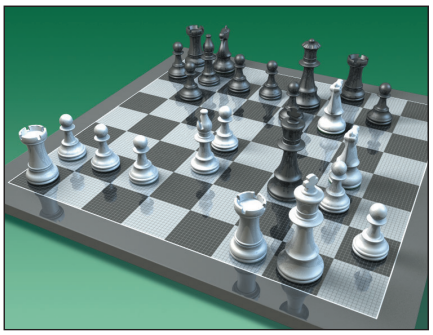
Editor: Dejan Milojičić • *dejan.milojicic@hp.com*

Autograding in the Cloud: Interview with David O'Hallaron



Dejan Milojičić • *Hewlett-Packard Laboratories*

In this installment of Trend Wars, I discuss autograding with David O'Hallaron, a professor of computer science and electrical and computer engineering at Carnegie Mellon University. Featured here is an excerpt from the in-depth interview, which ranged from discussions on autograding to cloud computing. The video is available at www.computer.org/portal/web/computingnow/videos/trendwars.

Dejan Milojičić: It's a pleasure to introduce you to David O'Hallaron, a professor of CS and ECE at Carnegie Mellon University who recently completed a three-year rotation as director of Intel Labs Pittsburgh. I've heard a lot about the autograding system that he built at CMU to automate the evaluation of student programming assignments. I was so excited about this idea that I wanted to share it with this community. So, David, can you please tell us a little bit about autograding?

David O'Hallaron: Randy Bryant and I developed the notion of autograding over the past 10 years in the context of a CMU course we created called Introduction to Computer Systems [ICS]. This is a core course for CS and ECE that aims to help students become better programmers by teaching them about the parts of a computer system that affect the performance, correctness, and utility of their C codes.

A key design goal of the ICS course is that it be very hands-on, that students should learn about systems by experimenting and trying things out on real systems. So, from the very beginning, we centered the course around a set of programming assignments, what we call labs. Because of the size of the course [250 students each term] and the complexity of the labs, we found it essential to develop a Web-based autograding service, called Autolab, to handle the quantitative evaluation of the students' work.

Autograding has had a profound effect on the experience of both students and instructors at CMU. We can see this clearly in Figure 1, which shows the anonymous student ratings for the systems core course that Randy and I were teaching before and after the introduction of autograded labs.

Milojičić: Can you give us an example of an autograded lab?

O'Hallaron: The Data Lab teaches our students about the low-level representations of integers and real numbers. In this lab, students solve a series of programming puzzles. Each puzzle is an initially empty function body that should implement a function such as computing the absolute value, counting the number of ones in a data word, performing a logical shift, or computing the base two log of a number. What makes this challenging is that students have to implement each puzzle with straight-line code, using only a limited set of bitwise operators. The autograder automatically checks the student's work for both correctness by running a set of unit tests and adherence to the programming guidelines by parsing with a specially modified C compiler.

We hand out the autograder with the assignment. Each time students run the autograder, their score is posted on a real-time Web scoreboard [see Figure 2] that shows the performance of the class, rank ordered by score. Students are identified only by the often hilarious nicknames that they give themselves. To get official credit, students hand in to Autolab, which runs the autograder and updates the scoreboard. Students can hand in as often as they like.

Milojičić: What are the three key value propositions of your autograding service, both for students and professors?

O'Hallaron: First, autograding improves the quality of the experience for both the instructors and the students. Labs are no longer limited by the instructor's ability to grade them, but rather by imagination and cleverness in developing autograding software.

Second, autograding expands the potential reach of our labs beyond CMU. We can imagine a hosted Autolab service that provides autograding services on a shared repository of great labs to all of the world's universities.

 1089-7801/11/$26.00 © 2011 IEEE Published by the IEEE Computer Society

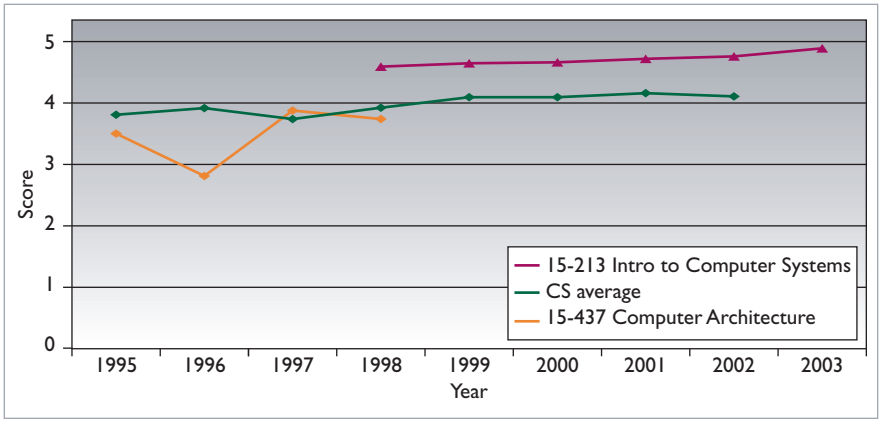


Figure 1. Student course evaluations before and after the introduction of autograded labs in 1998.

