Paper Review Form (1000 words maximum)
cs940: The Multikernel: A New OS Architecture for Scalable Multicore Systems [1]

January 22, 2018

Paper Summary

3–5 sentences. Briefly summarise the **contributions** *of the paper, i.e., what it adds over the state of the art. Paraphrase and extract the essentials rather than simply copying chunks of text. Be objective; later sections allow for your own opinion.*

A few years into the wider availability of multicore processors in the consumer market, constraints imposed by conventional shared-memory kernel architectures became apparent in high-performance applications. After demonstrating the negative performance impacts of these constraints on shared state multiprocessing, the authors proposed an alternative kernel model based on inter-core message passing. The implemented *Barrelfish* multikernel achieved better scaling performance than conventional kernels in selected test cases. The authors also suggested directions of future work to reduce the complexity in porting conventional kernel applications, and to improve the multikernel's networking and file system capabilities.

Pros and Cons

6 bullets. Succinctly state three positives and three negatives of the paper.

I believe that the paper has the following positive features:

- The authors made very accurate observations on the state of multicore support in operating systems of the time, and clearly identified the disadvantages of shared memory kernels running on increasing number of processor cores.
- The paper presented benchmarking results on data updates under both shared-memory and message passing concurrency, as a clear motivation to develop a multikernel based on inter-core message passing.
- The authors identified negative effects of some of their design decisions; and where possible, provided justifications on why these decisions can benefit the multikernel overall, or what they would have done in hindsight.

I believe that the paper has the following negative features:

- Some design decisions were made based on optimistic predictions of future hardware development, resulting in limited practical performance boost when consumer-grade processors only reached six physical cores at the 8 year mark from publication [2].
- The negative effects of long message queues were mentioned, but not sufficiently explored (e.g. through a case study).
- As the multikernel still requires significant manual optimisation for each hardware architecture (e.g. x86) to gain full performance benefits from message passing, it is unclear whether the goal of hardware-neutral OS structure can be achieved.

The Problem/Motivation

1–2 sentences per question. What is the motivation for the work, or the problem being solved? Why is it important? If there is prior art, how was it insufficient? If the problem had not previously been solved, why not?

Modern general-purpose operating systems are expected to work on a wide range of hardware systems, each with opportunities for optimisation that require the OS to specifically adapt for, resulting in increasingly complex operating system designs. The authors believe that a fundamental redesign of shared-memory kernels in these operating systems is required.

Hardware development also means that it is now possible to optimise for heterogeneous cores and take advantage of reduced cost in message passing between cores. The authors believe that these features can be better exploited with a message passing kernel model to create more self-containing cores.

Earlier researches [3, 4] had already identified the potential of (but not yet implemented) message passing kernels. In implementing an operational multikernel, authors took inspirations from prior OS research [5, 6, 7]. They also identified some complementary work tackling similar problems with different methods [8, 9].

The Solution/Approach

5–10 sentences. What have they done? How does it address the issues set out above? How is it unique and/or innovative (if, indeed, it is)? Give details, again using the paper as the source but again, not just copying text. Instead, focus on paraphrasing/synopsising, and extracting the essential details.

The authors seek to construct a message passing multikernel which scales better than shared-memory kernels in multicore performance, although not necessarily faster at running low-process-count applications, as later shown.

The use of explicit inter-core communication allows the OS to provide isolation and resource management on heterogeneous cores, and enable more efficient use of waiting time. A hardware-neutral OS structure reduces the cost of constantly optimising the OS for new

hardware-specific features. The move from shared to replicated states allows maintaining consistency through more efficient and lower latency message passing, and enables limited shared states to be implemented in the future.

On the implementation aspect, the OS is split into a privileged-mode CPU driver and a user-mode monitor on each core. The former provides traditional critical kernel features such as protection, scheduling and access control; while the latter is responsible for inter-core communication and coordinating global consistency. Dispatcher objects are used to represent processes on each core, each responsible for scheduling process threads on the core. Global consistency of memory and shared address space still have to be maintained between cores, resulting in complexities in implementation and performance. Knowledge of underlying hardware is still needed to provide topology-aware message passing and memory allocation.

Evaluation

3–4 sentences. How do they evaluate their work? What questions does their evaluation set out to answer? What does their evaluation say about the strengths and weaknesses of their system? What is your opinion of the strengths and weaknesses of the evaluation itself? Give highlights, not a point by point reproduction of the evaluation section(s). In rare cases, systems papers may not have any evaluation, in which case write 'N/A' below.

The evaluations performed were to address two primary areas of concern: whether the message passing multikernel can achieve comparable or better performance than shared-memory kernels at the time; and whether the new multikernel enabled greater scaling potential in view of future hardware development.

