R.I.P. Geomean Speedup

Use Equal-Work (or Equal-Time) Harmonic Mean Speedup Instead

Lieven Eeckhout (Ghent University)

Abstract—How to accurately summarize average performance is challenging. While geometric mean speedup is prevalently used, it is meaningless. Instead, this paper argues for harmonic mean speedup which accurately summarizes how much faster a workload executes on a target system relative to a baseline. We propose the equal-work and equal-time harmonic mean speedup metrics to explicitly expose the different assumptions they make, and we further suggest that equal-work speedup is most relevant to computer architecture research. The paper demonstrates that which average speedup is used matters in practice as inappropriate averages may lead to incorrect conclusions.

1 Introduction

Measuring computer system performance is fundamental to research and development. Speedup is arguably the most frequently used performance metric to quantify relative performance differences. How to summarize speedup across a set of benchmarks has led to a vivid, multi-decade debate. While some argue for geometric mean speedup, others argue for harmonic mean speedup, and yet others argue, under specific (perhaps hypothetical) circumstances, for arithmetic mean speedup. Geometric mean speedup is prevalent: most computer architecture research articles use geometric mean speedup to summarize performance.

In this paper, we argue that geometric mean speedup, or geomean speedup for short, lacks physical meaning and leads to misleading and incorrect conclusions in contrast to harmonic mean speedup which accurately reports how much faster a workload executes on average on a system of interest relative to a baseline system. We demonstrate that it matters in practice: using SPEC CPU2017 performance results we identify (many) examples where geomean speedup reports that system A outperforms system B by a non-negligible margin, while harmonic mean reaches the opposite conclusion.

This paper proposes the equal-work (EWS) and equal-time (ETS) harmonic mean speedup metrics to comprehensively summarize average performance. The meaning of EWS and ETS differs because of the difference in the underlying assumptions: EWS gives equal weight to the amount of work done by each benchmark, while ETS gives equal weight to the amount of time each benchmark spends on the baseline system. Because computer architecture performance analysis typically values the work done by each benchmark equally, EWS is the appropriate average speedup to use. This result calls for action from our community to (1) abandon geomean speedup, (2) use harmonic mean speedup instead, and (3) explicitly state whether EWS or ETS is used.

2 Background

While it is generally agreed upon that 'performance is not a single number', it is convenient to summarize performance across a set of benchmarks using a single performance number. We first revisit the different means.

2.1 Arithmetic and Harmonic Mean

Consider n measurements $M_i, 1 \leq i \leq n$, typically representing a metric of interest M for n different workloads or benchmarks. The *arithmetic mean (AM)* is defined as

$$AM(M) = \frac{1}{n}\sum_{i=1}^{n} M_i, \tag{1}$$

while the *harmonic mean (HM)* is defined as

$$HM(M) = \frac{n}{\sum_{i=1}^{n}\frac{1}{M_i}}. \tag{2}$$

John [6] provides a comprehensive and mathematically sound discussion when to use the arithmetic versus harmonic mean for metrics M that are a ratio of other metrics A and B, i.e., $M = A/B$. Many metrics used in computer architecture research and development are ratios, e.g., cycles per instruction (CPI), million instructions per second (MIPS), etc. For a metric $M = A/B$ with equal Bs across the benchmarks, it turns out that the arithmetic mean is the correct mean:

$$\begin{aligned}\frac{\sum_{i=1}^{n} A_i}{\sum_{i=1}^{n} B_i} = \frac{\sum_{i=1}^{n} A_i}{n \cdot B} &= \frac{1}{n}\sum_{i=1}^{n}\frac{A_i}{B} \\ &= \frac{1}{n}\sum_{i=1}^{n}\frac{A_i}{B_i} = AM(A/B).\end{aligned} \tag{3}$$

