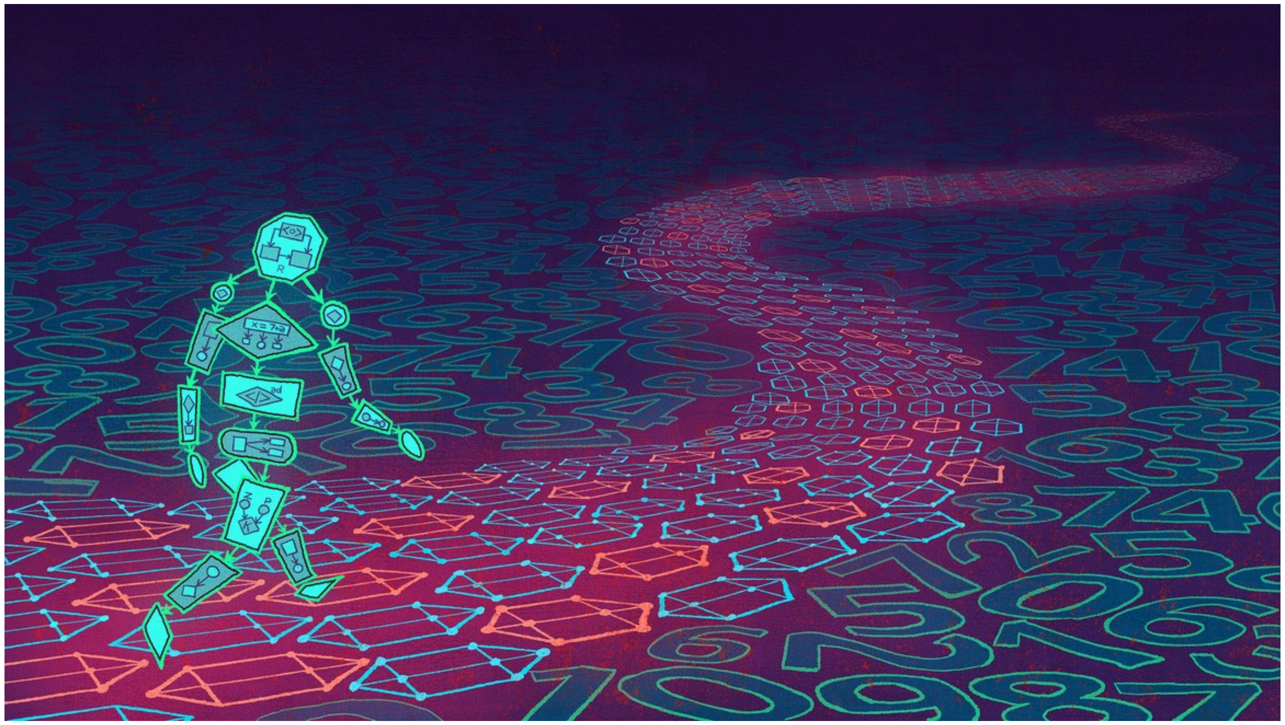
KEVIN HARTNETT SCIENCE 10.29.17 07:00 AM

BEST-EVER ALGORITHM FOUND FOR HUGE STREAMS OF DATA


ANTOINE DORÉ/ QUANTA MAGAZINE

to declare a brief pause, so you can figure out what's trending. That's not feasible, though, so instead you need to find a way to tally hashtags on the fly.

Computer programs that perform these kinds of on-the-go calculations are called streaming algorithms. Because data comes at them continuously, and in such volume, they try to record the essence of what they've seen while strategically forgetting the rest. For more than 30 years computer scientists have worked to build a better streaming algorithm. Last fall a team of researchers invented one that is just about perfect.

"We developed a new algorithm that is simultaneously the best" on every performance dimension, said Jelani Nelson, a computer scientist at Harvard University and a co-author of the work with Kasper Green Larsen of Aarhus University in Denmark, Huy Nguyen of Northeastern University and Mikkel Thorup of the University of Copenhagen.

