Benchmarking network processors

Andreas Ehliar and Dake Liu
Dept. of Electrical Engineering
Linköping University
S-581 83 Linköping, Sweden
{ehliar,dake}@isy.liu.se

Abstract

In this paper the need for standardized benchmarks for network processors are discussed. Benchmarks are important for developers of high performance network related hardware who need to decide upon which Network Processor to use. The difference between application level benchmarks and micro level benchmarks are discussed. High quality micro level benchmarks are proposed as a way of easily assembling a large set of benchmark results that are relevant for a wide variety of applications. Three typical systems are proposed and network related tasks that could be elaborated into useful microbenchmarks are identified on these systems.

1. Introduction

Network bandwidth has increased dramatically during the last few years and the available bandwidth is still increasing. The computational load of protocol processing have increased correspondingly for all systems connected to a network.

Systems with only one connection to the network, *network terminals*, have kept up with the load by virtue of ever increasing CPU speeds. But even today, network terminals with a network connection in the gigabit range have trouble keeping up with the protocol processing load. Network interfaces designed for terminals have few, if any features to offload the protocol processing from the host CPU. Typically, the network interface might incorporate easily isolated tasks like hardware accelerated checksum computation.

Core routers are used on the backbone of the Internet to direct network traffic to the correct destination. These routers are connected to extremely high bandwidth links. As the Internet is growing, the routing tables have increased as well. To cope with the performance demands, specialized hardware is used in core routers. The critical aspects of the hardware is usually hardwired and the flexibility is low.

The situation is much worse for routers. The bandwidth have increased, the number of nodes on the Internet have increased dramatically, giving rise to larger routing tables. Specialized high performance *core routers* use custom hardware that is highly optimized for packet forwarding at high speed. This hardware is usually hardwired and the flexibility is low.

Edge routers are routers used on borders between the Internet backbone and smaller networks. Large corporate networks might use this kind of router internally as well. The flexibility demands on edge routers are high but the bandwidth requirements are not as high as for core routers. An edge router might be required to do firewalling or guarantee a certain bandwidth or latency to certain users.

A brief summary of the amount of specialized hardware in the various devices is shown in table I.

To cope with the high computational load required to deal with the bandwidth and flexibility requirements, specialized architectures have emerged. These architectures are called Network Processors. These usually offer a flexible programming model to ease product development.

Network processors is a relatively new field of research. There are a wide variety of different architectures available aimed at different market segments. A network processor can be a fairly standard microprocessor with a few extra instructions added to improve the performance of protocol processing or a complete custom architecture with several cores that operate in parallel. In some cases each core might be heavily optimized for a certain operation like pattern matching or encryption.

The widely varying nature of current network pro-

Network Device	Acceleration	Flexibility
Core router	High (hardwired)	low
Edge router	High	Moderate
Network Terminal	Very limited	Very high

Table I. Protocol offload in different devices

cessors makes it difficult to compare them. The architectures are very different and it is hard to make a reasonable performance estimation by just looking at the architecture description. In order to make reasonable buying decisions it is necessary to compare network processors against each other to discover the individual advantages and disadvantages.

One way of doing this comparison is to perform benchmarks. A good benchmark will have the following properties[1]:

- Linearity: If the performance of one machine is three times better than another machine, the benchmark score for the first machine should be three times higher than the score of the second machine.
- Reliability: If the benchmark score for one machine is higher than the benchmark score for another machine, the performance of the first machine should always be higher for the benchmarked metric.
- Repeatability: The benchmark score should be the same if the benchmark is repeated.
- Easiness of measurement: If it is not easy to measure it is unlikely that anyone is interested in the result. There is also a higher probability of errors in the benchmark itself.
- Consistency: The definition of the performance metric is valid on different machines.
- Independence: The benchmark has not been influenced by input from manufacturers wishing to highlight the best parts of their products.

Constructing good benchmarks for network processors is hard as network processing is inherently dependant upon the contents of the packets coming from the network. This makes it hard to construct a fair benchmark. Existing benchmarks deal with this problem in different ways, for example by running the benchmark with only certain packet sizes or a certain mix of packet sizes. It would be good if the performance of a network processor could be measured independent of the network traffic.

The rest of this paper is organized as follows, section 2 discusses application versus microbenchmarks, section 4 discusses a number of network related benchmarks, section 3 discusses the advantages of microbenchmarks, section 5 proposes a number of possible microbenchmarks and section 6 provides the conclusion of this paper.

