Cross-Platform Estimation of Network Function Performance

Amedeo Sapio  
amedeo.sapio@polito.it

Mario Baldi  
mario.baldi@polito.it

Gergely Pongrácz  
ergely.pongracz@ericsson.com
Goal

Model network functions (NFs) to estimate their performance on specific platforms

- Platform independent representation
- Hardware characterization

Estimates computed automatically
Applications

• Orchestrator: deployment of VNFs
  – Efficient resources allocation
  – Steering of traffic among VNFs

• Scheduler: choice of execution platform
  – Optimize placement
Elementary Operations

- Generally most NFs perform a rather small set of recurring operations when processing the average packet
  - a well-defined alteration of packet headers, coupled with a data structure lookup.

- Elementary NF operations (EO): informally defined as the longest sequence of elementary steps (e.g., CPU instructions or ASIC transactions) that is common among multiple NFs processing tasks

- An NF can be modeled by splitting its functionality in EOs (hardware independent)
Performance estimation

- Each EO is mapped on the hardware component(s)/function(s) involved in its execution
  - Hardware platform specific

- Performance is measured for the specific EO on the specific component

- NF performance is obtained by composing performance of the individual EOs
The process

The execution can raise errors (if an EO is not supported in the HW architecture)
Preliminary list of EOs

- **Packet copy between I/O and memory**
  
  A packet is copied from/to I/O buffer to/from memory. 
  
  $L1n$ bytes are preferably stored in L1 cache, otherwise in L2 or external RAM. 
  
  $L2n$ bytes are preferably stored in L2 cache, otherwise in L3 or external RAM. 
  
  $\text{mem}_{I/O}(L1n, L2n)$

- **Parsing a data field of $b$ bytes from memory**
  
  $\text{parse}(b)$
Preliminary list of EOs

- *Direct access to a byte array in memory*
  
  Each entry has size $es$. The array has at most $max$ entries.
  
  $\text{array\_access}(es, max)$

- *Simple hash table lookup*
  
  A simple lookup in a direct, XOR based hash table.

  The hash key is made of $N$ components and each entry has size equal to $HE$. The table has at most $max$ entries. The collision probability is $p$.

  $\text{hash\_lookup}(N, HE, max, p)$
Preliminary list of EOs

- *Increase/Decrease a field of $b$ bytes* (must already be in a register)
  \[
  \text{increase}(b)
  \]

- *Checksum*
  Standard IP checksum computation performed on $b$ bytes.
  \[
  \text{checksum}(b)
  \]

- *Sum*
  Two operands of $b$ bytes are added
  \[
  \text{sum}(b)
  \]

...
Assumptions

• We assume that the number of registries is larger than the number of packet fields that must be processed simultaneously.

• We assume that the L1 cache is manually managed. This behavior is common for NPUs, while in CPUs there are some assembler instructions that can influence the cache logic.
Use Case: Ethernet Switch

- $\text{mem}_I/O(14, 1486)$ (*header 14B, 1500 max pkt size*)
- $\text{parse}(6)$ (*DMAC*)
- $\text{hash\_lookup}(1, 12, 2M, 0)$
  - 1 key component
  - 12 bytes entry (*key 6B, value 6B*)
  - Max 2M entries
  - Collision probability negligible
- $\text{mem}_I/O(14, 1486)$ (*header 14B, 1500 max pkt size*)
Simple NPU/CPU model

- I/O
  - total bandwidth
  - supported configurations
- Accelerators
  - (e.g., RE engines, TCAM, hw queue, encryption)
  - instructions per sec
  - number of units
  - frequency
- Processing unit(s)
  - (e.g., pipeline, execution unit)
  - features
  - size
  - deployment (shared, per-unit)
- On-Chip memory
  - (e.g., cache, scratchpad)
  - transactions per sec
  - size
  - deployment (shared, per-unit)
- External Resource Control
  - (e.g., optional TCAM, external memory)
  - transactions per sec
  - supported configurations
- Internal bus
  - total bandwidth

Source:
Intel Xeon E5-2630 (Sandy Bridge)

- **I/O**
  - 2x 10 GbE
  - 5 Gtps
  - PCIe v2.0 (x8)
  - Max 32 Gbps

- **PCIe v3.0**
  - 8 Gtps
  - 126 Gbps (x16)