This best-in-class streaming algorithm works by remembering just enough of what it's seen to tell you what it's seen most frequently. It suggests that compromises that seemed intrinsic to the analysis of

Trend Spotting

Streaming algorithms are helpful in any situation where you're monitoring a database that's being updated continuously. This could be AT&T keeping tabs on data packets or Google charting the never-ending flow of search queries. In these situations it's useful, even necessary, to have a method for answering real-time questions about the data without re-examining or even remembering every piece of data you've ever seen.

Here's a simple example. Imagine you have a continuous stream of numbers and you want to know the sum of all the numbers you've seen so far. In this case it's obvious that instead of remembering every number, you can get by with remembering just one: the running sum.

The challenge gets harder, though, when the questions you want to ask about your data get more complicated. Imagine that instead of calculating the sum, you want to be able to answer the following question: Which numbers have appeared most frequently? It's less obvious what kind of shortcut you could use to keep an answer at the ready.

This particular puzzle is known as the "frequent items" or "heavy hitters" problem. The first algorithm to solve it was developed in the early 1980s by David Gries of Cornell University and Jayadev Misra of the University of Texas, Austin. Their program was effective in a number of ways, but it couldn't handle what's called major event such as Hurricane Irma.

“It’s a coding problem—you’re encoding information down to compact summary and trying to extract information that lets you recover what was put in initially,” said Graham Cormode, a computer scientist at the University of Warwick.

Jelani Nelson, a theoretical computer scientist at Harvard University, co-developed the new algorithm. YAPHET TEKLU

Over the next 30-plus years, Cormode and other computer scientists improved Gries and Misra’s algorithm. Some of the new algorithms were able to detect trending terms, for example, while others were able to work with a more fine-grained definition of what it means for a term to be frequent. All those algorithms made trade-offs, like sacrificing speed for accuracy or memory consumption for reliability.

Most of these efforts relied on an index. Imagine, for example, you are trying to identify frequent search terms. One way to do it would be to assign a number to every word in the English language and then pair that number with a second number that keeps track of how many times that word has been searched. Maybe “aardvark” gets indexed as word number 17 and appears in your database as (17, 9), moment, you know exactly how many times each word has been searched.

Still, it has drawbacks—namely that it takes a lot of time for an algorithm to comb through the hundreds of thousands of words in the English language.

But what if there were only 100 words in the dictionary? Then “looping over every word in the dictionary wouldn’t take that long,” Nelson said.

Alas, the number of words in the dictionary is what it is. Unless, as the authors of the new algorithm discovered, you can break the big dictionary into smaller dictionaries and find a clever way to put it back together.

Small Data

Small numbers are easier to keep track of than big numbers.

Imagine, for example, that you’re monitoring a stream of numbers between zero and 50,000,000 (a task similar to logging internet users by their IP addresses). You could keep track of the numbers using a 50,000,000-term index, but it’s hard to work with an index that size. A better way is to think of each eight-digit number as four two-digit numbers linked together.

Say you see the number 12,345,678. One memory-efficient way to remember it is to break it into four two-digit blocks: 12, 34, 56, 78. Then you can send each block to a sub-algorithm that calculates item frequencies: 12 goes to copy one of the algorithm, 34 goes to copy two, 56 goes to copy three, and 78 goes to copy four.

version never sees anything bigger than a two-digit number, each index only runs from 0 to 99.

An important feature of this splitting is that if the big number—12,345,678—appears frequently in your overall data stream, so will its two-digit components. When you ask each sub-algorithm to identify the numbers it has seen the most, copy one will spit out 12, copy two will spit out 34, and so on. You'll be able to find the most frequent members of a huge list just by looking for the frequent items in four much shorter lists.

LUCY READING-IKKANDA/QUANTA MAGAZINE

“Instead of spending 50 million units of time looping over the entire universe, you only have four algorithms spending 100 units of time,” Nelson said.

The main problem with this divide-and-conquer strategy is that while it's easy to split a big number into small numbers, the reverse is trickier—it's hard to fish out the right small numbers to recombine to give you the right big number.

Imagine, for example, that your data stream frequently includes two numbers that have some digits in common: 12,345,678 and 12,999,999. Both start with 12. Your algorithm splits each number into four smaller numbers, then sends each to a sub-algorithm. Later, you ask each sub-algorithm, “Which numbers have you seen most case, to two different numbers.

"The challenge is to figure out which two-digit blocks to concatenate with which other two-digit blocks," Nelson said.