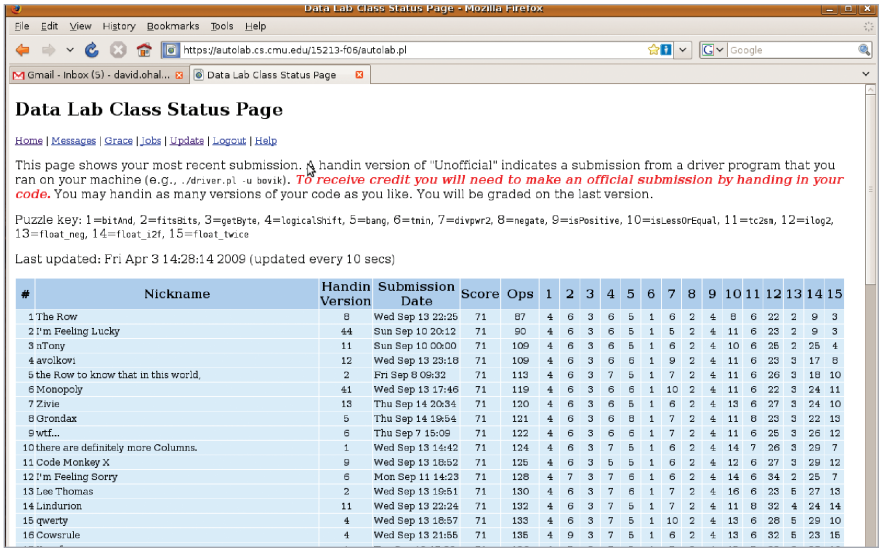


Figure 2. Snapshot of a Data Lab scoreboard.

Third, a hosted autograding service has the potential to foster a new reputation-based community for teachers. Teachers typically work by themselves, and they don't help each other very much. Given a hosted autograding service, instructors with great labs would have a standard way to offer them to other instructors. The service would track adoptions, and instructors with good adoption rates would develop reputations. This is exactly how research works. You develop results, which are used and cited by other researchers. If you do good work, then you earn credit that helps you get promotions, students, grants, and awards. A hosted autograding service would help create a similar kind of community for instructors.

Milojičić: So you told us how it looks from the outside. How is it implemented?

O'Hallaron: The current Autolab system was developed by Hunter Pitelka, David Kosbie, and myself in spring and summer of 2010. It is hosted at CMU and consists of a Linux front-end machine with an 8-Tbyte RAID array and 10 Linux back-end autograding machines. The front end runs a Web server written in Ruby on Rails, a MySQL database, a Tashi cluster manager that manages virtual machines, and a Tango daemon that provides the interface between the Web server and Tashi. The back-end machines run KVM virtual machines on behalf of the front end.

When a student hands in [his] work to the Web server, Tango copies the student's work and the necessary autograding code to a loopback SCSI device. It then directs Tashi to create a copy-on-write instance of a suspended VM and then resumes the instance on one of the back-end machines. When the VM resumes, it mounts the loopback device and runs the autograder, which creates an output file on the loopback device. The VM then unmounts the device and shuts down. Tango extracts the output file from the loopback device, parses the output, and stores the result in the database, where it is available to students via the Web server.

Using the loopback device as the transport mechanism between front and back end is key because it allows us to run untrusted code in network isolated VMs that don't even have [virtual] network cards.

Milojičić: How has technology enabled what you want? For example, would you be able to do that without virtualization?

O'Hallaron: No, it wouldn't be possible. I've been dreaming of a public autograding service for years and could never do it because of the problem of running untrusted code on our servers. Virtual machines solve the problem by providing a secure way to run untrusted code. Virtual machine technology is now sufficiently robust and mainstream that we can use it for autograding.

Milojičić: And how does the cloud play in all of this?

O'Hallaron: One of the big challenges

in providing an autograding service is that every new autograding request must run in a new VM instance. Cloud technology provides us with the capability to spin up these virtual machines, to manage those virtual machines, and to place them on physical machines and run them.

Milojičić: What really made me excited about your work is your promise that you'd be willing to make this available to the IEEE community as a whole.

O'Hallaron: The ultimate goal is to create a centralized hosted service for all the world's universities. This semester, we are deploying a beta Autolab service at Carnegie Mellon for 800 students in two different courses. Later, we will expand the service to other courses at CMU, then create a national service for US schools, and finally an international service. [Readers can try out some of the CMU autograded labs for themselves by visiting http://greatwhite.ics.cs.cmu.edu/autoPublic/.]

