchangco, V.; Moir, M.: Snzi: scalable nonzero indicators, , 13-22 (2007) · Zbl 1283.68119

[19] Flanagan, C.; Freund, S. N.: Atomizer: A dynamic atomicity checker for multithreaded programs, Sci. comput. Programming 71, No. 2, 89-109 (2008) · Zbl 1146.68350 · doi:10.1016/j.scico.2007.12.001

[20] B. Goetz, Optimistic thread concurrency, 2006. http://www.azulsystems.com/products/whitepaper/wp_otc.pdf.

[21] Gordon, M. I.; Thies, W.; Amarasinghe, S.: Exploiting coarse-grained task, data, and pipeline parallelism in stream programs, , 151-162 (2006)

[22] Gray, J.; Reuter, A.: Transaction processing: concepts and techniques, (1992) · Zbl 0781.68006

[23] Grossman, D.; Manson, J.; Pugh, W.: What do high-level memory models mean for transactions?, , 62-69 (2006)

[24] Guerraoui, R.; Herlihy, M.; Pochon, B.: Toward a theory of transactional contention managers, , 258-264 (2005) · Zbl 1314.68088

[25] Hammond, L.; Wong, V.; Chen, M.; Carlstrom, B. D.; Davis, J. D.; Hertzberg, B.; Prabhu, M. K.; Wijaya, H.; Kozyrakis, C.; Olukotun, K.: Transactional memory coherence and consistency, , 102 (2004)

[26] Harris, T.: Exceptions and side-effects in atomic blocks, Sci. comput. Programming 58, No. 3, 325-343 (2005) · Zbl 1080.68544 · doi:10.1016/j.scico.2005.03.005

[27] Harris, T.; Fraser, K.: Language support for lightweight transactions, , 388-402 (2003)

[28] T. Harris, M. Herlihy, S. Marlow, S. Peyton-Jones, Composable memory transactions, in: Proc. Symposium on Principles and Practice of Parallel Programming, 2005.

[29] Herlihy, M.: Apologizing versus asking permission: optimistic concurrency control for abstract data types, ACM trans. Database syst. 15, No. 1, 96-124 (1990)

[30] M. Herlihy, J.E.B. Moss, Transactional memory: Architectural support for lock-free data structures, in: Proceedings of the 20th Annual International Symposium on Computer Architecture, 1993, pp. 289–300.

[31] M. Hicks, J.S. Foster, P. Prattikakis, Lock inference for atomic sections, in: Proceedings of the First ACM SIGPLAN Workshop on Languages, Compilers, and Hardware Support for Transactional Computing, 2006.

[32] Krena, B.; Letko, Z.; Tzoref, R.; Ur, S.; Vojnar, T.: Healing data races on-the-fly, , 54-64 (2007)

[33] Kumar, S.; Chu, M.; Hughes, C. J.; Kundu, P.; Nguyen, A.: Hybrid transactional memory, , 209-220 (2006)

[34] Larus, J. R.; Rajwar, R.: Transactional memory, (2007)

[35] Y. Lev, M. Moir, D. Nussbaum, PhTM: Phased transactional memory, in: TRANSACT 07: Proceedings of the Second ACM SIGPLAN Workshop on Languages, Compilers, and Hardware Support for Transactional Computing, 2007.

[36] Mccloskey, B.; Zhou, F.; Gay, D.; Brewer, E.: Autolocker: synchronization inference for atomic sections, , 346-358 (2006)

[37] V. Menon, S. Balensiefer, T. Shpeisman, A.-R. Adl-Tabatabai, R.L. Hudson, B. Saha, A. Welc, Single global lock semantics in a weakly atomic STM, in: 3rd Workshop of Transactional Computing, TRANSACT, 2008.

[38] Menon, V.; Balensiefer, S.; Shpeisman, T.; Adl-Tabatabai, A. -R.; Hudson, R. L.; Saha, B.; Welc, A.: Practical weak-atomicity semantics for Java STM, , 314-325 (2008)

[39] C.C. Minh, J. Chung, C. Kozyrakis, K. Olukotun, STAMP: Stanford transactional applications for multi-processing, in: IISWC'08: Proc. IEEE International Symposium on Workload Characterization, 2008.

[40] Minh, C. C.; Trautmann, M.; Chung, J.; Mcdonald, A.; Bronson, N.; Casper, J.; Kozyrakis, C.; Olukotun, K.: An effective hybrid transactional memory system with strong isolation guarantees, , 69-80 (2007)

[41] K.E. Moore, J. Bobba, M.J. Moravan, M.D. Hill, D.A. Wood, LogTM: Log-based transactional memory, in: Proceedings of the 12th International Symposium on High-Performance Computer Architecture, 2006, 254–265.