2. Benchmark categories

Most engineers will tell you that the best way to benchmark a system is to benchmark the intended application running on the intended system. As this is not very practical for a more generic benchmark, approximations must be made. There are basically two

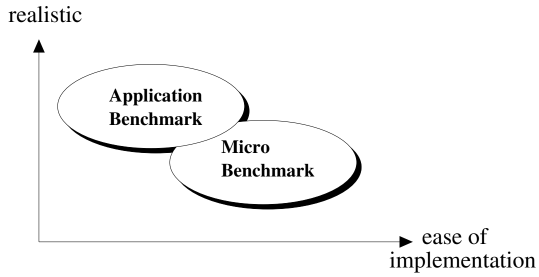

Figure 1. The relation between application level benchmarks and microbenchmarks

practical approaches to benchmarks:

- Benchmark typical applications
- Benchmark only one aspect of the system at a time.

An application is something which is useful in itself while a task is something which is not useful in and of itself but rather a specific part of a larger application. Routing is a typical network processing application while the queue management in a router is a typical task.

2.1. Benchmarking an application

Benchmarking a typical application is ideal if the application you are interested in is similar to the benchmarked application. But the results are not very useful if the intended application is not similar to the benchmarked application. Examples of application level benchmarks include the 3Dmark[4] suite for Windows that is supposed to benchmark the performance of a 3D accelerator, or the routing benchmarks from the Network Processor Forum[3]. A huge drawback of this approach is that it is quite time consuming to implement a huge number of applications as optimized as possible on several different systems.

2.2. Microbenchmarks

Another way to measure performance is to identify typical tasks and measure the performance of these in microbenchmarks. Examples of this include BDTI's[5] DSP benchmarks and the SPEC CPU benchmarks. The benchmarks are intended for users who are aware of the tasks involved in a certain application. Microbenchmarks are useful in this case, not to decide once and for all which system to use, but to indicate a number of processors that are interesting. A more thorough investigation about these processors can then be performed. The main drawback of this kind of benchmark is of course that it is hard to find small and yet representative tasks. Another drawback is that it is easy to draw the wrong conclusions from microbenchmarks if the intended application is not well understood. The main advantage is that it is reasonable to benchmark a wide variety of systems because the benchmarks themselves are not that hard to create.

3. Motivation for microbenchmarks

Figure 1 is an illustration of the relationship between application level benchmarks and microbenchmarks. A microbenchmark might in a few situations be a stand alone application, but most applications are too large to be considered as a microbenchmark.

This compactness of a microbenchmark makes it possible to collect a wide set of microbenchmarks which in the end might be more useful for most users than a few application benchmarks.

For example, a developer wishes to construct a hypothetical product aimed at broadband providers:

- The product should act as a router and support around 500 users with an uplink capacity of 2 gigabit/s.
- Each user should be able to have individual firewall settings. These settings are configurable through a web browser.
- The product should do transparent HTTP proxying and caching to improve the response time of popular websites.

A reasonable configuration but not common enough to warrant a special benchmark. But the developer might look at microbenchmarks that are important to the performance of a packet filtering firewall and benchmarks that are important for the performance of TCP handling. This is the main virtue of microbenchmarks, that a single representative microbenchmark can be useful in many contexts.

4. Available benchmarks

There are already a host of network related benchmarks. Most are aimed at benchmarking applications, like SPECweb99[2] which is a benchmark to measure the performance of webservers. However, network processors are not used in the context of applications like webservers. Network processors are used in applications like routing, TCP acceleration, firewalls, packet classification and network address translation. This kind of application will be the focus of this paper henceforth.

4.1. Network Processor forum

The benchmarks at the network processor forum[3] are application benchmarks aimed at measuring the performance of core routers. The benchmarks are performed via a traffic generator and a traffic analyzer on real hardware. Among the benchmarked values are:

- Aggregate forwarding rate
- Throughput
- Latency
- Power consumption

	General Purpose	DSP	Network Processor
Application benchmarks	Yes	Yes	Yes
Microbenchmarks	Yes	Yes	No

Figure 2. A comparison of the availability of standardized benchmarks for different platforms

4.2. Embedded microprocessor benchmark consortium

The embedded microprocessor benchmark consortium has a limited number of network related benchmarks[6]. These are mainly aimed at measuring the performance of a general purpose microprocessors used as a router.

