Ad-hoc composable cache coherent systems - a fairy tale?

Jasmin Schult
ETH Zurich
Switzerland
jasmin.schult@inf.ethz.ch

Timothy Roscoe
ETH Zurich
Switzerland
troscoe@inf.ethz.ch

Emerging open cache coherent interconnect standards like CXL [2] allow us to build dynamically composable, heterogeneous NUMA systems that transparently maintain system-wide cache coherence in hardware.

Our experience in building the Enzian platform [1] has convinced us that these composable, transparently coherent NUMA systems will never be realised. Interoperability and implementation challenges aside, we expect that the performance of transparently maintained coherence will suffer greatly from the lack of explicit co-design and integration of dynamically composed components. With this work, we aim to demonstrate that the traditional, transparent management of private caches is inherently unsuited to ad-hoc composed systems, because it relies on careful integration and co-design of all involved components to achieve acceptable performance.

The commodity systems built over the last two decades transparently manage their cache hierarchies in hardware and only expose a simple load and store interface to software. As a result, the cache allocation policy of these systems is simple and rigid: The data is pulled into the local private cache on access, and coherence with shared caches is resolved as indicated by the sharing metadata. For NUMA systems, this metadata is maintained by so-called *sparse directories*, which track the presence of a NUMA node's local memory in remote caches. These sparse directories are themselves *caches*, requiring eviction of sharing information and the invalidation of the corresponding remotely cached data when their capacity or set-associativity is exceeded.

The *thrashing* of NUMA coherence directories fundamentally undermines the utility of remote data caching, because a directory entry must be *maintained* for as long as possibe to allow remotely cached data to be reused. This is in contrast to other globally shared resources, such as link and shared memory bandwidth, which are only *transiently* used to process events. With directory thrashing, the usefulness of remote caching decreases up to the point where no reuse of data is possible, leading to a 2x bandwidth and latency overhead compared to not caching remote memory at all and performing in-place accesses instead. It is therefore vital to avoid such thrashing.

Existing coherent NUMA systems avoid directory thrashing by carefully provisioning their directories with respect to the size and associativity of the last-level node caches and the ratio of remote data accesses.

Future, ad-hoc composed coherent systems are likely to have mismatched directory and cache provisioning, severely degrading performance due to the resulting thrashing. This is due to the ad-hoc composition of nodes of heterogeneous types, different vendors and generations. In the current paradigm, such a mismatch cannot be mitigated in software, because the management of caches performed transparently and rigidly in hardware: The directory resources are *managed and used by separately designed hardware mechanisms*, so software has no influence over the global directory contention problem arising between ad-hoc assumbled pieces of hardware. If composable, transparently coherent systems were built, we would be resigned to watch from the sidelines while the effort of implementing global, full-fleged coherence are wasted.

To validate our claim, we plan to perform a thorough analysis of the behavior of ad-hoc composed cache coherent systems in simulation.

With our poster, we want to raise the issue of mismatched directory with the community, using an abstract example where a directory starts thrashing because of a streaming-like memory access by a remote node with a comparatively oversized cache. We want to spark a discussion about whether it is worth solving this issue to enable composable coherent systems and if so, how we should introduce more software control over hardware-maintained cache coherence to mitigate resource mismatches.

References

[1] Cock, D., Ramdas, A., Schwyn, D., Giardino, M., Turowski, A., He, Z., Hossle, N., Korolija, D., Licciardello, M., Martsenko, K., Achermann, R., Alonso, G., and Roscoe, T. Enzian: An open, general, CPU/FPGA platform for systems software research. Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, ACM, pp. 434–451.

[2] Sharma, D. D. Compute Express Link (CXL): Enabling Heterogeneous Data-Centric Computing With Heterogeneous Memory Hierarchy. *IEEE Micro 43*, 2 (Mar. 2023), 99–109.