The performance of typical coordinated memory operations, compute-bound operations and heavy IO workloads were evaluated in comparison with shared-memory kernels'. The multikernel outperformed shared-memory kernels at greater number of cores, as well as in IO testing, owing to its scaling advantages. Other tests revealed comparable or worse performance from the multikernel due to its somewhat increased single-core overhead and limited optimisation based on hardware knowledge.

I believe that the evaluations were fairly comprehensive and confirmed both the advantages and the disadvantages to be expected with the design.

Your Opinion

At least 3 sentences. This is the fun part where you get to judge both the paper and the work it reports! Is the motivation convincing? The problem important? The approach a good or bad idea? Why? Which specific things annoyed you, or you thought were cool, or cool-but-flawed? Justify your opinions! Make an argument which will convince others of your opinion.

David Chisnall has argued the fallacy of investing extensive effort in C compilers to pretend that the language still represents a close abstraction of the underlying hardware, which is no

longer the case with hardware optimisations such as out-of-order execution and speculative execution, occasionally leading to serious vulnerabilities [10]. This is a point that can and has been well made for shared-memory kernels as well, providing a good motivation to explore the possibility of a fundamentally-redesigned kernel model.

Barrelfish became one of the first operational multikernel implementations, with very promising results shown in evaluations. There are nevertheless difficulties preventing its wider adaptation, but it is conceivable that a functional prototype can encourage further commercial and academic efforts being made to pursue improved solutions in this area, notably in extending Barrelfish to achieve practical performance benefits [11], and in adapting the idea of networked single-core environments to produce minimised VM images [12].

Questions for the Authors

Finally, imagine you're attending a talk about this paper given by one of the authors. Give at least 2 questions that you would like to ask, specific to the paper and the research it reports.

- As the user-mode monitor is responsible for coordinating global memory consistency and inter-core communication, could malicious monitors cause disruptions to the system through, for example, denial of service by spamming page remapping requests? Are there any other security implications associated with moving these traditional kernel operations into user-mode?
- What do you perceive to be the business cost of porting a typical Linux device driver to Barrelfish? To what degree would a high cost hinder its adoption?

References

[1] A. Baumann, P. Barham, P.-E. Dagand, T. Harris, R. Isaacs, S. Peter, T. Roscoe, A. Schüpbach, and A. Singhania, "The multikernel: a new os architecture for scalable multicore systems," in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*. ACM, 2009, pp. 29–44.

[2] I. Corporation, "Intel® Core™ i7-8700K Processor," December 2017. [Online]. Available: https://ark.intel.com/products/126684/Intel-Core-i7-8700K-Processor-12M-Cache-up-to-4_70-GHz

[3] E. M. Chaves, P. C. Das, T. J. LeBlanc, B. D. Marsh, and M. L. Scott, "Kernel-kernel communication in a shared-memory multiprocessor," *Concurrency and Computation: Practice and Experience*, vol. 5, no. 3, pp. 171–191, 1993.

[4] H. C. Lauer and R. M. Needham, "On the duality of operating system structures," *ACM SIGOPS Operating Systems Review*, vol. 13, no. 2, pp. 3–19, 1979.

[5] J. Liedtke, *On micro-kernel construction*. ACM, 1995, vol. 29, no. 5.

[6] B. Gamsa, O. Krieger, J. Appavoo, and M. Stumm, "Tornado: Maximizing locality and concurrency in a shared memory multiprocessor operating system," in *OSDI*, vol. 99, 1999, pp. 87–100.

[7] A. S. Tanenbaum and R. Van Renesse, "Distributed operating systems," *ACM Computing Surveys (CSUR)*, vol. 17, no. 4, pp. 419–470, 1985.

[8] D. Shelepov and A. Fedorova, "Scheduling on heterogeneous multicore processors using architectural signatures," 2008.

[9] D. Wentzlaff and A. Agarwal, "Factored operating systems (fos): the case for a scalable operating system for multicores," *ACM SIGOPS Operating Systems Review*, vol. 43, no. 2, pp. 76–85, 2009.

[10] P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution," *arXiv preprint arXiv:1801.01203*, 2018.

[11] S. Peter, J. Li, I. Zhang, D. R. Ports, D. Woos, A. Krishnamurthy, T. Anderson, and T. Roscoe, "Arrakis: The operating system is the control plane," *ACM Transactions on Computer Systems (TOCS)*, vol. 33, no. 4, p. 11, 2016.

[12] A. Madhavapeddy, R. Mortier, C. Rotsos, D. Scott, B. Singh, T. Gazagnaire, S. Smith, S. Hand, and J. Crowcroft, "Unikernels: Library operating systems for the cloud," in *ACM SIGPLAN Notices*, vol. 48, no. 4. ACM, 2013, pp. 461–472.