- **AVX**
  - VT-d, VT-x + EPT

- **MCT**
  - 4 channels
  - DDR3
  - Max 340.8 Gbps
  - CAS lat. 9

- **DDR3**
  - 1333 Mtps
  - Max 85.2 Gbps

- **L1**
  - per core
  - i=32KB
  - d=32KB

- **L2**
  - per core
  - 256 KB

- **L3**
  - per slot
  - 15 MB

- **x86**
  - 64-bit architecture
  - 6 cores / slot
  - 2 threads / core
  - 2.3 – 2.8 GHz
  - AVX
  - VT-d, VT-x + EPT

- **L1**
  - per core
  - i=32KB
  - d=32KB

- **L2**
  - per core
  - 256 KB

- **L3**
  - per slot
  - 15 MB
Intel Xeon E5-2630: EO Performance

- \text{mem\_I/O}(L1n, L2n)

\[
4 \times \left[ \frac{\min(32KB, L1n)}{64B} \right] + 12 \times \left[ \frac{\min\left(256KB, \max(0, L1n - 32KB) + L2n\right)}{64B} \right] \\
+ 40 \times \left[ \frac{L1n + L2n}{64B} \right] \text{ clock cycles} \\
+ \left[ \frac{\max\left(0, \max(0, L1n - 32KB) + L2n - 256KB\right)}{64B} \right] \text{ L3 or DRAM accesses.}
\]

- \text{parse}(b)

\[
4 \times \left[ \frac{b}{8B} \right] \text{ clock cycles} \left\{ + \left[ \frac{b}{64B} \right] \text{ L3 or DRAM accesses} \right\} \text{ if in L1} \\
12 \times \left[ \frac{b}{8B} \right] \text{ clock cycles} \left\{ + \left[ \frac{b}{64B} \right] \text{ L3 or DRAM accesses} \right\} \text{ if in L2}
\]
Intel Xeon E5-2630: EO Performance

- $\text{increase}(b) - \text{sum}(b)$
  $$\left\lfloor 0.33 \times \frac{b}{8B} \right\rfloor \text{ clock cycles}$$

- $\text{array\_access}(es, \text{max})$
  $$1 + \left\lfloor \frac{es}{8B} \right\rfloor \text{ clock cycles}$$
  $$+ \left\lfloor \frac{es}{64B} \right\rfloor \text{ DRAM accesses}$$
Intel Xeon E5-2630: EO Performance

- **hash_lookup(N, HE, max, p)**
  \[
  \left(4 \times N + 106 + 4 \times \left\lceil \frac{\text{HE}}{8B} \right\rceil + 4 \times \left\lceil \frac{\text{HE}}{32B} \right\rceil \right) \times (1 + p) \text{ clock cycles}
  \]
  \[
  + \left\lceil \left(\left\lceil \frac{\text{HE}}{64B} \right\rceil \times (1 + p) \right) \right\rceil \text{ DRAM accesses}
  \]

- **checksum(b)**
  \[
  7 \times \left\lceil \frac{b}{2} \right\rceil + 8 \text{ clock cycles}
  \]
  \[
  + \left\lceil \frac{b}{64B} \right\rceil \text{ L3 or DRAM accesses}
  \]
Ethernet Switch on Intel Xeon E5-2630

- Resources required to process a 1500B packet:
  
  2630 clock cycles + 1 DRAM accesses

  a single core of an Intel Xeon E5-2630 at 2.8 Ghz can process
  $\approx 1.09 \text{ Mpps}$, while a DDR3 memory can support 70.16 Mpps.

  As a result a single core can process $\approx 12.95 \text{ Gbps}$

- Resources required to process a 64B packet:
  
  238 clock cycles + 1 DRAM accesses

  a single core of an Intel Xeon E5-2630 at 2.8 Ghz can process
  $\approx 12.05 \text{ Mpps}$, while a DDR3 memory can support 70.16 Mpps.

  As a result a single core can process $\approx 7.9 \text{ Gbps}$
Experimental evaluation: Throughput with one 10GbE NIC
Experimental evaluation: Throughput with bi-directional traffic using 2 cores

- 9% error
- 0.2% error
- 6% error

![Diagram showing throughput vs packet size with different error percentages for various estimates and benchmarks.]
Final remarks

• The model provides a good estimation of the maximum reachable limit of NFs performance.

• This methodology will be further improved considering also the effects of packets interaction and concurrence.

• The presence of accelerators does not affect performance in a simple use case as an Ethernet switch, but could be more relevant in more complex use cases (e.g. BNG).
Questions?