If on the other hand, the As are equal across the benchmarks, the harmonic mean is the correct mean:

$$\begin{aligned}\frac{\sum_{i=1}^{n} A_i}{\sum_{i=1}^{n} B_i} = \frac{n \cdot A}{\sum_{i=1}^{n} B_i} &= \frac{n}{\sum_{i=1}^{n} B_i/A} \\ = \frac{n}{\sum_{i=1}^{n}\frac{1}{A/B_i}} &= \frac{n}{\sum_{i=1}^{n}\frac{1}{A_i/B_i}} = HM(A/B).\end{aligned} \tag{4}$$

Based on these derivations it is clear that depending on the performance metric of interest, one has to choose for either the harmonic or arithmetic mean. In particular, whether the As or Bs are equal determines what mean to use. For example, if instruction count is constant across a set of benchmarks, average IPC (instructions executed per cycles) needs to be computed using the harmonic mean, while average CPI (cycles per instruction) is obtained using the arithmetic mean.

In practice though, the As or Bs are not necessarily equal, e.g., instruction count may differ across benchmarks. In such cases, one needs to resort to weighted averages. The *weighted arithmetic mean (WAM)* is defined as

$$WAM(M) = \sum_{i=1}^{n} w_i \cdot M_i, \tag{5}$$

and the *weighted harmonic mean (WHM)* is defined as

$$WHM(M) = \frac{1}{\sum_{i=1}^{n} \frac{w_i}{M_i}}, \tag{6}$$

with w_i the weights such that $\sum_{i=1}^{n} w_i = 1$. John [6] demonstrates that weighted arithmetic or harmonic mean can be used interchangeably provided that the appropriate weights are applied. In general, for a metric $M = A/B$, the arithmetic mean with the weights of B is identical to the harmonic mean with the weights of A. For example, when computing average IPC using the weighted harmonic mean, one needs to weigh the IPC of benchmark i with its respective relative instruction count $w_i = I_i / \sum_{j=1}^{n} I_j$. Alternatively, one can compute the average IPC using the weighted arithmetic mean with the weights based on cycle count, i.e., $w_i = C_i / \sum_{j=1}^{n} C_j$.

2.2 Geometric Mean

The *geometric mean (GM)* is defined as

$$GM(M) = \sqrt[n]{\prod_{i=1}^{n} M_i}. \tag{7}$$

The geometric mean has an appealing property for rate-based metrics, namely the geometric mean of the ratios is the same as the ratio of the geometric means [5]:

$$GM(A/B) = \sqrt[n]{\prod_{i=1}^{n} \frac{A_i}{B_i}} = \frac{\sqrt[n]{\prod_{i=1}^{n} A_i}}{\sqrt[n]{\prod_{i=1}^{n} B_i}} = \frac{GM(A)}{GM(B)}. \tag{8}$$

2.3 The War of the Means

The debate started in 1986 with Fleming and Wallace [4] who argue in favor of the geometric mean when computing performance ratios or speedup. Smith [9] argues against the geometric mean for rate-based metrics and in favor of harmonic mean. John [6] provides a solid and comprehensive discussion when to use the (weighted) arithmetic versus harmonic mean depending on the metric. Mashey [7] uses a statistical argument in favor of geomean mean based on the assumption that performance results is typically log-normally distributed. Citron et al. [3] concludes that the average used has little impact on the overall conclusion. Hennessy and Patterson [5] use geometric mean throughout their seminal textbook.

2.4 Computing Average Speedup

While the war of means included a variety of metrics, we focus on speedup in this paper which is the single most frequently used metric to quantify relative performance differences. Speedup S is defined as execution time on the baseline system B divided by the time on the optimized system O:

$$S = \frac{T_B}{T_O}. \tag{9}$$

Speedup above one means higher performance, while speedup below one means the optimized system effectively yields a slowdown. Computing speedup for individual benchmarks is easy and uncontested, however, how to compute average speedup across a set of benchmarks is more controversial.

Geometric mean speedup is prevalent. Geomean speedup is used by benchmarking consortia including SPEC. In particular, SPEC CPU reports SPECratio or the geomean speedup of a machine relative to a reference machine, namely a Sun Ultra5_10 workstation with a 300 MHz SPARC processor and 256 MB main memory. To compute the relative performance difference between two machines one can simply divide the respective SPECratios — the execution time on the reference machine drops out meaning that the choice of the reference machine becomes irrelevant [5], which is convenient.