[42] Moravan, M. J.; Bobba, J.; Moore, K. E.; Yen, L.; Hill, M. D.; Liblit, B.; Swift, M. M.; Wood, D. A.: Supporting nested transactional memory in logtm, , 359-370 (2006)

[43] Necula, G. C.; Mcpeak, S.; Rahul, S. P.; Weimer, W.: CIL: intermediate language and tools for analysis and transformation of C programs, , 213-228 (2002) · Zbl 1051.68756 · link.springer.de

[44] Ni, Y.; Welc, A.; Adl-Tabatabai, A. -R.; Bach, M.; Berkowits, S.; Cownie, J.; Geva, R.; Kozhukow, S.; Narayanaswamy, R.; Olivier, J.; Preis, S.; Saha, B.; Tal, A.; Tian, X.: Design and implementation of transactional constructs for c/c++, , 195-212

(2008)

[45] R. Rajwar, J. Goodman, Speculative lock elision: Enabling highly concurrent multithreaded execution, in: 34th International Symposium on Microarchitecture, December 2001, p. 294. http://doi.ieeecomputersociety.org/10.1109/MICRO.2001.991127.

[46] Rajwar, R.; Herlihy, M.; Lai, K.: Virtualizing transactional memory, , 494-505 (2005)

[47] Rinard, M. C.: Effective fine-grain synchronization for automatically parallelized programs using optimistic synchronization primitives, ACM trans. Comput. syst. 17, No. 4, 337-371 (1999)

[48] Rossbach, C. J.; Hofmann, O. S.; Porter, D. E.; Ramadan, H. E.; Bhandari, A.; Witchel, E.: Txlinux: using and managing hardware transactional memory in an operating system, , 87-102 (2007)

[49] A. Roy, S. Hand, T. Harris, A runtime system for software lock elision, in: Eurosys, 2009.

[50] Saha, B.; Adl-Tabatabai, A. -R.; Jacobson, Q.: Architectural support for software transactional memory, , 185-196 (2006)

[51] Savage, S.; Burrows, M.; Nelson, G.; Sobalvarro, P.; Anderson, T.: Eraser: A dynamic data race detector for multi-threaded programs, , 27-37 (1997)

[52] Iii, W. N. Scherer; Scott, M. L.: Advanced contention management for dynamic software transactional memory, , 240-248 (2005)

[53] Schneider, F. T.; Menon, V.; Shpeisman, T.; Adl-Tabatabai, A. -R.: Dynamic optimization for efficient strong atomicity, , 181-194 (2008)

[54] N. Shavit, D. Touitou, Software transactional memory, in: PODC, 1995, pp. 204–213. · Zbl 1373.68178

[55] Shpeisman, T.; Menon, V.; Adl-Tabatabai, A. -R.; Balensiefer, S.; Grossman, D.; Hudson, R. L.; Moore, K. F.; Saha, B.: Enforcing isolation and ordering in STM, , 78-88 (2007)

[56] Shriraman, A.; Spear, M. F.; Hossain, H.; Marathe, V. J.; Dwarkadas, S.; Scott, M. L.: An integrated hardware-software approach to flexible transactional memory, , 104-115 (2007)

[57] Smaragdakis, Y.; Kay, A.; Behrends, R.; Young, M.: Transactions with isolation and cooperation, (2007)

[58] Y. Smaragdakis, A. Kay, R. Behrends, M. Young, General and efficient locking without blocking, in: ACM SIGPLAN Workshop on Memory Systems Performance and Correctness, MSPC, 2008.

[59] M. Spear, M. Michael, M. Scott, Inevitability mechanisms for software transactional memory, in: 3rd Workshop of Transactional Computing, TRANSACT, 2008.

[60] H. Volos, A.J. Tack, N. Goyal, M.M. Swift, A. Welc, xCalls: Safe I/O in memory transactions, in: EuroSys'09: Proc. 4th ACM European Conference on Computer Systems, 2009, pp. 247–260.

[61] Von Praun, C.; Gross, T. R.: Object race detection, , 70-82 (2001)

[62] A. Welc, A.L. Hosking, S. Jagannathan, Transparently reconciling transactions with locking for java synchronization, in: European Conference on Object-Oriented Programming, 2006, pp. 148–173.

[63] Welc, A.; Saha, B.; Adl-Tabatabai, A. -R.: Irrevocable transactions and their applications, , 285-296 (2008)