Researchers develop speedier network analysis for a range of computer hardware
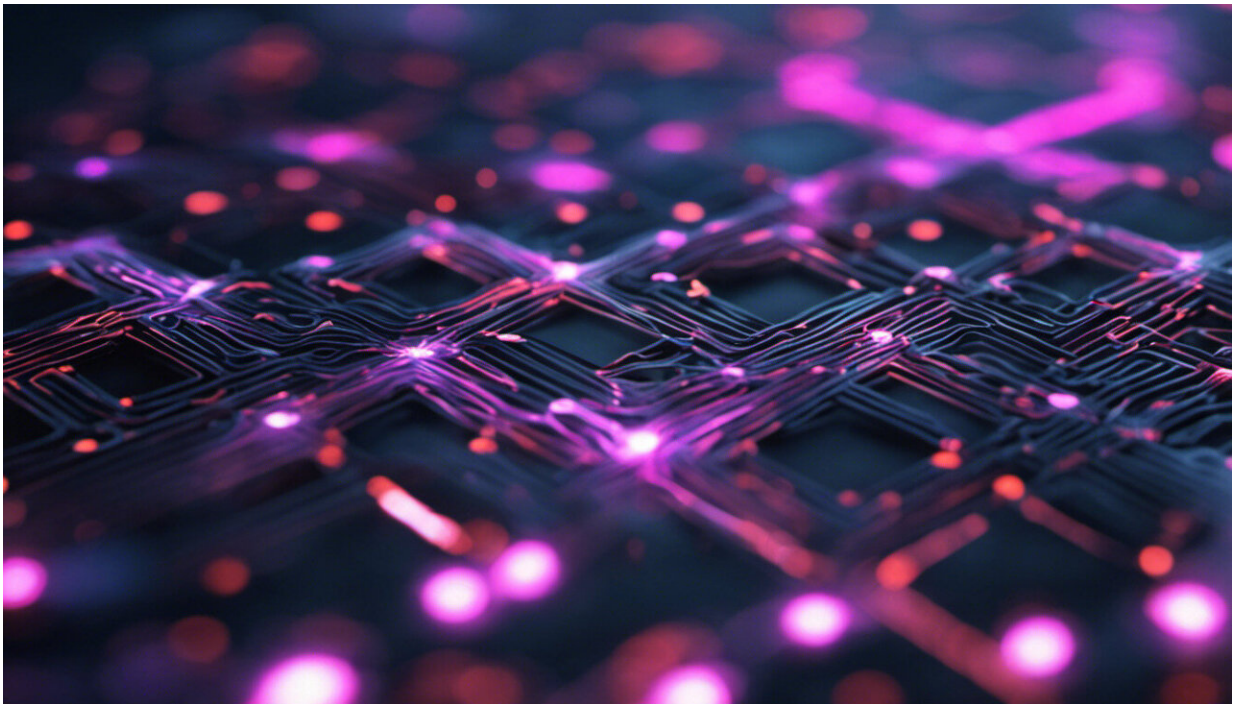
February 22 2021, by Daniel Ackerman

Credit: AI-generated image (disclaimer)

Graphs—data structures that show the relationship among objects—are highly versatile. It's easy to imagine a graph depicting a social media network's web of connections. But graphs are also used in programs as diverse as content recommendation (what to watch next on Netflix?) and navigation (what's the quickest route to the beach?). As Ajay

Brahmakshatriya summarizes, "Graphs are basically everywhere."

Brahmakshatriya has developed software to more efficiently run graph applications on a wider range of computer hardware. The software extends GraphIt, a state-of-the-art graph programming language, to run on graphics processing units (GPUs), hardware that processes many data streams in parallel. The advance could accelerate graph analysis, especially for applications that benefit from a GPU's parallelism, such as recommendation algorithms.

