MIT INITIATIVE ON THE DIGITAL ECONOMY 2021, VOL. 6

HOW FAST DO ALGORITHMS IMPROVE?

By Yash Sherry, Neil C. Thompson

IN THIS BRIEF

One way to improve computer performance is to change their algorithms — the step-by-step procedures used by computers to solve problems.

While many claims have been made about the rapid pace of algorithmic progress, these claims have been largely anecdotal. This research reviewed data from 57 textbooks and nearly 1,140 research papers to provide the first comprehensive analysis of algorithmic progress.

Analysis reveals a wide discrepancy. On the one hand, roughly half of all algorithm families experience little or no improvement. On the other, 14% experience improvements so transformative, they radically change how and where the algorithms can be used.

The study also compared algorithmic progress with that of hardware, as expressed by Moore's Law, which states that processor power doubles every two years. Overall, the research finds that algorithmic progress for moderate-sized problems increased by less than Moore's Law, while for sufficiently big data problems it increased by more than Moore's Law.

Algorithms — the step-by-step procedures used by computers to solve problems — are among the central pillars of computer science. When algorithms improve, they enable scientists and others to use computers for tackling larger problems and exploring new domains

Many bold claims have been made about the pace of algorithmic progress, but these claims have often been based on either inadequate data or narrowly focused problems.

As a result, our knowledge of algorithmic progress has remained fragmented. Is progress faster for some algorithms than others, and if so, by how much?

To answer these and other related questions, we provide the first comprehensive analysis of algorithmic progress, considering data from 57 textbooks and 1,137 research papers. We look systematically at when specific algorithms were discovered, how they have improved, and how the scale of these improvements compares with other sources of innovation, including computer hardware.

Some prior research has quantified progress for particular algorithms, including maximum flow (Leiserson et al., 2020), Boolean satisfiability and factoring (Grace, 2013), and linear solvers (including Bixby, 2002; and Womble, 2004).

Other research has looked at progress on benchmarks such as computer chess ratings or weather prediction (including Thompson et al., 2020; and Hernandez and Brown, 2020). But these benchmarks are not strictly comparable to algorithms; they lack either mathematically defined problem statements or verifiably optimal answers. So what's the best way to assess progress?

RESEARCH METHODOLOGY

For this analysis, we categorize algorithms into families. Two or more algorithms can be considered members of the same family if they solve the same underlying problem. In theory, an infinite number of algorithm families could be created by subdividing existing domains so that special cases can be addressed separately. But these subdivisions do not present an asymptotic, or mathematically significant, improvement for the full problem, and therefore we exclude such special cases in our analysis.

To focus on algorithms that are particularly consequential, we limit our consideration to families considered important by the authors of the 57 textbooks we examined. Based on these inclusion criteria, we identify 113 algorithm families, by which we mean they solve the same underlying problem. Each family, in turn, comprises an average of eight algorithms.

We consider an algorithm to be an improvement if it reduces its family's worst-case asymptotic time complexity. Asymptotic time is useful analytical shorthand when discussing algorithms. Basically, given a sufficiently high number of inputs, if one algorithm has a higher asymptotic time complexity than another, it will take more steps to run.

Based on these criteria, we find 276 initial algorithms and subsequent improvements, giving us an average of 1.44 improvements for each initial algorithm.

We also consider algorithm discovery and improvement over time (*Figure 1*). For this analysis, we examine several factors, including:

- The number of new algorithm families discovered each decade
- The percentage of algorithm families that improved each decade

We also investigate what are known as time complexity classes. These provide a way of classifying algorithms by the number of operations they require. For example, nearly a third of algorithm families (31%) belong to the exponential complexity category, meaning that as their input size grows, they require at least exponentially more operations to complete. Over time, algorithms can move from one complexity class to another. This occurs mainly when algorithm designers find new, more efficient ways of implementing a particular algorithm.

