

GPU-based Networking

15-740 Computer Networks Course - Project Proposal

Matthew Mukerjee, David Naylor, Bruno Vavala
Carnegie Mellon University
Email: {mukerjee, dnaylor, bvavala}@cs.cmu.edu

URL: <http://www.cs.cmu.edu/~bvavala/misc/project740/>

I. PROJECT DESCRIPTION

Today's router based algorithms are typically run on CPUs, or on network processing units, in order to provide the necessary flexibility (with respect to ASICs) to adapt them to several applications/scenarios. Just to cite a few examples, longest-prefix matching, firewall rule matching and flow rule matching all require to be adapted over time according to the application and, due to the high speed of today's networks, the processing must be performed at a very high rate.

Despite more and more parallelism is available on the CPUs to run several threads concurrently, these chips are not targeted at manipulating a huge (and constantly increasing) amount of data. Different architectures are more appealing. GPUs for instance have a highly parallel structure that makes them more suitable for such operations. They have lots of smaller cores that run more lightweight threads, typically operating in parallel on a dedicated portion of data.

The objective of this project is thus to analyze the convenience of exploiting GPUs for packet processing. By considering that today's networks transfer packets at a very high rate and also that the processing is mostly stateless, then the GPU architecture looks like an interesting candidate to achieve efficiency. In fact, each thread could be assigned a different packet to process (perhaps a different field of the packet in the case of a rule matching algorithm). The simple operations on the packets and the availability of cores would allow to achieve the high throughput required by the application in the case of longest prefix matching for packet routing, or necessary for an accurate and up-to-date analysis of the traffic in the case of intrusion detection systems.

Our 75% goal is to implement some router based algorithms (like the ones listed above) both CPU-only based and GPU-accelerated, compare their performance, and analyze strengths, limits and bottlenecks of the architectures in order to justify the results. Our 100% goal is to run a fair comparison between our solution and the one in [1] (and possibly improve on it). It will be interesting to analyze the impact of different design choices. Our 125% goal is to integrate our work into a network (or firewall, or SDN) prototype. The objective would be to investigate the convenience of using GPU based solutions by comparing the gain in performance with the cost of the additional hardware and implementation. The 100% and 125% goals are listed unordered.

II. LOGISTICS

Plan of Attack and Schedule. Week 1. Familiarize with the CUDA framework, discuss and retrieve the software we need to use/modify for our work. Week 2. Develop an initial version of the GPU-accelerated longest-prefix match algorithm (Bruno, Matt) and firewall rule matching (Matt, David); set up a data/packet generator (Bruno, David). Week 3. Profile code (Matt, David), optimize it (Matt, Bruno), discuss tests to perform (Matt, David, Bruno). Implement result data processing (i.e., data management and plots) (Bruno, David). Week 4. Try to have Packetshader working (possibly its packet generator, or another available tool) (Matt, David). Run some preliminary tests (David, Bruno). Review the schedule based on progress (more optimizations needed? benchmarks/benchmarking tools available and ready? better to switch to the integration with a prototype?) (Bruno, David, Matt). Report progress and preliminary results in the milestone (Bruno, Matt). Week 5. Continue with implementation, code profiling and optimization. Run

tests (Matt, David). Find and discuss/justify bottlenecks (Bruno, David). Consider the complexity and feasibility of further improvements (Bruno, Matt). Week 6. Based on progress, if low then continue developing the algorithms. Otherwise, set up the prototype (XIA of other software) for the integration (Matt, David). Port the solution in the new environment (Bruno, David). Run some tests, with and without our configuration (Bruno, Matt). Final report (Bruno, Matt, David).

