CS 5341

Oscar Galindo

**HMG: Extending Cache Coherence Protocols Across Modern Hierarchical Multi-GPU Systems**

This paper begins by stating that “as the demand for GPU compute continues to grow beyond what a single die can deliver … vendors are turning to new packaging technologies such as multichip modules … and new networking solutions”. “However due to physical limitations, the large bandwidth discrepancy between existing inter-GPU links and on-package integration technologies can contribute to Non-Uniform Memory Access behavior that often bottlenecks performance”.

To address the previously mentioned performance bottleneck the authors propose the adoption of a cache coherence protocol called “HMG” (i.e., Hierarchical Multi-GPU). Which is a “hardware-managed cache coherence protocol for distributed L2 caches in hierarchical multi-GPU platforms”. The idea is to implement NHCC, which is a current cache coherence protocol for intra-GPU, multichip modules, at the scale inter-GPU communication scale. The current implementation of NHCC allows the hardware to determine through hashing the “home” of a cached item. At the home module there is a list of the modules that contain a copy of the locally cached item. This allows to quickly compute the “home” of the needed item and request to that module a copy of the cached item is sent back to the requester. In addition, this permits to more efficiently determine the invalidation messages that must be sent, rather than sending an invalidation message to all modules. But above all, an important enhancement is that any item in L2 cache has only two states, invalid and valid. An important performance improvement comes from the fact that under NHCC Non-synchronizing stores do not require acknowledgements. The mechanisms of how the stores and loads cause invalidations are outlined in the paper.

To implement their proposed “HMG” cache coherence protocol, the authors take the NHCC’s implementation and extend it to have a multi-GPU scope. Basically, this implementation permits to reduce the necessity to load from remote locations unless needed. The scheme allows to define a global home for every cached item, this means it is known at every point on what GPU the line is found at a global scale which is called the “Sys home” of the item. In addition, for every cached line, the protocol defines/assigns one of the modules in every GPU as the “local home” of the cached item if the item is used by some module within a GPU. This permits to maximize the number of requests that are kept local to the machine once a line is locally cached within a GPU, therefore lowering the number of requests that travel from GPU to GPU in the interconnection network. It is easy to see that with such mechanism the requests for non-local-data-to-the-GPU cause the interconnection network to request from GPU to GPU the line of data, but once the item is cached in the local/“GPU home” of the item, the rest of the modules can address the home module of the item to access the data. Invalidation works similarly but in reverse. As in NHCC, HMG requires only two states for every data item, and adds only one extra transition than those found in NHCC. The fact that this coherence protocol offers less communication overhead over the states of data and faster responses of propagation through the hierarchy even when non-local data is modified aids in reducing the effects of NUMA, which was the goal of the authors. Further explanation of the stores and loads are found in the paper, as well as release and acquire mechanisms.

During the evaluation part of the work, the authors present impressive results with the application of this coherence protocol. To begin, the authors mention that “… HMG is able to deliver 97% of the ideal speedup that inter-GPU caching can possibly enable.” The authors do offer graphs that explain that as L2 cache size is increased, the inter-GPU bandwidth is increased, and even when the size of the directories needed to keep references of the cached lines increase performance still increases and approaches speedups similar to those of having no cache-coherence as a policy, which is impressive. In addition, the authors offer the speedup results when the HMG coherence protocol is applied to a system running common tasks like training an LSTM neural network or training GoogLeNet. In all cases the performance speedup of HMG matches or even outperforms a system where the same tasks are run without cache coherence, given equal HW and SW conditions. Not being enough of a contribution already, the authors mention the relative size of the overhead incurred by following HMG as a coherence protocol is minimal (i.e., around 2.7% of the cache space).

With all the previous data in mind I presume the contributions of this paper will be utilized in forthcoming systems, in fact I do not think there exists a single thing that could be negative about the work. Great work overall.