TABLE 1: Example illustrating why geomean speedup lacks physical meaning.

Benchmark	Execution time		Speedup
	baseline	*optimized*	
A	1	1	1
B	1	0.01	100
Geometric mean speedup			10
Harmonic mean speedup			1.98

Researchers also widely use geomean speedup. A sample survey of the 79 papers accepted at the premier 2023 IEEE/ACM International Symposium on Computer Architecture (ISCA) reveals that out of the 31 papers reporting average speedup across a set of benchmarks, 24 use the geometric mean, 6 do not specify which average is used, and only one paper uses harmonic mean.

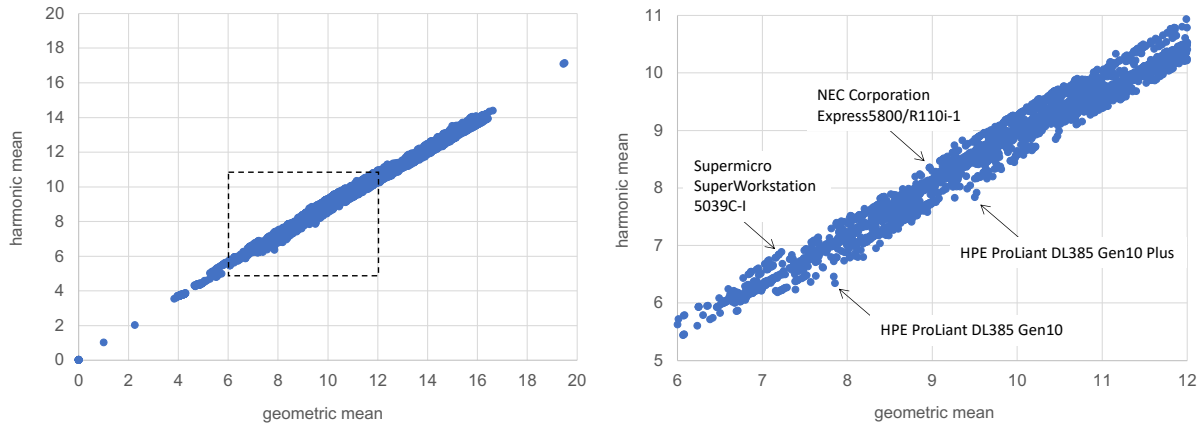
3 R.I.P. Geomean Speedup

Despite its convenience and widespread use, geomean speedup is meaningless. A speedup of a factor S, per its definition, implies that the optimized system is S times faster than the baseline system, i.e., the work gets done in S fewer time units. However, this is not what the geometric mean speedup computes. In fact, geomean speedup lacks physical meaning.

The example in Table 1 illustrates this. Consider two benchmarks A and B that run equally long on the baseline system, i.e., their normalized execution time equals one unit of time. Assume now that benchmark A runs equally fast on the optimized system, i.e., executing A on the optimized system also takes one unit of time. In contrast, benchmark B executes 100× faster on the optimized system, i.e., its execution time equals 0.01 time units on the optimized system. The geometric mean speedup across the two benchmarks equals $GM = \sqrt{1 \cdot 100} = 10\times$. Based on the definition of speedup, the intuitive understanding is thus that the optimized system reduces execution time by a factor 10. Unfortunately, this does not reflect reality. Executing benchmarks A and B on the baseline system takes 2 time units while taking 1.01 time units on the optimized system. The speedup hence equals $2/1.01 = 1.98\times$, not $10\times$!

In contrast, the harmonic mean speedup across the two benchmarks equals $HM = 2/(1/1 + 0.01/1) = 1.98\times$, which is exactly what the average speedup is supposed to report: it reports the reduction in execution time on the optimized system relative to the baseline system. In other words, harmonic mean speedup provides a precise physical meaning, namely it reports how much faster a workload executes on average on the optimized system relative to the baseline system.