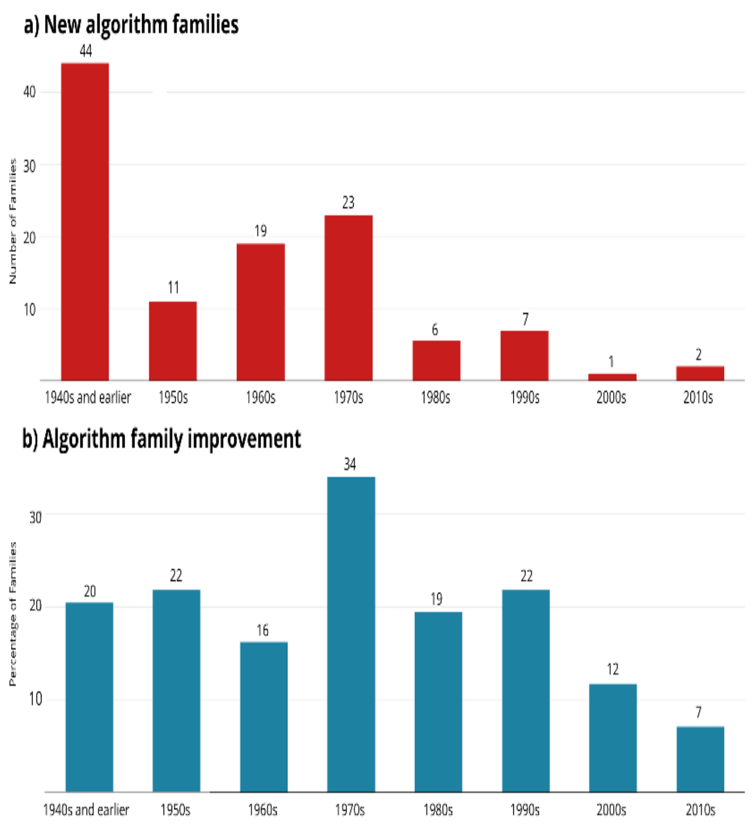

Fig. 1: Algorithm discovery and improvement. (a) Number of new algorithm families discovered each decade, in red. (b) Share of known algorithm families improved each decade, in blue.

Asymptotic time complexity is divided into classes, analogously to how biologists divide life into animals, plants, etc. These time complexity classes provide a way to classify algorithms by the number of operations they require. For example, nearly a third of algorithm families (31%) belong to the exponential complexity category, meaning that as their input size grows, they require at least exponentially more operations to complete. Over time algorithms can move from one complexity class to another. This occurs when algorithm designers find new, more efficient ways of implementing an algorithm.

Additionally, we compare algorithm improvements with innovations resulting from Moore's Law and other hardware advances. Moore's Law — originally formulated in the 1960s by Gordon Moore, a co-founder of Intel Corp. — observes that the overall processing power of computers roughly doubles every two years. Over time, the gains become exponentially far greater. While Moore's Law led to hardware improvements happening smoothly over time, algorithms experience large, but infrequent improvements.

including machine learning, cryptography, and databases. Next, for each of these subdomains, we analyzed 57 algorithm textbooks, some dating back to the 1960s. We used the authors' categorization of problems to determine both those algorithm families that are important to the field, and which algorithms correspond to each family.

In our analysis, we focus on exact algorithms with exact solutions. That is, we look only at cases that meet two criteria: First, the problem statement can be met exactly —