Brahmakshatriya, a Ph.D. student in MIT's Department of Electrical Engineering and Computer Science and the Computer Science and Artificial Intelligence Laboratory, will present the work at this month's International Symposium on Code Generation and Optimization. Co-authors include Brahmakshatriya's advisor, Professor Saman Amarasinghe, as well as Douglas T. Ross Career Development Assistant Professor of Software Technology Julian Shun, postdoc Changwan Hong, recent MIT Ph.D. student Yunming Zhang Ph.D. '20 (now with Google), and Adobe Research's Shoaib Kamil.

When programmers write code, they don't talk directly to the computer hardware. The hardware itself operates in binary—1s and 0s—while the coder writes in a structured, "high-level" language made up of words and symbols. Translating that [high-level language](#) into hardware-readable binary requires programs called compilers. "A compiler converts the code to a format that can run on the hardware," says Brahmakshatriya. One such compiler, specially designed for graph analysis, is GraphIt.

The researchers developed GraphIt in 2018 to optimize the performance of graph-based algorithms regardless of the size and shape of the graph. GraphIt allows the user not only to input an algorithm, but also to schedule how that algorithm runs on the hardware. "The user can provide different options for the scheduling, until they figure out what works

best for them," says Brahmakshatriya. "GraphIt generates very specialized code tailored for each application to run as efficiently as possible."

A number of startups and established tech firms alike have adopted GraphIt to aid their development of graph applications. But Brahmakshatriya says the first iteration of GraphIt had a shortcoming: It only runs on central processing units or CPUs, the type of processor in a typical laptop.

"Some algorithms are massively parallel," says Brahmakshatriya, "meaning they can better utilize hardware like a GPU that has 10,000 cores for execution." He notes that some types of graph analysis, including recommendation algorithms, require a high degree of parallelism. So Brahmakshatriya extended GraphIt to enable graph analysis to flourish on GPUs.

Brahmakshatriya's team preserved the way GraphIt users input algorithms, but adapted the scheduling component for a wider array of hardware. "Our main design decision in extending GraphIt to GPUs was to keep the algorithm representation exactly the same," says Brahmakshatriya. "Instead, we added a new scheduling language. So, the user can keep the same algorithms that they had before written before [for CPUs], and just change the scheduling input to get the GPU code."

This new, optimized scheduling for GPUs gives a boost to graph algorithms that require high parallelism—including recommendation algorithms or internet search functions that sift through millions of websites simultaneously. To confirm the efficacy of GraphIt's new extension, the team ran 90 experiments pitting GraphIt's runtime against other state-of-the-art graph compilers on GPUs. The experiments included a range of algorithms and graph types, from road networks to social networks. GraphIt ran fastest in 65 of the 90 cases and was close

behind the leading algorithm in the rest of the trials, demonstrating both its speed and versatility.

GraphIt "advances the field by attaining performance and productivity simultaneously," says Adrian Sampson, a computer scientist at Cornell University who was not involved with the research. "Traditional ways of doing graph analysis have one or the other: Either you can write a simple algorithm with mediocre performance, or you can hire an expert to write an extremely fast implementation—but that kind of performance is rarely accessible to mere mortals. The GraphIt extension is the key to letting ordinary people write high-level, abstract algorithms and nonetheless getting expert-level performance out of GPUs."

Sampson adds the advance could be particularly useful in rapidly changing fields: "An exciting domain like that is genomics, where algorithms are evolving so quickly that high-performance expert implementations can't keep up with the rate of change. I'm excited for bioinformatics practitioners to get their hands on GraphIt to expand the kinds of genomic analyses they're capable of."

Brahmakshatriya says the new GraphIt extension provides a meaningful advance in graph analysis, enabling users to go between CPUs and GPUs with state-of-the-art performance with ease. "The field these days is tooth-and-nail competition. There are new frameworks coming out every day," He says. But he emphasizes that the payoff for even slight optimization is worth it. "Companies are spending millions of dollars each day to run graph algorithms. Even if you make it run just 5 percent faster, you're saving many thousands of dollars."

  More information: Compiling Graph Applications for GPUs with GraphIt. intimeand.space/docs/gpu-graphit.pdf

This story is republished courtesy of MIT News (web.mit.edu/newsoffice/), a popular site that covers news about MIT research, innovation and teaching.

Provided by Massachusetts Institute of Technology