4.3. Microbenchmarks for Network Processors

In figure 2 the availability of more or less standardized application and microbenchmarks for different platforms are shown. To date, there are no standardized microbenchmarks available for network processors. There are a few proposed microbenchmarks, like CommBench[7], NpBench[8] and NetBench[9] that try to identify typical tasks for a network processor. But this research area is far from mature.

5. What to benchmark

It is important to figure out a well balanced set of microbenchmarks. The benchmarks should cover a large number of cases that is relevant to most network applications. Some very common operations, like the Ethernet CRC calculation, might not be very useful to benchmark as the network interface most likely already contain an accelerator for such tasks. This section tries to partition the network processing into different core tasks suitable for microbenchmarks. Table II lists a proposed set of core tasks related to a number of different network devices. Further work is necessary to create well defined microbenchmarks out of these tasks.

Network processors are potentially useful in different devices. In this section the tasks of three typical systems are analyzed to collect a set of core tasks

Network device	Core task
Router	Validate IP packet Route lookup Queue management
Firewall	Packet classification Connection tracking Content classification Validate IP packet Queue management (Route lookup)
Server	TCP offload Encryption Queue management

Table II. Core tasks of different network devices

that would be interesting as microbenchmarks. Microbenchmarks for these devices will cover the most important tasks of network processing.

5.1. Router

The workload for a core router basically consist of receiving the packet, validating it as a valid IP-packet, doing a route lookup, updating the TTL of the packet and enqueuing the packet on the correct output queue.

An edge router might also be required to reserve bandwidth for certain users or guarantee a certain upper bound on the latency experienced on certain connections. This requires advanced queue management algorithms.

5.2. Firewall

A firewall will have the same workload as a router in principle, although the routing decisions might be much easier as a firewall is often employed in a location where it only has to know about a few subnets and only one default route to the Internet. In addition to the tasks performed by a router however, a firewall has to be able to defragment fragmented IP packets, look into the firewall ruleset to determine if a packet should be dropped and possibly keep track of established connections. An advanced firewall might also have to look at the payload of the packets and drop packets that contain for example the signature of a virus or a worm.

5.3. Server

A server is used to represent network terminals as the network processing needs of a client are very similar to that of a server. A server in this context could be a fileserver or a webserver for example. A lot of the tasks associated with such a server is not network related, such as for example database queries. Network processing related tasks can still be accelerated however, like encryption or TCP processing. Most servers of today do not have any hardware to do anything but the most simple of protocol offload though.

6. Conclusion and further work

In this paper I have discussed the importance of having standardized microbenchmarks for network processors. While application level benchmarks are important, microbenchmarks are a useful tool to cover a large variety of applications with a small set of benchmarks.

Having such benchmarks will make it easier to compare the performance of different network processors. This will make it easier for developers to decide upon which network processor to use. Three typical systems have been proposed and important network related tasks have been identified on these systems. Further work is necessary to actually distill the proposed microbenchmarks into a well defined set of benchmarks.

References

[1] Lilja, David J, *Measuring Computer Performance: A practitioner's guide* Cambridge pp. 10-12

[2] *SPECweb99* http://www.specbench.org/web99/

[3] Network Processing Forum, *Benchmarking implementation agreements*
http://www.npforum.org/techinfo/approved.shtm#ias

[4] *3DMARK03*
http://www.futuremark.com/products/3dmark03/

[5] Berkeley Design Technology, Inc. *Evaluating DSP Processor Performance*
http://www.bdti.com/articles/benchmk_2000.pdf

[6] Embedded microprocessor benchmark consortium, *Netmark* http://www.eembc.hotdesk.com/

[7] Wolf, T.; Franklin, M., *CommBench-a telecommunications benchmark for network processors* Performance Analysis of Systems and Software, 2000. ISPASS. 2000 IEEE International Symposium on, Vol., Iss., 2000

[8] Byeong Kil Lee; John, L.K., *Npbench: a benchmark suite for control plane and data plane applications for network processors* Computer Design, 2003. Proceedings. 21st International Conference on, Vol., Iss., Oct. 13-15, 2003

[9] Memik, G.; Mangione-Smith, W.H.; Hu, W., *NetBench: a benchmarking suite for network processors* Computer Aided Design, 2001. ICCAD 2001. IEEE/ACM International Conference on, Vol., Iss., 2001