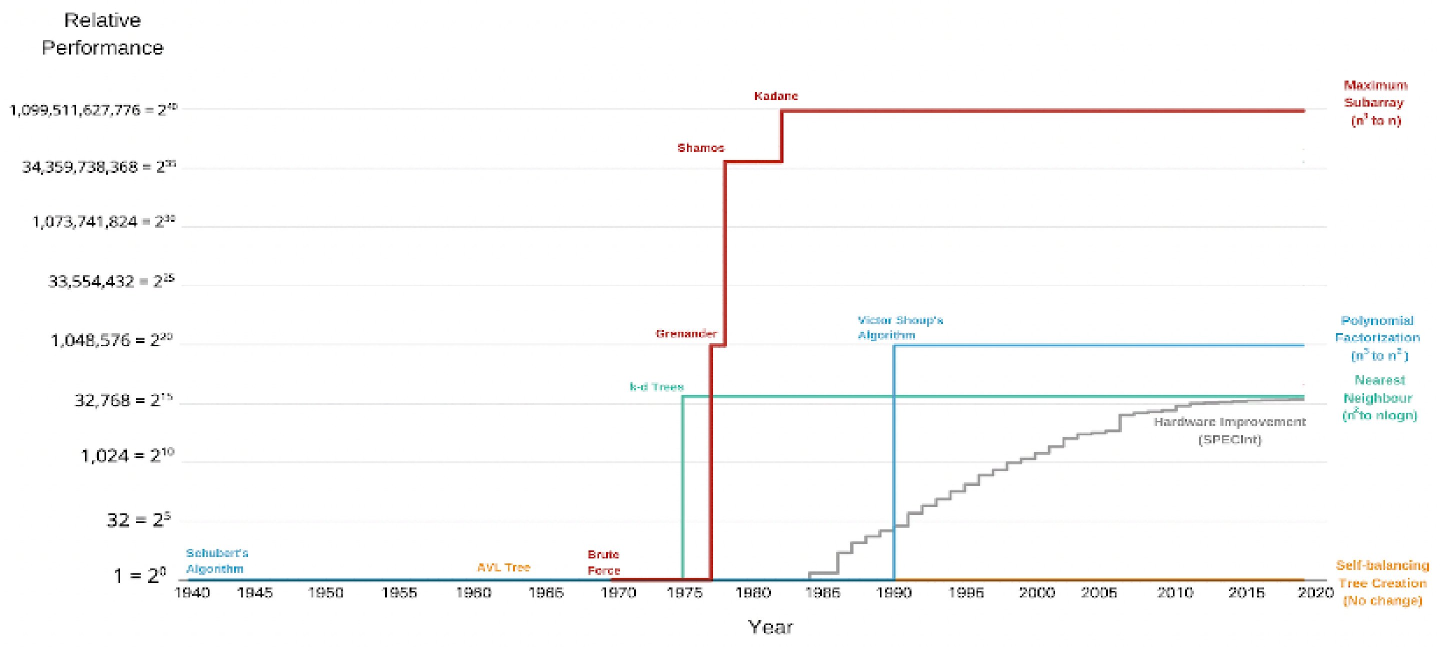

Fig. 2. Relative performance improvement for algorithm families, as calculated using changes in asymptotic time complexity. The comparison line is the SPECInt benchmark performance. Historical improvements for four algorithm families are compared with the first algorithm in that family (n = 1 million).

MEASURING ALGORITHM IMPROVEMENT

An algorithm family's overall performance improves when new algorithms are discovered that can solve the same problem, but with fewer operations. To measure this progress, we focus on discoveries that improve asymptotic complexity.

We generated a list of algorithms and their groupings by analyzing coursework from 20 leading computer-science university programs. This yielded 11 algorithm subdomains, for example, "find the shortest path between two nodes on a graph." Second, there has to be an optimal solution. In this example, the solution is that the shortest path has been identified. We then calculate historical improvements by examining the initial algorithm in each family as well as all substantial substantial algorithms that improve time complexity.

CONCLUSIONS

Our analysis finds a wide range of algorithmic improvement from 1978 to 2017 (*Figure 2*). At one extreme, roughly half of all algorithm families experience little or no improvement. At the other, 14% experience improvements so transformative, they radically change how and where the algorithms can be used. This was surprising, because it expands how computers can be used.

Overall, algorithmic progress for the median algorithm family increased substantially. But it progressed by less than Moore's Law for moderate-sized problems. For example, for moderate-sized problems, only 30% to 43% of algorighmic families had improvements comparable to or greater than those from Moore's Law and other hardware advances *(Figure 3).* For larger problems, however, algorithm improvement became the dominant source of progress, outpacing Moore's Law.

