Putting the squeeze on data

December 21 2009, by Larry Hardesty

(PhysOrg.com) -- Data compression is one of the fundamental research areas in computer science, letting information systems do more with less. It’s the reason the iPod nano can hold thousands of songs instead of hundreds, and it’s what keeps transmitted images from choking the Internet. If every digital file is a string of bits — zeroes and ones — then compression is a way to represent the same information with fewer bits.

Most compression techniques trade space for time: while the compressed file takes up less memory, it has to be decoded before its contents are intelligible. In applications where memory is in short supply but data needs constant updating, it can be prohibitively time consuming to keep decompressing a file, modifying it, and then recompressing it. As a result, such applications — monitoring Internet traffic, for instance, or looking for patterns in huge collections of scientific data — often use a type of compression called linear compression. With linear compression, a computer program can modify the data in a compressed file without first decoding it.

Last year, Associate Professor Piotr Indyk of MIT's Computer Science and Artificial Intelligence Laboratory and his graduate student Radu Berinde introduced two different versions of a new linear-compression algorithm that perform as well as any yet invented — and for some applications, better. Both versions of the algorithm, however, had limitations: under certain extreme conditions, they'd just stop working. But this fall, at the Allerton Conference on Communication, Control, and Computing hosted by the University of Illinois at Urbana-Champaign, Indyk and Berinde presented a new version of the algorithm that combined the advantages of its predecessors and overcame their drawbacks.

“The two previous algorithms each had their own faults,” says Deanna Needell, a postdoc at Stanford who helped develop one of the other leading linear-compression algorithms. “And this tends to be the case in many situations in this area, that if you have two algorithms, one is good at one thing and the other is good at another. And [the new algorithm] sort of merged the two benefits. It's like, here's this algorithm that does both of the good things.”

Some compression techniques, like the zip algorithm commonly used for Internet downloads, are what's called “lossless”: when you unzip the zipped version of a file, you recover every bit of the original. Other compression techniques are “lossy”: the MP3 version of a song, for example, takes up about a tenth as much space as the CD version, but it irreversibly discards a lot of subtle audio data.

Linear compression is lossy: expanding the compressed file doesn't give you all the data in the original. But for many applications, that doesn't matter. Take, for instance, Internet traffic monitoring. Packets of data traveling over the Internet pass through a succession of special-purpose computers called routers; each router examines the packet's ultimate destination and tells it where to go next. There's no way a router could store information about all the packets that pass through it in the course of a day, but with linear compression, it can store an approximation. Decoding the data can still disclose what Indyk calls the "heavy hitters" — the sites that are sending and receiving the most packets — which is what most researchers are interested. In other applications, the heavy hitters might be the members of a large population whose blood tests positive for a disease, or the concentrations of particular molecules in a chemical sample.

Going the distance

According to Indyk, there are three principal criteria for evaluating the performance of a linear-compression algorithm. One is the degree of compression: how much smaller the compressed file is than the uncompressed data. The second is recovery time: how long it takes to decode the compressed data. (Indyk says that some of the early linear-compression algorithms would take "hours or even days" to reconstruct an image captured by a one-megapixel camera.) And since linear compression is lossy, the third is how accurately the algorithm can reconstruct the original file.

In the last seven years, Indyk says, the field has progressed to the point where linear-compression algorithms can perform well along any two of those three parameters at the expense of the third. Indyk and Berinde chose to trade some fidelity in reconstruction for efficient extraction and good compression. Indeed, Indyk and some of his other students have recently demonstrated that there's a mathematical limit to how much space savings linear compression can afford — and his and Berinde's algorithm reaches it.

The insight behind the MIT researchers' algorithm is fairly technical, but Indyk tried to explain it in layman's terms. If you take two very different files — strings of ones and zeroes — of similar size, "the difference between them has a geometric interpretation," Indyk explains. That is, there's a way to mathematically describe the difference between the files in terms of distance: one file can be thought of as being close to or far away from the other.

With linear compression, there's generally a trade-off between how fast the compression algorithm is and how much of the original file can be recovered. Slower but more accurate algorithms tend to preserve the geometric distance between files: if two uncompressed files are far apart, the compressed versions will be, too. With faster algorithms, on the other hand, the compressed files tend to be much closer to each other than the source files were.

Indyk and Berinde found a way to analyze the difference between compressed files using a different mathematical notion of geometric distance; under that analysis, some fast compression algorithms still preserve the distance between files. By taking advantage of this new perspective, the researchers were able to devise a decompression algorithm that recovers much more information from the original file without sacrificing any speed.

Work like Indyk and Berinde's holds out the hope that soon, linear-compression algorithms will no longer need to sacrifice performance along one of the three parameters that Indyk mentioned — compression, time and accuracy. "I think we're actually pretty close to that," says Needell. "We're pretty much to the end: we're almost there." She adds, however, that "there's plenty of other directions that the field will go in." For instance, she says, many of the mathematical techniques that linear-compression algorithms rely on could be adapted to improve the "recommendation engines" on web sites like Netflix or Amazon, which try to predict which books or movies a customer might like on the basis of prior history.

Provided by Massachusetts Institute of Technology

Citation: Putting the squeeze on data (2009, December 21) retrieved 26 April 2024 from https://phys.org/news/2009-12-putting-the-squeeze-on-data.html