The above example assumed that the execution on the baseline system is the same for each benchmark, which is why harmonic mean provides the right answer. Of course, in reality benchmarks execute for a different amount of time, implying that a weighted average needs to used, which is affected by the relative execution times of the benchmarks. This raises the fundamental question: how important are the relative execution times of the benchmarks on the reference machine? According to SPEC at least — and this is likely true in most benchmarking scenarios — the relative execution times are irrelevant, which implies that one should use the (unweighted) harmonic mean. The implicit assumption here is that the time spent on the baseline system is weighed equally for each of the benchmarks.



(a) All-results view (b) Zoomed-in view

Fig. 1: Harmonic mean (vertical axis) versus geometric mean (horizontal axis) of SPECratios for integer speed base SPEC CPU2017. *While the geometric and harmonic means strongly correlate, they may reach opposite conclusions for pairwise comparisons.*

4 Does It Really Matter?

One may (be tempted to) think that which mean to use does not have much impact in practice. To illustrate the contrary, consider the SPEC CPU2017 integer speed base results, downloaded on November 2, 2023 from https://www.spec.org/cpu2017/results/. The data set contains speedup numbers for the ten integer benchmarks for a total of 7,815 machines submitted over a period of about six years. Figure 1 reports the harmonic mean (vertical axis) versus geometric mean (horizontal axis) of SPECratios[1] for the complete data set (figure on the left) and for a subset of the data (figure on the right). There is clearly a strong correlation between the harmonic and geometric means, see Figure 1(a) which seems to align with the general findings of Citron et al. [3] that '*the choice of the mean used is of little consequence*'.

However, upon closer inspection, see Figure 1(b), when comparing specific systems, it is clear that the geometric and harmonic means do not necessarily agree on which system yields higher performance. For example, consider the HPE ProLiant DL385 Gen10[2] versus the Supermicro SuperWorkstation 5039C-I[3]. The HPE ProLiant achieves a 7.85× geomean speedup while the Supermicro SuperWorkstation achieves a 7.16× speedup. However, the harmonic mean reports the opposite: the HPE ProLiant achieves a 6.34× speedup versus 6.79× for the Supermicro SuperWorkstation. Note that the difference is non-negligible, i.e., 9% in one direction versus 7% in the opposite direction. Figure 1(b) illustrates another example where geomean reports that the HPE ProLiant DL385 Gen10 Plus[4] outperforms the NEC Corporation Express5800/R110i-1[5], while harmonic mean reports the opposite.

These cases — and there are many more examples — illustrate that how average speedup is computed really matters in practice: geomean speedup may reach an incorrect conclusion which could have been avoided by using the harmonic mean speedup. The implications could be severe, steering research and development in a sub-optimal direction. Or, more practically, one may be making mistakes when using SPEC's geomean speedup numbers to guide purchasing decisions.

1. Note that the harmonic mean of SPECratios differs from EWS and ETS because the ratio of harmonic means is not the same as the harmonic mean of ratios, as discussed later.
2. /cpu2017/results/res2019q3/cpu2017-20190903-17794.html
3. /cpu2017/results/res2019q2/cpu2017-20190430-13461.html
4. /cpu2017/results/res2020q2/cpu2017-20200330-21613.html
5. /cpu2017/results/res2018q4/cpu2017-20181029-09331.html

5 Equal-Work and Equal-Time Speedup

The above analysis advocates for harmonic mean speedup to summarize average performance across a set of benchmarks. An implicit assumption underpinning harmonic mean speedup though is that, as aforementioned, one weighs the time spent on the original system for each of the benchmarks equally. This implies that different benchmarks may execute a different amount of work, e.g., a memory-intensive application may execute fewer instructions than a compute-intensive application in the same amount of time. Another way to compute average performance using the harmonic mean speedup is to weigh the amount of work done for each of the benchmarks equally. This leads to the equal-work and equal-time harmonic mean speedup metrics we introduce and discuss now.

