System that automatically handles database caching in server farms increases speed and reliability

January 23 2017, by Larry Hardesty

Researchers from MIT's Computer Science and Artificial Intelligence Laboratory have designed a new system that automatically handles caching of database queries for web applications written in the web-programming language Ur/Web. Credit: Jose-Luis Olivares/MIT

Today, loading a web page on a big website usually involves a database query—to retrieve the latest contributions to a discussion you're participating in, a list of news stories related to the one you're reading, links targeted to your geographic location, or the like.

But database queries are time consuming, so many websites store—or "cache"—the results of common queries on web servers for faster delivery.

If a site user changes a value in the database, however, the cache needs to be updated, too. The complex task of analyzing a website's code to identify which operations necessitate updates to which cached values generally falls to the web programmer. Missing one such operation can result in an unusable site.

This week, at the Association for Computing Machinery's Symposium on Principles of Programming Languages, researchers from MIT's Computer Science and Artificial Intelligence Laboratory presented a new system that automatically handles caching of database queries for web applications written in the web-programming language Ur/Web.

Although a website may be fielding many requests in parallel—sending different users different cached data, or even data cached on different servers—the system guarantees that, to the user, every transaction will look exactly as it would if requests were handled in sequence. So a user won't, for instance, click on a link showing that tickets to an event are available, only to find that they've been snatched up when it comes time to pay.

In experiments involving two websites that had been built using Ur/Web, the new system's automatic caching offered twofold and 30-fold speedups.

"Most very popular websites backed by databases don't actually ask the database over and over again for each request," says Adam Chlipala, an associate professor of electrical engineering and computer science at MIT and senior author on the conference paper. "They notice that, 'Oh, I seem to have asked this question quite recently, and I saved the result, so I'll just pull that out of memory.'"

"But the tricky part here is that you have to realize when you make changes to the database that some of your saved answers are no longer necessarily correct, and you have to do what's called 'invalidating' them. And in the mainstream way of implementing this, the programmer needs to manually add invalidation logic. For every line of code that changes the database, the programmer has to sit down and think, 'Okay, for every other line of code that reads the database and saves the result in a cache, which ones of those are going to be broken by the change I just made?'"

Chlipala is joined on the paper by Ziv Scully, a graduate student in computer science at Carnegie Mellon University, who worked in Chlipala's lab as an MIT undergraduate.

Exhaustive search

Ur/Web, which Chlipala invented, lets web developers completely specify their sites' functionality using just one programming language. The Ur/Web compiler then automatically generates all the different types of code required to power a website—HTML, JavaScript, SQL database queries, and cascading style sheets—while providing certain performance and security guarantees. Chlipala and Scully's new system is a modification of the compiler, so Ur/Web users can simply recompile their existing code to get all the benefits of database caching. The language itself remains unchanged.

The new compiler begins by analyzing the Ur/Web code and determining

what data to cache and how to organize it. For instance, if certain types of queries are almost always performed in conjunction with each other, Ur/Web will cache the associated data in a single table. The compiler also decides whether to cache raw data, HTML code, or, if the program structure permits it, entire webpages.

Then the compiler goes through the code and compares every operation that updates a value in the database with every operation that queries the database, to determine which cached values need to be invalidated when, and it adds the appropriate cache-invalidation commands in the appropriate places.

Like many programming languages, Ur/Web has an associated "runtime," a small program that runs in the background to manage the execution of applications written in the language. Chlipala and Scully also updated the Ur/Web runtime to monitor the frequency with which cached database queries are reused. Any that prove superfluous are deleted, which improves the system's efficiency.

Painless improvements

In addition to testing the system on two full websites, Chlipala and Scully also tested it on a handful of smaller programs, also written in Ur/Web, which are part of a standard set of benchmarks used to compare different Web development frameworks. On those, the new system offered speedups of between twofold and fivefold.

They did not, however, compare sites developed using their system to sites that use hand-coded caching because, as Chlipala explains, "Ur/Web's so much faster than everything else."

"Even if it turns out that someone could put in the extra work and get a tripling of the throughput, our argument is that it's a pretty good deal to

get a doubling of your throughput with no extra work," he adds.

"Many websites and other programs that use databases use caches to improve performance," says Andrew Myers, a professor of computer science at Cornell University. "However, it's quite tricky to do this correctly, since the results of database queries can become invalid. So the attempt to speed up applications in this way leads to bugs that can be very challenging to find. Rather than requiring the programmer to identify the results that need to be discarded, the [MIT researchers'] system does this automatically, by analyzing the different requests that the application makes to the database."

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

Citation: System that automatically handles database caching in server farms increases speed and reliability (2017, January 23) retrieved 26 April 2024 from https://techxplore.com/news/2017-01-automatically-database-caching-server-farms.html