AI system optimally allocates workloads across thousands of servers to cut costs, save energy

August 22 2019, by Rob Matheson

A novel system by MIT researchers automatically "learns" how to allocate data-processing operations across thousands of servers.

A novel system developed by MIT researchers automatically "learns" how to schedule data-processing operations across thousands of servers—a task traditionally reserved for imprecise, human-designed algorithms. Doing so could help today's power-hungry data centers run far more efficiently.

Data centers can contain tens of thousands of servers, which constantly run data-processing tasks from developers and users. Cluster scheduling algorithms allocate the incoming tasks across the servers, in real-time, to efficiently utilize all available computing resources and get jobs done fast.

Traditionally, however, humans fine-tune those scheduling algorithms, based on some basic guidelines ("policies") and various tradeoffs. They may, for instance, code the algorithm to get certain jobs done quickly or split resource equally between jobs. But workloads—meaning groups of combined tasks—come in all sizes. Therefore, it's virtually impossible for humans to optimize their scheduling algorithms for specific workloads and, as a result, they often fall short of their true efficiency potential.

The MIT researchers instead offloaded all of the manual coding to machines. In a paper being presented at SIGCOMM, they describe a system that leverages "reinforcement learning" (RL), a trial-and-error machine-learning technique, to tailor scheduling decisions to specific workloads in specific server clusters.

To do so, they built novel RL techniques that could train on complex workloads. In training, the system tries many possible ways to allocate incoming workloads across the servers, eventually finding an optimal tradeoff in utilizing computation resources and quick processing speeds. No human intervention is required beyond a simple instruction, such as, "minimize job-completion times."

Compared to the best handwritten scheduling algorithms, the researchers' system completes jobs about 20 to 30 percent faster, and twice as fast during high-traffic times. Mostly, however, the system learns how to compact workloads efficiently to leave little waste. Results indicate the system could enable data centers to handle the same workload at higher speeds, using fewer resources.

"If you have a way of doing trial and error using machines, they can try different ways of scheduling jobs and automatically figure out which strategy is better than others," says Hongzi Mao, a Ph.D. student in the Department of Electrical Engineering and Computer Science (EECS). "That can improve the system performance automatically. And any slight improvement in utilization, even 1 percent, can save millions of dollars and a lot of energy in data centers."

"There's no one-size-fits-all to making scheduling decisions," adds co-author Mohammad Alizadeh, an EECS professor and researcher in the Computer Science and Artificial Intelligence Laboratory (CSAIL). "In existing systems, these are hard-coded parameters that you have to decide up front. Our system instead learns to tune its schedule policy characteristics, depending on the data center and workload."

Joining Mao and Alizadeh on the paper are: postdocs Malte Schwarzkopf and Shaileshh Bojja Venkatakrishnan, and graduate research assistant Zili Meng, all of CSAIL.

RL for scheduling

Typically, data processing jobs come into data centers represented as graphs of "nodes" and "edges." Each node represents some computation task that needs to be done, where the larger the node, the more computation power needed. The edges connecting the nodes link connected tasks together. Scheduling algorithms assign nodes to servers, based on various policies.

But traditional RL systems are not accustomed to processing such dynamic graphs. These systems use a software "agent" that makes decisions and receives a feedback signal as a reward. Essentially, it tries to maximize its rewards for any given action to learn an ideal behavior in a certain context. They can, for instance, help robots learn to perform a task like picking up an object by interacting with the environment, but that involves processing video or images through an easier set grid of pixels.

To build their RL-based scheduler, called Decima, the researchers had to develop a model that could process graph-structured jobs, and scale to a large number of jobs and servers. Their system's "agent" is a scheduling algorithm that leverages a graph neural network, commonly used to process graph-structured data. To come up with a graph neural network suitable for scheduling, they implemented a custom component that aggregates information across paths in the graph—such as quickly estimating how much computation is needed to complete a given part of the graph. That's important for job scheduling, because "child" (lower) nodes cannot begin executing until their "parent" (upper) nodes finish, so anticipating future work along different paths in the graph is central to making good scheduling decisions.

To train their RL system, the researchers simulated many different graph sequences that mimic workloads coming into data centers. The agent then makes decisions about how to allocate each node along the graph to each server. For each decision, a component computes a reward based on how well it did at a specific task—such as minimizing the average time it took to process a single job. The agent keeps going, improving its decisions, until it gets the highest reward possible.

Baselining workloads

One concern, however, is that some workload sequences are more difficult than others to process, because they have larger tasks or more complicated structures. Those will always take longer to process—and, therefore, the reward signal will always be lower—than simpler ones. But that doesn't necessarily mean the system performed poorly: It could make good time on a challenging workload but still be slower than an easier workload. That variability in difficulty makes it challenging for the model to decide what actions are good or not.

To address that, the researchers adapted a technique called "baselining" in this context. This technique takes averages of scenarios with a large number of variables and uses those averages as a baseline to compare future results. During training, they computed a baseline for every input sequence. Then, they let the scheduler train on each workload sequence multiple times. Next, the system took the average performance across all of the decisions made for the same input workload. That average is the baseline against which the model could then compare its future decisions to determine if its decisions are good or bad. They refer to this new technique as "input-dependent baselining."

That innovation, the researchers say, is applicable to many different computer systems. "This is general way to do reinforcement learning in environments where there's this input process that effects environment, and you want every training event to consider one sample of that input process," he says. "Almost all computer systems deal with environments where things are constantly changing."

Aditya Akella, a professor of computer science at the University of Wisconsin at Madison, whose group has designed several high-performance schedulers, found the MIT system could help further improve their own policies. "Decima can go a step further and find opportunities for [scheduling] optimization that are simply too onerous to realize via manual design/tuning processes," Akella says. "The schedulers we designed achieved significant improvements over techniques used in production in terms of application performance and cluster efficiency, but there was still a gap with the ideal improvements we could possibly achieve. Decima shows that an RL-based approach can discover [policies] that help bridge the gap further. Decima improved on our techniques by a [roughly] 30 percent, which came as a huge surprise."

Right now, their model is trained on simulations that try to recreate incoming online traffic in real-time. Next, the researchers hope to train the model on real-time traffic, which could potentially crash the servers. So, they're currently developing a "safety net" that will stop their system when it's about to cause a crash. "We think of it as training wheels," Alizadeh says. "We want this system to continuously train, but it has certain training wheels that if it goes too far we can ensure it doesn't fall over."

More information: Learning Scheduling Algorithms for Data Processing Clusters. arXiv:1810.01963 [cs.LG] arxiv.org/abs/1810.01963

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: AI system optimally allocates workloads across thousands of servers to cut costs, save energy (2019, August 22) retrieved 10 April 2024 from https://techxplore.com/news/2019-08-ai-optimally-allocates-workloads-thousands.html