**Assignment 1** **Akshat Tandon**
CS4850 Performance Evaluation of Computer Systems 201503001

# Performance evaluation of a Warehouse Scale Computer

**Goal and System Definition** The goal of the paper[1] is to understand how do modern warehouse scale applications interact with the underlying hardware, for the purpose of future specializations and micro-architectural optimizations of server system-on-chips and processors. The system comprises of Google's cluster of interconnected server processors.

**Services offered** For this study, only a small subset of services offered by Google's servers are considered. These include (1)**ad-targeting services**; (2)**distributed storage system services**; (3)**gmail services**; (4)**indexing services**; (5)**search services**; (6)**video processing services**.

**Performance Metrics** The system is viewed from two different perspectives. The first is from an applications perspective and the other is from a micro-architecture perspective. Metrics used under application perspective are (1)**fraction** of WSC(Warehouse Scale Computer) **processor cycle**s spent on **top 50 application**s; (2)**fraction** of WSC **processor cycles** in different components of **datacenter tax**(inter application building blocks such as RPC, memory allocation, compression, serialization routines etc.) Under mirco-architecture perspective, a Top-Down performance analysis methodology is used. The metrics associated with such an approach are (1)**fraction** of **time spent by a micro-op**($\mu op$) in **Retiring**, **Bad-Speculation**, **Front-end bound** and **Back-end bound** phases of the $\mu op$ pipeline queue of the processor; (2)**fraction** of processor **cycles** completely **starved on front-end** bottlenecks(when the $\mu op$ queue is delivering 0 $\mu ops$ to the back-end); (3)instruction misses in the L2 cache(measured in **MPKI**, misses per kilo instructions); (4) **instruction cache working set** size of a specific application; (5) **IPC**(Instructions Per Cycle) of a specific application; (6) **time** spent serving a **data-cache request** of a specific application; (7) **ILP**(Instruction Level Parallelism) measured in terms of fraction of cycles utilizing (1-2), (3-4), (5-6) execution cores; (8)**memory**(DRAM) **bandwidth utilization**.

**Workload** C++ binaries of 12 major Google applications were considered as workload. These include binaries for ads, bigtable, gmail, search, indexing and videos. The criteria for selecting these binaries was diversity among the various workload categories(described below).

**System Parameters** The paper does not explicitly mention the server types and networking configurations of the Google WSC. However, it can be safe to assume that the servers under consideration are Intel Xeon Processor E7-4809 v2 since these are the only **ivy-bridge** processors having **six cores**(both of these facts were specified). Its major parameters are a 22nm fabrication, 12 threads, base frequency of 1.9GHz, bus speed of 6.4 GT/s QPI and a max memory bandwidth of 68 GB/s. Other parameters include the kernel overhead(core of OS) due to which a fifth of all WSC cycles are spent in the kernel.

**Workload Parameters** Workload parameters are mostly categorical. The major categories and their associated workloads are (1) **batch based**: video; (2) **latency-conscious**: all workloads except video; (3)**low-level services**: disk, bigtable; (4)**back-ends**: gmail, search; (5)**front-end**:gmail-fe; different degrees of (6)**data-cache pressures**; (7)**front-end bottlenecks**; (8)**IPC**.

**Key Factors** As the system under study is a live WSC and not a controlled one, there are a very few factors which can be tuned. Chief among them is (1) performing a top-level analysis of $\mu op$ queue pipeline either with **simultaneous multi-threading(SMT) enabled or disabled**. Other factors such as (2) **Types of** application considered as **workload** and (3) **Types of** low-level, self-contained routines selected as **datacenter tax**, remain fixed throughout the measurements.

**Evaluation Technique** Performance evaluation technique used in both the application and microarchitecture perspectives is **measurement**. The measurements are taken over a period of three years.

**Experimental Design** Google Wide Profiling(GWP) tool is used for collecting performance related data as well as correlating the measurements with an application's call-stack. GWP is also used for collecting micro-architecture specific data by randomly selecting 20,000 ivy-bridge processors and then profiling all jobs running on them to gather 1 second samples of a few particular performance counters.