Milestone. We plan to have some working router based algorithms and report about their performance. We plan to give some initial insights on the GPU architecture and the design choices they induce. We should have an environment ready to run some tests and process the results. We should have at least a set of benchmarks

Literature Search. Despite the intuitive usefulness of GPUs in successfully dealing with high-rate packet processing, we found few works published in the literature. Packetshader [1] is a software router that addresses the problem of CPU bottleneck by moving the IP table lookup and IPSec encryption operations to the GPU. Though a substantial speed-up is reached, it is highlighted in [2] that it would be possible to exploit further the GPU's memory architecture to improve the performance. The crucial point is to increase the utilization of the small constant memory of the GPU (opposed to the slower global memory) by maintaining indices of the forwarding table.

In [3] it is proposed a GPU accelerated IP lookup algorithm. The main idea is to keep in the global memory of the GPU all of the possible IP prefix entries. This is (almost) feasible due to the current limited IP address space, but the work has not been extended to cover the larger prefixes of the IPv6 protocol.

The work in [4] investigates the usefulness of GPU-accelerated network algorithms. The authors devise a new string matching algorithm using Bloom filters and a regular expression matching algorithm using finite automata, as these are basic building blocks for intrusion detection systems. Then they analyze the difference in performance of some data structures in the long prefix match problem. A performance improvement of network intrusion detection algorithms through a graphics card processor was originally described in [5].

A common feature of the works is that as they improve the performance of some particular algorithm, the speed up is then limited by the bandwidth between the CPU and the GPU, which thereby becomes a bottleneck.

Finally, the Hermes [6] micro architecture exploits the design of closely-coupled CPU and GPU to share memory between them. This allows to reduce the latency and increase the throughput.

Resources Needed. The main software needed for the project is the CUDA framework for GPU programming, which is freely available online. Also our notebooks are all equipped with a GPU. Some software related to [1] is available online to run an experimental evaluation. A network prototype (XIA) is also available to integrate our solution.

Getting Started. The CUDA framework has been installed and minimally tested. The only constraint is the CUDA syntax and how threads and memory are managed by it (but it is part of the familiarization step).

REFERENCES

- [1] S. Han, K. Jang, K. Park, and S. Moon, "Packetshader: a gpu-accelerated software router," in *Proceedings of the ACM SIGCOMM 2010 conference*, ser. SIGCOMM '10. New York, NY, USA: ACM, 2010, pp. 195–206. [Online]. Available: <http://doi.acm.org/10.1145/1851182.1851207>
- [2] Y. Lee, M. Jeong, S. Lee, and E.-J. Im, "Fast forwarding table lookup exploiting gpu memory architecture," in *Information and Communication Technology Convergence (ICTC), 2010 International Conference on*, nov. 2010, pp. 341–345.
- [3] J. Zhao, X. Zhang, X. Wang, and X. Xue, "Achieving o(1) ip lookup on gpu-based software routers," in *Proceedings of the ACM SIGCOMM 2010 conference*, ser. SIGCOMM '10. New York, NY, USA: ACM, 2010, pp. 429–430. [Online]. Available: <http://doi.acm.org/10.1145/1851182.1851246>
- [4] S. Mu, X. Zhang, N. Zhang, J. Lu, Y. S. Deng, and S. Zhang, "Ip routing processing with graphic processors," in *Proceedings of the Conference on Design, Automation and Test in Europe*, ser. DATE '10. 3001 Leuven, Belgium, Belgium: European Design and Automation Association, 2010, pp. 93–98. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1870926.1870950>
- [5] G. Vasiliadis, S. Antonatos, M. Polychronakis, E. P. Markatos, and S. Ioannidis, "Gnort: High performance network intrusion detection using graphics processors," in *Proceedings of the 11th international symposium on Recent Advances in Intrusion Detection*, ser. RAID '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 116–134. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-87403-4_7
- [6] Y. Zhu, Y. Deng, and Y. Chen, "Hermes: an integrated cpu/gpu microarchitecture for ip routing," in *Proceedings of the 48th Design Automation Conference*, ser. DAC '11. New York, NY, USA: ACM, 2011, pp. 1044–1049. [Online]. Available: <http://doi.acm.org/10.1145/2024724.2024953>