Assume we have n benchmarks executing a fixed number of instructions, e.g., 1 B instructions. Alternatively, the amount of work by each benchmark differs but the experimenter weighs the work done by each benchmark equally. Assume further we want to compare performance for the optimized system O relative to a baseline system B. For each benchmark we measure time in cycles, and we can thus compute $IPC_{B,i}$ and $IPC_{O,i}$ for each benchmark $i, 1 \leq i \leq n$. (If execution time is measured in seconds, one can compute *IPS* or instructions per second. For multi-threaded workloads for which IPC and IPS are harmful, one can use work-related metrics or more specifically work units per unit of time, e.g., transactions per second [1]. We consider *IPC* without loss of generality.)

We define and compute *equal-time harmonic mean speedup* or *equal-time speedup (ETS)* as follows. We first compute speedup S_i of the optimized system over the baseline system for each of the benchmarks:

$$S_i = \frac{IPC_{O,i}}{IPC_{B,i}}. \tag{10}$$

Note that this is equivalent to Equation 9 because instruction count is constant. Computing the harmonic mean over these speedup values yields equal-time speedup:

$$ETS = \frac{n}{\sum_{i=1}^{n} \frac{1}{S_i}}. \tag{11}$$

ETS computes per-benchmark speedup before computing the harmonic mean. Informally speaking, ETS is the 'harmonic

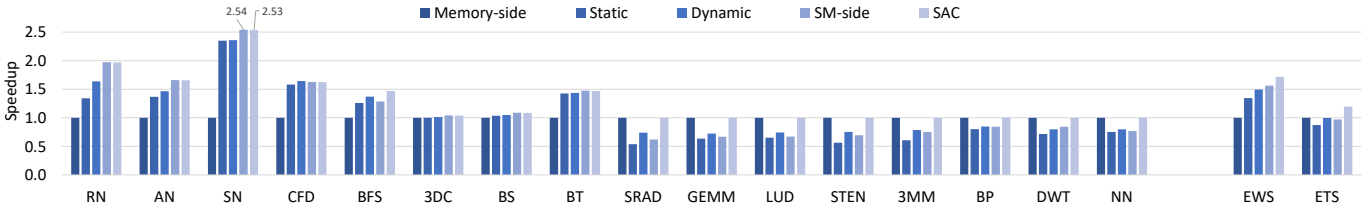


Fig. 2: Comparing different multi-chip GPU LLC organizations using the EWS and ETS averages. *EWS and ETS provide different average performance numbers, exposing the underlying difference in meaning and assumptions.*

mean of ratios', i.e., first compute ratios (speedups), then harmonic mean. By doing so, ETS gives equal weight to the time spent by each benchmark on the baseline system.

We define and compute *equal-work harmonic mean speedup* or *equal-work speedup (EWS)* as follows. We first compute the average IPC on the optimized system and the baseline system across all the benchmarks. Because instruction count is fixed, we have to use the harmonic mean:

$$IPC_B = \frac{n}{\sum_{i=1}^{n} \frac{1}{IPC_{B,i}}} \tag{12}$$

and

$$IPC_O = \frac{n}{\sum_{i=1}^{n} \frac{1}{IPC_{O,i}}}. \tag{13}$$

We subsequently compute the speedup of the optimized system over the baseline system to obtain equal-work speedup:

$$EWS = \frac{IPC_O}{IPC_B}. \tag{14}$$

EWS computes the harmonic mean IPC per system before computing speedup. Informally, one could say that EWS is the 'ratio of harmonic means', i.e., first compute harmonic means, then the ratio. Hence, EWS gives equal weight to the amount of work done by each benchmark.

Note that if one were to replace the harmonic mean with geometric mean in the above formulas, there would be no difference between ETS and EWS, because the geomean of the ratios equals the ratio of geomeans. This convenience presumably contributes to the popularity of the geomean speedup.

In contrast, ETS and EWS provide different numbers, and they should, because their meaning differs. Which metric to use, EWS or ETS, depends on the experimenter's perspective. If one considers the amount of work done by each benchmark to deserve equal weight, one should use EWS. If on the other hand, one considers the amount of time spent by each benchmark on the baseline system to deserve equal weight, one should use ETS. There is no right or wrong metric as it depends on the context. In any case, and at the very least, the EWS and ETS metrics clearly expose what the underlying assumptions are when reporting average speedup numbers.