**Analysis Results/Insights** (1)There is **no single killer application** which dominates in terms of fraction of total WSC cycles consumed. It takes around 50 applications to cover 60% of total WSC cycles.(2) $22 - 27\%$ **of WSC cycles** are **spent in** different components of **datacenter tax**. Much of this tax can be reduced by having dedicated hardware on server *soc.*(3) Applications are having **severe instruction cache bottlenecks**, having misses in the range of 5-20 MPKI, an order of magnitude more than the worst cases in SPEC CPU2006. Also instruction cache working sets are 4-5 times larger than the largest in SPEC and are growing at a rate of 27.7% annually. (4) **Lower ILP(Instruction Level Parallelism)** as only 28% of cycles utilize 3 to 4 cores and less than 4% utilize 5 to 6 cores (6 being the maximum). (5) **Low memory**(DRAM) **bandwidth utilization**. An immediate consequence of this is that memory latency is more important for todays WSC applications. (6) **Simultaneous multi-threading** (SMT) **reduces front-end bound cycles** by 6%, **front-end starvation cycles** by 1% and **increases** the **throughput**(Instructions Per Cycle IPC) by a factor of 0.5

**Common Mistakes** (1) **Unrepresentative workload** due to consideration of only a **few major C++ binaries** as workload. For higher-level server-side code, Java is used extensively. Applications written in Java(and other languages) are completely ignored in this study. (2) **Only ivy-bridge processors are considered** for micro-architectural analysis. Google's WSC is huge and contains all sorts of processors. Restricting the study to only one form of architecture does not present a complete picture. (3) **Overlooks important system parameters** since the WSC server and network specifications as well as configurations are nowhere mentioned in the study. (4) **Ignores significant factors** such as speed of the network and size of the parameters of datacenter tax components.

**Resolving Mistakes** The above mistakes can be rectified by considering a workload which is more diverse in terms of programming languages, conducting measurements on different micro-architectures present in WSC and contrasting them with each other, stating explicitly the exact server and networking specifications as well as configuration settings, including factors such as speed of the network as well as parameters of datacenter tax components.

# Research papers on Computer Systems Performance Analysis


[1] Kanev, Svilen, Juan Pablo Darago, Kim Hazelwood, Parthasarathy Ranganathan, Tipp Moseley, Gu-Yeon Wei, and David Brooks. "Profiling a warehouse-scale computer." In Computer Architecture (ISCA), 2015 ACM/IEEE 42nd Annual International Symposium on, pp. 158-169. IEEE, 2015. `http://dl.acm.org/ft_gateway.cfm?ftid=1585104&id=2750392`

[2] Kanakala, V. RaviTeja, V. Krishna Reddy, and K. Karthik. "Performance analysis of load balancing techniques in cloud computing environment." In Electrical, Computer and Communication Technologies (ICECCT), 2015 IEEE International Conference on, pp. 1-6. IEEE, 2015. `http://www.iaescore.com/journals/index.php/IJEECS/article/download/4239/3504`

[3] Ahmed, Ejaz, Adnan Akhunzada, Md Whaiduzzaman, Abdullah Gani, Siti Hafizah Ab Hamid, and Rajkumar Buyya. "Network-centric performance analysis of runtime application migration in mobile cloud computing." Simulation Modelling Practice and Theory 50 (2015): 42-56. `https://umexpert.um.edu.my/file/publication/00005818_106003.pdf`

[4] Barbierato, Enrico, Marco Gribaudo, and Mauro Iacono. "Performance evaluation of NoSQL big-data applications using multi-formalism models." Future Generation Computer Systems 37 (2014): 345-353. `http://www.sciencedirect.com/science/article/pii/S0167739X14000028`

[5] Jouppi, Norman P., Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates et al. "In-datacenter performance analysis of a tensor processing unit." arXiv preprint arXiv:1704.04760 (2017). `https://arxiv.org/pdf/1704.04760`