Milojičić: When you think about providers of this service, who are really your customers? Academia? How about industry? How about nonprofit organizations?

O'Hallaron: The main customers are academics, but I'm getting interest from industry as well. For example, Intel has a number of ongoing efforts to teach programmers how to write parallel code on multicore processors. A hosted autograding service would be a natural way for companies like Intel to offer self-study courses, and it would also be a nice way to run programming contests.

Milojičić: How easy is it for students to use it, and how easy for professors to write these assignments?

O'Hallaron: It's very easy for students to use. However, developing a good autograded lab requires a month or two of effort, with a couple of iterations to work out the bugs. But once you've got it right, then it's golden.

Milojičić: You've been using it already for a while. What are some key lessons learned from doing this?

O'Hallaron: The biggest lesson we've learned is that the real-time scoreboard is a powerful motivation for students, creating a healthy competition that seems to benefit everyone. On one hand, students at the top of the class finish the assignment pretty quickly, but then notice that someone else has done it better, which drives them to go back and improve their work. On the other hand, students in the bottom and middle of the class have a clear idea of what they need for full credit, can check their work any time, and can see firsthand that good solutions are indeed possible. Compare this to their typical experience where they hand in their work, get graded, and find out a week later what they did wrong.

Milojičić: When you look at your assignments, are there classes of assignments that are not well suited or those that are perfectly well suited? Or is it all up to professor and his creativity — how he makes them?

O'Hallaron: The complexity of writing an autograder increases with the complexity of the assignment. So the approach is more easily adopted for lower-level programming courses that have smaller assignments. However, more complex autograders are indeed possible. For example, CMU faculty have successfully developed autograders for systems such as Unix shells, malloc packages, and C compilers. That said, writing autograders for distributed or highly threaded programs is still quite difficult.

Milojičić: Are there any mistakes that you made in developing the autograder? I personally learn most from my own mistakes. But is there something that you have learned?

O'Hallaron: Oh, we've made a lot of mistakes. For example, we've had several misfires on autograders that involve assignments with concurrency. In general, autograding concurrency and parallelism is a real tough nut.

Milojičić: If you would do it all over again, would you do it the same way? Are there other implementation approaches that are possible?

O'Hallaron: I wrote the first version of Autolab in 2003, and we used it for seven years for the ICS course. We deployed a new version this fall based on all the mistakes I made with the first version. I'm pretty happy with the approach we're using right now: virtual machines on the back end to provide security and isolation, multiple grading servers for scalability, Ruby on Rails on the front end to provide a more sophisticated Web 2.0 look and feel and

About David O'Hallaron

David O'Hallaron is a professor of computer science and electrical and computer engineering at Carnegie Mellon University. His research interests include autograding, virtualization, cloud computing, and high-performance computing. O'Hallaron has a PhD in computer science from the University of Virginia. He is a senior member of IEEE and a member of the ACM. Contact him at droh@cs.cmu.edu.

capability, and a MySQL database to provide full grade books and all the other things that instructors expect.

Milojičić: Until recently, you've been working with me together on the Open Cirrus cloud computing testbed, so I know you're very familiar with that context. How do you think this autograding service fits onto the Open Cirrus model?

O'Hallaron: I think that an autograding service like Autolab is a really interesting vertical for [cloud computing]. It has some unique workload characteristics. Because we're spinning up a VM for every request, it's very dynamic and very interactive, so VM startup latency matters. In addition, the workload is very bursty; students always wait until the last minute to do assignments! So you get very low loads for a while, and as the due date approaches, you get this sort of exponentially increasing load until everything blows up at the end. At scale, it provides a good motivation for elastic acquisition of compute nodes from within the local site, and under extreme loads, from remote sites.

Milojičić: My last question for you is, where do you see your Autolab and autograding service growing?

O'Hallaron: If we're successful, and if this idea pans out, we will offer a hosted Autolab service that provides autograding services to all of the world's universities. We will have labs in the Autolab repository contributed by instructors around the globe for courses spanning low-level CS1 and CS2 courses to higher-level systems courses. And, because of this, we will have sparked the beginnings of a reputation-based community for teachers that could have a big impact on the teaching profession as a whole.

Milojičić: I'm looking forward to seeing this service available from Computing Now and to the whole community. So CS members, please make sure to come and check it out, and see whether you can beat NSD or Kara Serene and post a new record. Thanks again.

For much more on autograding, the entire video is available for viewing online at www.computer.org/portal/web/computingnow/videos/trendwars.

Selected CS articles and columns are also available for free at http://ComputingNow.computer.org.