Consolidated Review of
Indexing Millions of Packets per Second Using GPUs

1. Strengths:

Description of algorithms to allow commodity GPUs to support indexing of (185) millions of records per second, showing the feasibility of indexing traffic at multi-10-Gbps rates. The GPU-based algorithm provides a 20-fold improvement over a CPU and does not suffer from performance losses during high load. Hence it is more stable during abnormal/malicious network activity

2. Weaknesses

No evidence they are sharing tools/software.

The paper is too short to offer a proper analysis of the proposed system. E.g., it is very hard for a non-expert to pinpoint where the impressive performance exactly comes from, or what would be reasonable alternative designs.

3. Comments

Overall, this is a nicely written paper, with an interesting use for GPUs (there are missing citations for GPUs in networking, e.g., PacketShader from Sigcomm'10, intrusion detection via signatures http://pages.cs.wisc.edu/~estan/publications/gpusigmatching.pdf, and others). The authors tee up an important problem. And, they then methodically show us how to make progress on the problem in theory. And, finally they show us how a working prototype of the system performs. Nice and methodical.

This is not really a measurement study, but rather a method, and thus the evaluation does not provide any real-data analysis or new insights.

However, I still think that the application presented in this paper is interesting and well designed and implemented. It shows another cool use for GPUs in a networking setup.

The use of GPUs to index traffic seems at once natural and also not quite trivial. The performance improvements shown are impressive.

However, as a non-expert in GPUs, I have three complaints:

- ❖ Section 3.2 is hard to follow. It plunges into details without giving first any high-level intuition behind the various algorithm steps.
- ❖ I was not able to form a sense of the design space. At a superficial level, I realize that the hard part about deploying sophisticated processing on GPUs is to produce an algorithm that can run without CPU help. But I was not able to map this high-level point down to specific design choices made by the authors. In the end, I could not form a clear picture of how an indexing implementation for GPUs fundamentally differs from an indexing implementation for CPUs. As a result, I could not tell what alternative choices the authors could have made (if there are any) and why those would be worse than what they ended up doing. If the authors care for their paper to be accessible to more than GPU and packet-processing experts, I recommend that they get feedback on presentation from such an audience.
- ❖ In the evaluation section, what is the rationale behind comparing the particular GPU against the particular CPU? For a performance comparison to make sense, two systems must have some common denominator, e.g., the same amount of processing power or the same cost. Which is the common denominator in this case? Perhaps price and/or off-the-shelf availability?

A minor issue: In Section 2, second paragraph from the end, "the current chunk identifier is compared to the identifier of the last literal..." How come a chunk identifier is compared to a literal identifier? I thought a chunk consisted of many literals... Btw, the entire paragraph is way too dense to read without stress.

4. Summary from PC Discussion

This paper seems to be liked by most of us, so I believe we will have an easy decision on this one.

Strengths:

- ❖ The paper addresses an important and timely problem
- ❖ A new and cool usage for GPUs for indexing packets in real time.

Weaknesses:

- ❖ Evaluation is somewhat lacking.
- ❖ Potentially out of scope for IMC

5. Authors' Response

We would like to thank the reviewers for their useful feedback and for the opportunity to answer. These are the changes we introduced to address the precious comments:

- ❖ We have modified the introduction to include the suggested references.
- ❖ We have introduced a new sentence in the first paragraph of Section 3.2 to provide the intuition of our approach. GPUs provide high-speed integer sorting performance, and this is the capability we use to be able to process all the bitmap index columns in parallel. This also represents the main difference between CPU- and GPU-based indexing.
- ❖ For the evaluation we indeed compare a GPU with a CPU of similar price. Additionally, we choose a CPU that could provide the highest single-thread performance in that price range.
- ❖ Regarding the issue in Section 2, we modified the sentence to clarify this issue "Before appending a literal to its corresponding column". In fact, a chunk of 31 consecutive values can result in up to 31 literal symbols, or, differently said, can cause the update of up to 31 distinct bitmap index columns. Each update consists of appending exactly one literal to its corresponding column.