Shoal: smart allocation and replication of memory for parallel programs

Stefan Kaestle (student, presenting), Reto Achermann (student), Timothy Roscoe, Tim Harris[†]

Systems Group, Dept. of Computer Science, ETH Zurich [†]Oracle Labs, Cambridge, UK

Modern NUMA multicore machines exhibit complex latency and throughput characteristics, making it hard to allocate memory optimally for a given program's access patterns. However, good placement of and access to program data is crucial for application performance, and, if not carefully done, can significantly impact scalability [1, 3]. Although there is research (e.g. [1, 2]) for how to adapt software for concrete characteristics of such machines, many programmers struggle to apply these to their applications.

It is unclear which NUMA optimization to apply in which setting. For example, some programs have good access locality and it is worth co-locating code with the data it is accessing. Other programs exhibit random accesses and it is best to spread code and data across the entire machine to maximize total bandwidth to memory. The situation becomes even more complex as we consider large page sizes (where the data on a large page must be co-located physically), non-cache-coherent memory with different characteristics, or heterogeneous and specialized cores. With rapidly evolving and diversifying hardware, programmers must repeatedly make manual changes to their software to keep up with new hardware performance properties.

One solution to achieve better data placement and faster data access is to rely on automatic online monitoring of program performance to decide how to migrate data [3]. However, program's semantics then have to be guessed in retrospect from a narrow set of available information (i.e. from data based on sampling using performance counters). Such approaches are also limited to a relatively small number of optimizations. For example, it is hard to incrementally activate large pages or enable the use of DMA hardware for data copies dynamically.

We present Shoal, a system that abstracts memory access and provides a rich programming interface that accepts hints on memory access patterns at runtime. These hints can either be manually written or automatically derived from high-level descriptions of parallel programs. Shoal includes a machine-aware runtime that selects optimal implementations for this memory abstraction dynamically based on the hints and a concrete combination of machine and workload. If available, Shoal is able to exploit not only NUMA properties but also hardware features such as large and huge pages as well as DMA copy engines.

Our contributions are as follows: Firstly, we provide higher-level memory abstractions to allow Shoal to change memory allocation and access at runtime. In our prototype, we provide an array-like abstraction, designed for use in C/C++ code. Secondly, we introduce techniques for automatically selecting among several highly tuned array implementations using access patterns and machine characteristics. Finally, we modified Green-Marl [4], a graph analytics language, to show how Shoal can extract access patterns automatically from high-level descriptions. We demonstrate significant performance benefits when applying these techniques to Green-Marl when using Shoal for unmodified Green-Marl programs.

[1] APPAVOO, J., SILVA, D. D., KRIEGER, O., AUSLANDER, M., OSTROWSKI, M., ROSENBURG, B., WATERLAND, A., WISNIEWSKI, R. W., XENIDIS, J., STUMM, M., AND SOARES, L. Experience Distributing Objects in an SMMP OS. *ACM Transactions on Computer Systems 25*, 3 (Aug. 2007).

[2] BAUMANN, A., BARHAM, P., DAGAND, P.-E., HARRIS, T., ISAACS, R., PETER, S., ROSCOE, T., SCHÜPBACH, A., AND SINGHANIA, A. The Multikernel: A new OS architecture for scalable multicore systems. In *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles* (New York, NY, USA, 2009), SOSP '09, ACM, pp. 29–44.

[3] DASHTI, M., FEDOROVA, A., FUNSTON, J., GAUD, F., LACHAIZE, R., LEPERS, B., QUEMA, V., AND ROTH, M. Traffic Management: A Holistic Approach to Memory Placement on NUMA Systems. In *Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems* (New York, NY, USA, 2013), ASPLOS '13, ACM, pp. 381–394.

[4] HONG, S., CHAFI, H., SEDLAR, E., AND OLUKOTUN, K. Green-Marl: A DSL for Easy and Efficient Graph Analysis. In *Proceedings of the 17th International Conference on Architectural Support for Programming Languages and Operating Systems* (New York, NY, USA, 2012), ASPLOS XVII, ACM, pp. 349–362.