# PRODIGY speeds computer learning

Ask someone in New York how to get to the Statue of Liberty and he might start with, "Well, first you take the A-train. . . ." Ask a computer the same thing and the answer might be, "It's going to cost you $2.50, so first you have to get a job. . . ."

Explanations such as the latter – true but terribly time consuming and often unnecessary to follow – can be a major problem for computers using a technique called explanation-based learning (EBL), say computer scientists Steve Minton and Jaime Carbonell of Carnegie Mellon University in Pittsburgh. To avoid this problem, Minton, Carbonell and two colleagues created a program called PRODIGY/EBL to weed out "bad" explanations, using a technique that could simplify some types of problem solving for computers and robots.

The ability to learn from experience and subsequently use what is learned is a critical part of human intelligence. So artificial intelligence researchers have spent a lot of time developing computers that learn to do a task better each time they try. To do this, a computer using EBL will solve a problem and then learn an "explanation" for how to get to the solution so it doesn't have to try as many possible paths the next time around.

Explanation-based learning is a powerful tool that often improves the computer's problem-solving capabilities, but it doesn't *always* improve performance, Minton said last week in St. Paul, Minn., during a presentation to the Seventh National Conference on Artificial Intelligence. The problem is that the explanation the computer learns may be circuitous and contain irrelevant or redundant information. Such explanations are often fine for testing the system, but can lead to extremely long computing times when applied to real-world problems, Minton says.

PRODIGY/EBL combats this by simplifying and consolidating information. "It's the first EBL program to look for the best explanation rather than looking for any explanation," Minton says.

To do this, the computer first decides what elements should be in a good explanation. For example: A scientist has programmed a robot with enough information to get to the Statue of Liberty and instructs it to go there. Along the way, the robot comes out of a subway station and makes a wrong turn, then has to backtrack, and finally reaches its destination.

The robot's directional program has caused it to err on the journey, so it needs to learn how to get to the Statue of Liberty more directly. It will do this by finding an explanation that will help guide it. But it wouldn't be efficient to learn a catalog of its own actions all along the route when all that's needed to correct the problem is an explanation – in the form of a rule – that tells it, "If you're leaving the subway station, turn left," Minton says.

PRODIGY/EBL uses rules of thumb to zero in on the pertinent event, ignoring actions irrelevant to the mistake such as the robot's leaving the lab in the morning. Since such things are all done by default with the robot's preexisting knowledge, "it really doesn't help you to learn a long, complex rule" detailing those actions, Minton says.

Then PRODIGY/EBL eliminates redundancies in the explanation. Commands like "Turn left and don't turn right" are quite common in computer explanations and can slow computation significantly, Minton says. "Computers are quite good at details and will try to get a lot of details in there without seeing the big picture," he says.

Lastly, the program evaluates what has been learned and decides which explanations help the most. "If you have learned 10 things, some are going to help you and some are not," Minton says. If the computer has learned a lot of elaborate explanations that are rarely applicable, learning will actually slow computation because the computer will always be checking to see if such rules apply. "Sometimes you can learn too much," he says.

The computer also will recognize that two different explanations say the same thing and consolidate them into one rule. For example, if two explanations were the same except that the robot left through the front door of the lab in one and the back door in the other, PRODIGY/EBL would recognize that the point of exit was unimportant and that these two were really the same rule.

To make the most economical use of what has been learned, PRODIGY/EBL analyzes how often the rule is applied, the time saved by using the rule and the time it "costs" the computer to decide whether it should invoke the rule or not.

One of the best applications for PRODIGY/EBL is in scheduling tasks, Minton says. Scheduling is a good problem for artificial intelligence programs because they often must reconcile such a large number of competing demands that normal computers may find quite difficult, he adds.

Minton will soon move to NASA Ames Research Center in Mountain View, Calif. He expects to use PRODIGY/EBL to help schedule observing time for the many astronomers who want to use the Hubble Space Telescope, scheduled for launch into Earth-orbit in 1990. The program also may be useful for scheduling the machining of parts in a factory so that everything is at the right place at the right time. PRODIGY/EBL has been used only for simpler scheduling problems so far, but Minton thinks it will work on larger problems too. – *C. Vaughan*

# New protein piece for AIDS puzzle

Experiments by two groups working independently have revealed a previously undetected gene and its protein product in the AIDS-causing virus HIV-1. The list of identified HIV-1 proteins now tallies nine. Because the protein's function remains unknown, the scientists say their results do not point to a way to eliminate AIDS. More likely, they say, the protein can help in AIDS diagnosis.

Both groups began by examining a region of the HIV-1 genetic material known to code for several viral proteins. They identified within this region a sequence of nucleotides – the building blocks of genes – that genetic rules indicated should code for a protein. They then added the chemical ingredients necessary for gene expression to the potential gene, *vpu*, and found it does produce a protein.

The scientists next looked for evidence that *vpu* actually produces a protein outside of the test tube. Analyzing blood sera from a total of more than 30 patients, the two teams detected antibodies to the synthesized *vpu* protein in people infected with HIV-1 but not in any of the uninfected individuals. This indicates "the protein is certainly made in AIDS patients," says Eric Cohen of the Dana-Farber Cancer Institute in Boston. However, not all infected patients had antibodies against the *vpu* protein – a finding the researchers cannot yet explain.

The two groups then diverged in their investigations. Cohen, William Haseltine and their co-workers at Dana-Farber went on to examine HIV-2, a virus that differs from HIV-1 in structure but still causes AIDS (SN: 3/7/87, p.151). Until now, scientists have known of only one genetic difference between the two viral forms: HIV-2 contains a gene that HIV-1 does not. The new results, reported in the Aug. 11 NATURE, show another difference: HIV-2 lacks the *vpu* gene. Cohen says *vpu* protein has potential to serve as a tool in distinguishing HIV-1 from HIV-2.

Approaching the project from a different angle, Klaus Strebel and his colleagues at the National Institute of Allergy and Infectious Diseases mutated HIV-1 so that it did not make *vpu* protein. In the Sept. 2 SCIENCE, they report that while the mutants produce fewer whole viruses, they kill as many of their host cells as do the nonmutants. Attempting to explain this paradox, Cohen says *vpu* protein's job may be to help construct whole HIV-1 from independent viral components, but without *vpu* protein, the yet-to-be-assembled HIV proteins still may destroy cells. "*Vpu* may be like a conductor in an orchestra," he says. "Without *vpu*, the orchestra can still play, but is uncoordinated." – *M. Hendricks*