The results quantify two important lessons regarding the

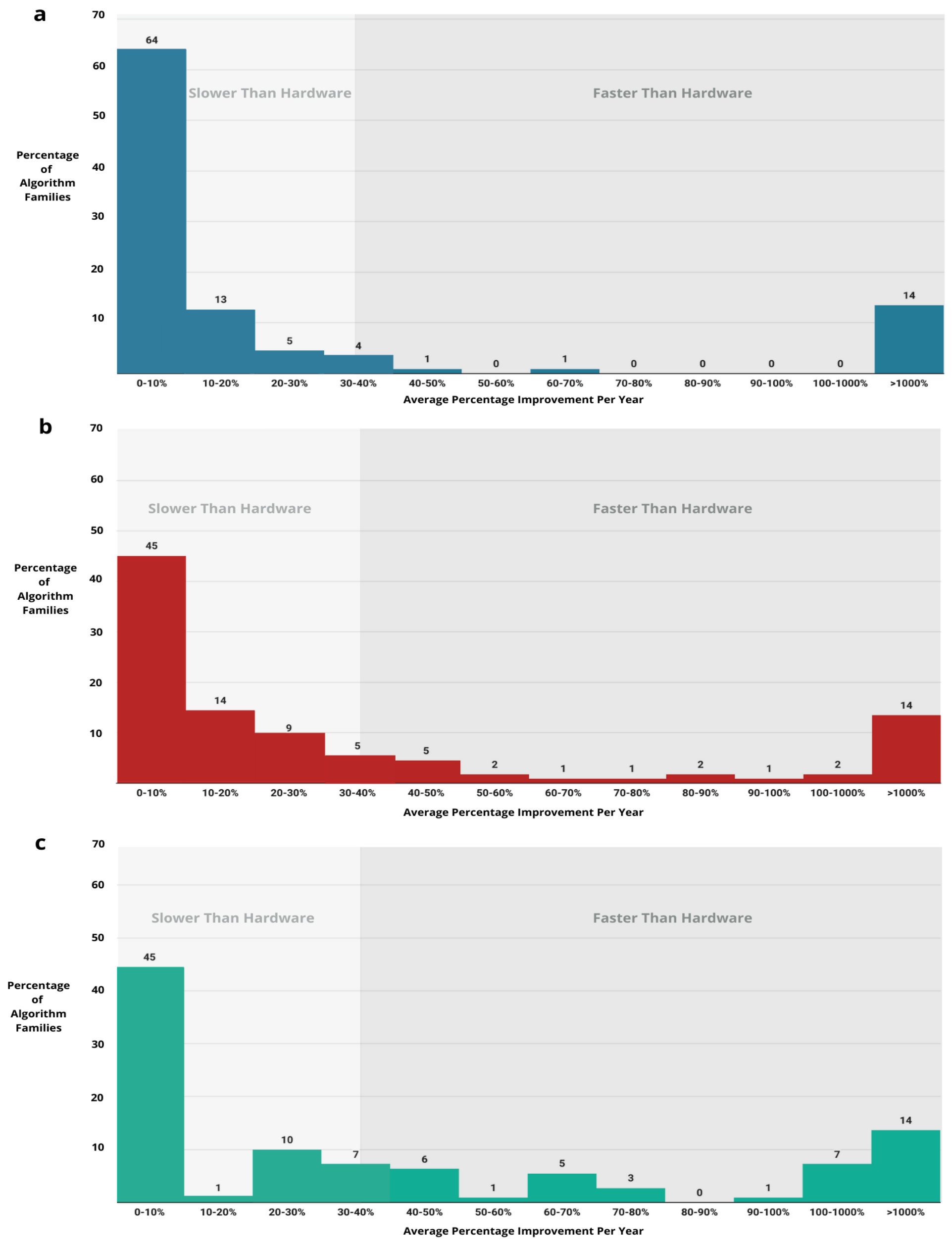

Fig. 3. Average yearly improvement rates for 110 algorithm families as calculated by asymptotic time complexity. Figure (a, blue) illustrates problems with 1,000 data points; figure (b, red), problems with 1 million data points; and figure (c, green), problems with 1 billion data points. The hardware-improvement line shows the average yearly growth rate in SPECint benchmark performance, which specifies CPU integer processing power, from 1978 to 2014(Hennessey and Patterson, 2019).