MORE QUANTA

KEVIN HARTNETT
Computer Scientists Close In on Perfect, Hack-Proof Code

JOHN RENNIE
Ecologists Are Drowning in Sea of Data. These Tools Could Help

ERICA KLARREICH
Landmark Algorithm Breaks 30-Year Impasse

The authors of the new work solve this dilemma by packaging each two-digit block with a little tag that doesn't take up much memory but still allows the algorithm to put the two-digit pieces back together in the right way.

To see one simple approach to how the tagging might work, start with 12,345,678 and split it into two-digit blocks. But this time, before you send each block to its respective sub-algorithm, package the block with a pair of unique identifying numbers that can be used to put the blocks back together. The first of these tags serves as the block's name, the second as a link. In this way, 12,345,678 becomes:

12, 0, 1 / 34, 1, 2 / 56, 2, 3 / 78, 3, 4

Here the number 12 has the name "0" and gets linked to the number named "1." The frequently, 12 goes looking for a number tagged with "1" and finds 34, then 34 goes looking for a number tagged with "2" and finds 56, and 56 goes looking for a number tagged with "3" and finds 78.

In this way, you can think of the two-digit blocks as links in a chain, with the links held together by these extra tagging numbers.

The problem with chains, of course, is that they're only as strong as their weakest link. And these chains are almost guaranteed to break.

Building Blocks

No algorithm works perfectly every time you run it—even the best ones misfire some small percentage of the time. In the example we've been using, a misfire could mean that the second two-digit block, 34, gets assigned an incorrect tag, and as a result, when it goes looking for the block it's supposed to be joined to, it doesn't have the information it needs to find 56. And once one link in the chain fails, the entire effort falls apart.

Mikkel Thorup, a computer scientist at the University of Copenhagen, helped develop an error-resistant way of remembering data.

UNIAVISEN.DK

To avoid this problem, the researchers use what's called an "expander graph." In an expander graph, each two-digit block forms a point. Points get connected by lines (according to the tagging process described above) to form a cluster. The important example, with 12,345,678, you connect 12 with 34 but also with 56, so that you can still tell that 12 and 56 belong in the same number even if the link between 12 and 34 fails.

An expander graph doesn't always come out perfectly. Sometimes it'll fail to link two blocks that should be linked. Or it'll link two blocks that don't belong together. To counteract this tendency, the researchers developed the final step of their algorithm: a "cluster-preserving" sub-algorithm that can survey an expander graph and accurately determine which points are meant to be clustered together and which aren't, even when some lines are missing and false ones have been added.

"This guarantees I can recover something that looks like the original clusters," Thorup said.

And while Twitter isn't going to plug in the expander sketch tomorrow, the techniques underlying it are applicable to a far wider range of computer science problems than tallying tweets. The algorithm also proves that certain sacrifices that previously seemed necessary to answer the frequent-items problem don't need to be made. Previous algorithms always gave up something — they were accurate but memory-intensive, or fast but unable to determine which frequent items were trending. This new work shows that given the right way of encoding a lot of information, you can end up with the best of all possible worlds: You can store your frequent items and recall them, too.

mathematics and the physical and life sciences.

RELATED VIDEO

SECURITY

Meet Big Data

Agent Tepple reveals a few tricks of the pre-digital trade when Winters attempts to explain to him how computers

MORE SCIENCE

Is the Universe a Hologram? Maybe! This Math Trick Shows How

NATALIE WOLCHOVER

BRIGHTEN UP

Sunscreen Regulations Haven’t Aged Well

MEGAN MOLTENI

DAYTONA 500

Estimate the Friction in That Massive Nascar Pile-Up

RHETT ALLAIN

WORLD RECORD

One Woman Pushes Hula-Hooping to Its Absurd, Glittery Limits

ROBBIE GONZALEZ

THE UNDEAD

The Space Shuttle Rises From the Dead to Power New Vehicles

MARK HARRIS

GET SCIENCE NEWSLETTER

Sign up to receive the latest science news.

Enter your email

SUBMIT

FOLLOW US ON TWITTER

Visit WIRED Photo for our unfiltered take on photography, photographers, and photographic journalism wired.com/category/photo

FOLLOW