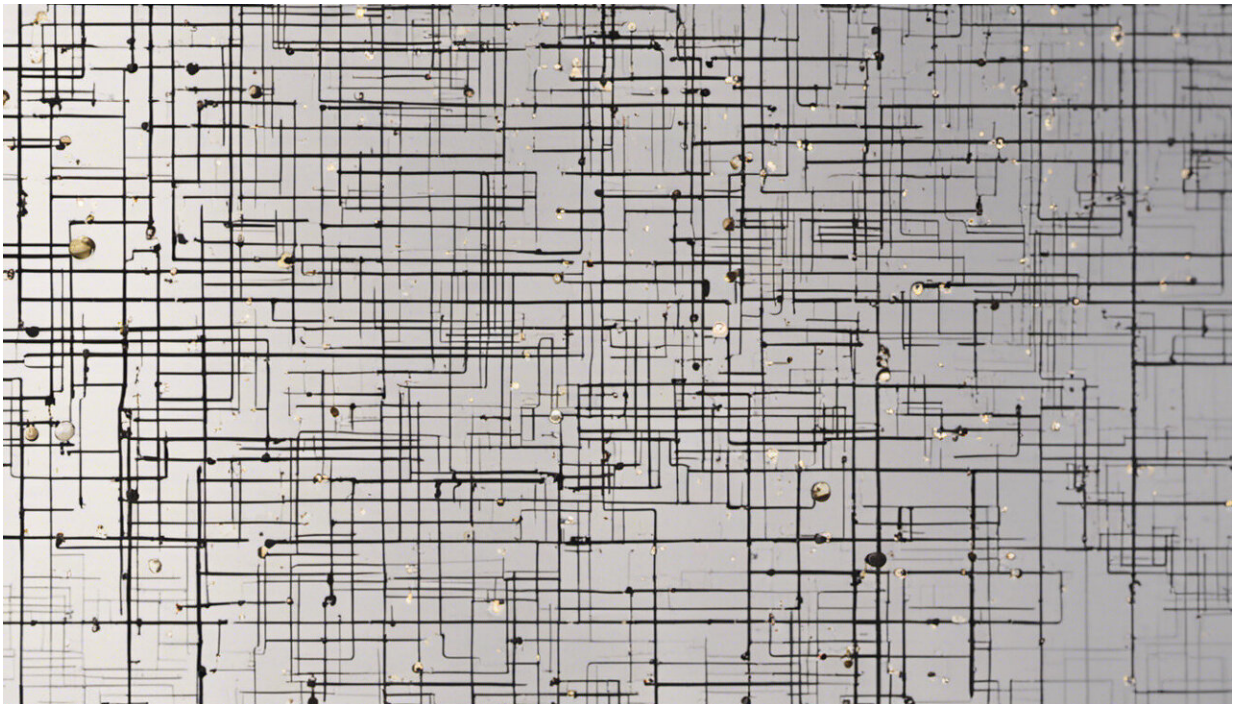# Planning algorithms evaluate probability of success, suggest low-risk alternatives

January 16 2015, by Larry Hardesty

Credit: AI-generated image (disclaimer)

Imagine that you could tell your phone that you want to drive from your house in Boston to a hotel in upstate New York, that you want to stop for lunch at an Applebee's at about 12:30, and that you don't want the trip to take more than four hours. Then imagine that your phone tells you that you have only a 66 percent chance of meeting those criteria—but that if

you can wait until 1:00 for lunch, or if you're willing to eat at TGI Friday's instead, it can get that probability up to 99 percent.

That kind of application is the goal of Brian Williams' group at MIT's Computer Science and Artificial Intelligence Laboratory—although the same underlying framework has led to software that both NASA and the Woods Hole Oceanographic Institution have used to plan missions.

At the annual meeting of the Association for the Advancement of Artificial Intelligence (AAAI) this month, researchers in Williams' group will present algorithms that represent significant steps toward what Williams describes as "a better Siri"—the user-assistance application found in Apple products. But they would be just as useful for any planning task—say, scheduling flights or bus routes.

Together with Williams, Peng Yu and Cheng Fang, who are graduate students in MIT's Department of Aeronautics and Astronautics, have developed software that allows a planner to specify constraints—say, buses along a certain route should reach their destination at 10-minute intervals—and reliability thresholds, such as that the buses should be on time at least 90 percent of the time. Then, on the basis of probabilistic models—which reveal data such as that travel time along this mile of road fluctuates between two and 10 minutes—the system determines whether a solution exists: For example, perhaps the buses' departures should be staggered by six minutes at some times of day, 12 minutes at others.

If, however, a solution doesn't exist, the software doesn't give up. Instead, it suggests ways in which the planner might relax the problem constraints: Could the buses reach their destinations at 12-minute intervals? If the planner rejects the proposed amendment, the software offers an alternative: Could you add a bus to the route?

Short tails

One aspect of the software that distinguishes it from previous planning systems is that it assesses risk. "It's always hard working directly with probabilities, because they always add complexity to your computations," Fang says. "So we added this idea of risk allocation. We say, 'What's your budget of risk for this entire mission? Let's divide that up and use it as a resource.'"

The time it takes to traverse any mile of a bus route, for instance, can be represented by a [probability](#) distribution—a bell curve, plotting time against probability. Keeping track of all those probabilities and compounding them for every mile of the route would yield a huge computation. But if the system knows in advance that the planner can tolerate a certain amount of failure, it can, in effect, assign that failure to the lowest-probability outcomes in the distributions, lopping off their tails. That makes them much easier to deal with mathematically.

At AAAI, Williams and another of his students, Andrew Wang, have a paper describing how to evaluate those assignments efficiently, in order to find quick solutions to soluble planning problems. But the paper with Yu and Fang—which appears at the same conference session—concentrates on identifying those constraints that prevent a problem's solution.

There's the rub

Both procedures are rooted in [graph theory](#). In this context, a graph is a data representation that consists of nodes, usually depicted as circles, and edges, usually depicted as line segments connecting the nodes. Any scheduling problem can be represented as a graph. Nodes represent events, and the edges indicate the sequence in which events must occur.

Each edge also has an associated weight, indicating the cost of progressing from one event to the next—the time it takes a bus to travel between stops, for instance.

Yu, Williams, and Fang's algorithm first represents a problem as a graph, then begins adding edges that represent the constraints imposed by the planner. If the problem is soluble, the weights of the edges representing constraints will everywhere be greater than the weights representing the costs of transitions between events. Existing algorithms, however, can quickly home in on loops in the graph where the weights are imbalanced. The MIT researchers' system then calculates the lowest-cost way of rebalancing the loop, which it presents to the [planner](#) as a modification of the problem's initial constraints.

"These papers are quite interesting," says Jiaying Shen, a research scientist at Nuance Communications, which developed the voice-recognition technology used by Apple's Siri. "They had a flurry of papers on chance constraints, but in the recent papers, they added uncertainty in there, which makes the problems that it can model more complicated and unpredictable and more realistic."

At Nuance, "We are very interested in constraint relaxation, including the chance constraints," Shen adds. "If you expose what you need to consider in the planning stage, then you have a much higher success rate in carrying out the plan."

More information: "Resolving Over-constrained Probabilistic Temporal Problems through Chance Constraint Relaxation": [people.csail.mit.edu/yupeng/fi … /aaai_ccUhura_Yu.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

Citation: Planning algorithms evaluate probability of success, suggest low-risk alternatives (2015, January 16) retrieved 25 April 2024 from https://phys.org/news/2015-01-algorithms-probability-success-low-risk-alternatives.html