References

[1] Intel and K. T. Nguyen, “Introduction to cache allocation technology in the intel® xeon® processor e5 v4 family,” https://software.intel.com/content/www/us/en/develop/articles/introduction-to-cache-allocation-technology.html, 2016. [Pages 1 and 8.]

[2] D. M. Harris and S. L. Harris, *Digital Design and Computer Architecture*, 2nd ed. Morgan Kaufmann, 2013. [Pages 6, 7, and 8.]

[3] Intel, “Quiet noisy neighbor with intel® resource director technology,” https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/intel-rdt-infrastructure-paper.pdf. [Pages xv, xvi, 8, 10, 14, 26, 65, 67, and 72.]

[4] P. Veitch, E. Curley, and T. Kantecki, “Performance evaluation of cache allocation technology for nfv noisy neighbor mitigation,” in *2017 IEEE Conference on Network Softwarization (NetSoft)*, 2017. doi: 10.1109/NETSOFT.2017.8004214 pp. 1–5. [Pages 8 and 10.]

[5] Intel, “User space software for intel(r) resource director technology,” https://github.com/intel/intel-cmt-cat, 2019. [Pages 8, 24, and 56.]

[6] A. Farshin, A. Roozbeh, G. Q. Maguire, and D. Kostić, “Make the most out of last level cache in intel processors,” 2019, https://doi.org/10.1145/3302424.3303977. [Page 11.]

[7] J. Seward, “bzip2 and libbzip2, version 1.0.8,” 2019, https://sourceware.org/bzip2/manual/manual.html#performance. [Page 14.]

[8] DB-Engines, “Db-engines ranking of key-value stores,” https://db-engines.com/en/ranking/key-value+store. [Page 14.]

[9] Redis, “Redis,” https://redis.io/. [Page 14.]

[10] Graph 500 Steering Committee, “Benchmark specification | graph500,” 2017, https://graph500.org/?page_id=12. [Page 14.]

[11] D. Black, “The case for the graph 500 – really fast or really productive? pick one,” *The Exascale Report*, 2012, https://insidehpc.com/2012/03/the-case-for-the-graph-500-really-fast-or-really-productive-pick-one/. [Page 14.]

[12] “Graph500 GitHub Repository,” https://github.com/graph500/graph500. [Page 15.]

[13] “STREAM GitHub Repository,” https://github.com/jeffhammond/STREAM. [Page 18.]

[14] Intel, “Intel® xeon® scalable processors,” https://www.intel.com/content/www/us/en/products/details/processors/xeon/scalable.html. [Page 21.]

[15] CPU-World, “Intel xeon 8275cl specifications,” https://www.cpu-world.com/CPUs/Xeon/Intel-Xeon%208275CL.html, 2020. [Page 22.]

[16] WikiChip, “Cascade lake - microarchitectures - intel,” https://en.wikichip.org/wiki/intel/microarchitectures/cascade_lake, 2021. [Page 22.]

[17] N. Jouppi, “Improving direct-mapped cache performance by the addition of a small fully-associative cache and prefetch buffers,” in *[1990] Proceedings. The 17th Annual International Symposium on Computer Architecture*, 1990. doi: 10.1109/ISCA.1990.134547 pp. 364–373. [Page 22.]

[18] L. Backes and D. A. Jiménez, “The impact of cache inclusion policies on cache management techniques,” in *Proceedings of the International Symposium on Memory Systems*, ser. MEMSYS ’19. New York, NY, USA: Association for Computing Machinery, 2019. doi: 10.1145/3357526.3357547. ISBN 9781450372060 p. 428–438. [Online]. Available: https://doi.org/10.1145/3357526.3357547 [Page 22.]

[19] Linuxtopia, “isolcpus scheduler options,” https://www.linuxtopia.org/online_books/linux_kernel/kernel_configuration/re46.html. [Page 23.]

[20] Intel, *Intel® 64 and IA-32 Architectures Software Developer's Manual*. Intel, 2016, vol. 2 (2A, 2B, 2C & 2D): Instruction Set Reference, A-Z, https://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-instruction-set-reference-manual-325383.pdf. [Page 23.]

[21] ——, "Pqos manual page," https://github.com/intel/intel-cmt-cat/blob/master/pqos/pqos.8, 2020. [Page 24.]

[22] Perf Wiki, "perf: Linux profiling with performance counters," 2020, https://perf.wiki.kernel.org/index.php/Main_Page. [Page 24.]

[23] ——, "Linux kernel profiling with perf," 2015, https://perf.wiki.kernel.org/index.php/Tutorial. [Page 25.]

[24] Redis, "How fast is redis?" https://redis.io/topics/benchmarks. [Page 28.]

[25] Intel, "Intel® resource director technology (intel® rdt)," https://www.intel.com/content/www/us/en/architecture-and-technology/resource-director-technology.html. [Page 54.]

Appendix A

Reproducibility Appendix

A.1 bzip2 Experiment Command Listings

Listing A.1: Command used to generate ∼1.3GB random data to compress by bzip2. Based on [3].

```
openssl rand -out /dev/shm/test1 -base64 $((2**30))
   $
```

Listing A.2: Command used to run bzip2. Based on [3].

```
time bzip2 -c9 < /dev/shm/test1
```

Listing A.3: Command used to run STREAM for the bzip2 experiment. Based on [3].

```
stream_c.exe > /dev/null &
```

Listing A.4: Commands used to run Intel CMT for the bzip2 experiment.

```
# Running CMT on CPU core 9 collecting metrics for
# bzip2 running on core 10.
# Output is written in CSV format to file.
taskset -ac 9 sudo pqos -m llc:10 -u csv -o <output
    file> &

# Running CMT on CPU core 9 collecting metrics
# for the 13 STREAM processes running on cores
   11-23.
taskset -ac 9 sudo pqos -m llc:[11-23] -u csv -o <
   output file> &
```