ways algorithm improvements affect computer science:

- When an algorithm family transitions from exponential to polynomial complexity, it transforms the manageability of that problem in a way no amount of hardware improvement can match.
- As problems increase to billions or even trillions of data points, algorithmic improvement becomes substantially more important than hardware improvements, or Moore's Law, in terms of the average yearly improvement rate. This is especially important for areas with large datasets, such as big data analytics and machine learning.

Taken together, our results highlight the fact that algorithms are one of the most important sources of improvements in computing — a systematic achievement that had previously been mostly undocumented.

REPORT

Read the full research paper **here**

ACKNOWLEDGEMENTS

The authors acknowledge generous funding from the Tides Foundation and the MIT Initiative on the Digital Economy. They also thank Charles Leiserson, the MIT Supertech Group, and Julian Shun for valuable input.

ABOUT THE AUTHORS

Yash Sherry is a researcher at the MIT Computer Science & Artificial Intelligence Laboratory.

Neil C. Thompson is a Research Scientist in MIT's Computer Science & Artificial Intelligence Laboratory. He's also a member of the research team at MIT's Initiative on the Digital Economy.

REFERENCES

Bixby, R.E. (2002). "Solving real-world linear programs: A decade and more of progress," Operations Research, vol. 50, no. 1, 3-15. doi: 10.1287/opre.50.1.3. 17780.

Grace, K. (2013). "Algorithm progress in six domains," Machine Intelligence Research Institute, Berkeley, Calif.; Tech. Rep.

Hennessey, J.L., and Patterson, D.A. (2019). Computer Architecture: A Quantitative Approach, 5th ed., San Mateo, Calif., Morgan Kaufmann.

Hernandez, D., and Brown, T. (2020). "A.I. and efficiency," OpenAI, San Francisco, Calif., Tech. Rep., abs/2005.04305 Arxiv.

Leiserson, C. E., et al. (2020). "There's plenty of room at the top: What will drive computer performance after Moore's law?" Science, vol. 368, no. 6495, Art. no. eaam9744.

Thompson, N., Ge, S., and Filipe, G. (2020). "The importance of (exponentially more) computing," Tech. Rep.

Thompson, N., Greenewald, K., and Lee, K. (2020). "The computation limits of deep learning," arXiv:2007.05558 [online]. Available: https://arxiv.org/abs/2007.05558.

Womble, D.E. (2004). "Is there a Moore's law for algorithms," Sandia National Laboratories, Albuquerque, N.M.; Tech. Rep.

SUBSCRIBE TO OUR NEWSLETTER

MIT
INITIATIVE ON THE
DIGITAL ECONOMY

MIT Initiative on the Digital Economy
MIT Sloan School of Management
245 First St, Room E94-1521
Cambridge, MA 02142-1347
617-452-3216

Our Mission: The MIT IDE is solely focused on the digital economy. We conduct groundbreaking research, convene the brightest minds, promote dialogue, expand knowledge and awareness, and implement solutions that provide critical, actionable insight for people, businesses, and government. We are solving the most pressing issues of the second machine age, such as defining the future of work in this time of unprecendented disruptive digital transformation.

Contact Us: David Verrill, Executive Director
MIT Initiative on the Digital Economy
MIT Sloan School of Management
245 First St, Room E94-1521
Cambridge, MA 02142-1347
617-452-3216
dverrill@mit.edu

Become a Sponsor: The generous support of individuals, foundations, and corporations help to fuel cutting-edge research by MIT faculty and graduate students, and enables new faculty hiring, curriculum development, events, and fellowships. **View all of our Sponsors**