A golden rule in computer architecture performance evaluation is to measure time to execute a unit of work, i.e., a complete benchmark or a representative region thereof [5]. When computing average computer system performance across a set of benchmarks, one thus makes the implicit assumption that the work done by each benchmark is valued to be equally important. This is made explicit when considering the same amount of instructions per benchmark. This suggests that EWS is the appropriate average speedup to use.

Case study. We now illustrate how EWS and ETS (may) lead(s) to different conclusions in practice. We consider the data set from Zhang et al. [10] in which different last-level cache (LLC) organizations are evaluated in the context of a multi-chip GPU system. The authors compared five LLC organizations: memory-side, static [2], dynamic [8], SM-side, and (the authors' own) sharing-aware caching (SAC). Figure 2 reports EWS and ETS for a collection of benchmarks normalized to the memory-side LLC. (Zhang et al. use the EWS metric although this is not explicitly mentioned in the paper.) It is remarkable that EWS and ETS lead to fairly different conclusions. In fact, for some of the LLC organizations, EWS and ETS disagree. In particular, EWS reports that the static, dynamic and SM-side LLCs outperform the memory-side LLC, while ETS reaches the opposite conclusion that memory-side LLC outperforms the static, dynamic (by a small margin) and SM-side LLCs. Both EWS and ETS agree that SAC outperforms the other LLC organizations, but the magnitude of the benefit varies significantly: 1.71× speedup for EWS versus 1.19× for ETS.

6 Conclusion

This article will hopefully convince researchers and performance analysts to finally abandon geomean speedup. The newly proposed equal-work and equal-time harmonic mean speedup metrics comprehensively summarize average performance by weighing work versus time equally, respectively. Given that the amount of work (rather than time on the baseline system) by each benchmark is typically valued equally, it is expected that equal-work (rather than equal-time) speedup is most appropriate and relevant.

References

[1] A. R. Alameldeen and D. A. Wood. IPC considered harmful for multiprocessor workloads. *IEEE Micro*, 26(4):8–17, July 2006.

[2] A. Arunkumar, E. Bolotin, B. Cho, U. Milic, E. Ebrahimi, O. Villa, A. Jaleel, C.-J. Wu, and D. Nellans. MCM-GPU: Multi-Chip-Module GPUs for Continued Performance Scalability. In *Proceedings of the IEEE/ACM International Symposium on Computer Architecture (ISCA)*, pages 320–332, 2017.

[3] D. Citron, A. Hurani, and A. Gnadrey. The harmonic or geometric mean: Does it really matter? *ACM SIGARCH Computer Architecture News*, 34(4):18–25, Sept. 2006.

[4] P. J. Fleming and J. J. Wallace. How not to lie with statistics: The correct way to summarize benchmark results. *Communications of the ACM*, 29(3):218–221, Mar. 1986.

[5] J. L. Hennessy and D. A. Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann Publishers, sixth edition, 2019.

[6] L. K. John. More on Finding a Single Number to Indicate Overall Performance of a Benchmark Suite. *ACM SIGARCH Computer Architecture News*, 32(4):1–14, Sept. 2004.

[7] J. R. Mashey. War of the benchmark means: Time for a truce. *ACM SIGARCH Computer Architecture News*, 32(4):1–14, Sept. 2004.

[8] U. Milic, O. Villa, E. Bolotin, A. Arunkumar, E. Ebrahimi, A. Jaleel, A. Ramirez, and D. Nellans. Beyond the Socket: NUMA-Aware GPUs. In *Proceedings of the IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 123–135, 2017.

[9] J. E. Smith. Characterizing computer performance with a single number. *Communications of the ACM*, 31(10):1202–1206, Oct. 1988.

[10] S. Zhang, M. Naderan-Tahan, M. Jahre, and L. Eeckhout. SAC: Sharing-Aware Caching in Multi-Chip GPUs. In *Proceedings of the IEEE/ACM International Symposium on Computer Architecture (ISCA)*, pages 43:1–43:13, 2023.