EPISODE 609

[INTRODUCTION]

[0:00:00.3] JM: Evolutionary algorithms can generate surprising effective solutions to our problems. Evolutionary algorithms are often let loose within a simulated environment. The algorithm is given a function to optimize for and the engineers expect that algorithm to evolve a solution that optimizes for the objective function given the constraints of the simulated environment, but sometimes these results from the evolutionary algorithm are not exactly what we're looking for.

For example, imagine an evolutionary algorithm that tries to evolve a creature that can do a front flip within a simulated physics engine that mirrors the real world. You could imagine all sorts of evolutionary traits. Maybe the creature will evolve to have legs that are like springs and let the creature jump high enough to do a flip. Maybe the creature will develop normal legs but have very strong muscles to propel the creature high enough to do a flip.

What you wouldn't expect is that the creature could just evolve to be extremely tall, because a tall creature can merely lean over fast enough so that the body of its top portion flips upside down, and there was one experiment where the computer scientist who built this simulation and had the creature evolve within it, that's exactly what they witnessed. This creature just grew very tall and flipped over after falling down.

In another similar experiment, there was an evolving creature that discovered a bug in the physics engine of the simulated environment. So this creature was able to exploit the problem with the physics engine to be able to move in ways that could not be possible in our real-world physical universe. It's not exactly what you're looking for when you're trying to make a simulated evolutionary environment that is assuming a normal physical environment.

Evolutionary algorithm sometimes evolve solutions in ways that we do not expect. Researches usually throw these results away because they don't contribute to the result that the researchers are looking for. The consequences that lots of interesting anecdotes get lost and perhaps we

miss out on lessons that are important to understand as artificial intelligence gets more widely used.

Joel Lehman, Dusan Misevic and Jeff Clune are the lead authors of the paper; The Surprising Creativity of Digital Evolution. The paper was a collection of anecdotes about strange results within the world of digital evolution. Joel, and Dusan, and Jeff join the show to describe what digital evolution is and some of the strange results that they surveyed in their paper. Joel and Jeff are engineers are Uber's artificial intelligence division. So this topic has applicable importance to them. Machine learning is all about evolution within simulated environments and developing safe algorithms for AI requires an understanding of what can go wrong in a poorly-defined evolutionary system.

To find all of our episodes about machine learning, you can download the Software Engineering Daily app for iOS or android. It has all 700+ of our episodes and you can download them, you can find related links. You can find discussions. You can get personalized recommendations about episodes of Software Engineering Daily because there are a lot of episodes and it can be hard to find the best ones to listen to.

If you want to find those apps, they are in the iOS and android app stores. You can also always send me an email if you have any feedback or suggestions about the show, jeff@softwareengineeringdaily.com.

[SPONSOR MESSAGE]

[0:03:46.4] JM: You listen to this podcast to raise your skills. You're getting exposure to new technologies and becoming a better engineer because of it. Your job should reward you for being a constant learner, and Hired helps you find your dream job. Hired makes finding a new job easy. On Hired, companies request interviews from software engineers with upfront offers of salary and equity so that you don't waste your time with a company that is not going to value your time.

Hired makes finding a job efficient, and they work with more than 6,000 companies, from startups to large public companies. Go to hired.com/sedaily and get $600 free if you find a job

through Hired. Normally, you get $300 for finding a job through Hired, but if you use our link, hired.com/sedaily, you get $600, plus you're supporting SE Daily. To get that $600 signing bonus upon finding a job, go to hired.com/sedaily. Hired saves you time and it helps you find the job of your dreams. It's completely free. Also, if you're not looking for a job but you know someone who is, you can refer them to hired and get a $1,337 bonus. You can go to hired.com/sedaily and click "Refer a Friend".

Thanks to Hired for sponsoring Software Engineering Daily.

[INTERVIEW]

[0:05:28.9] JM: Dusan, and Jeff, and Joel, you guys are the lead authors of the paper; The Surprising Creativity of Digital Evolution. Welcome go Software Engineering Daily.

[0:05:37.7] DM: Thank you.

[0:05:39.5] JL: Thanks for having us.

[0:05:40.7] JM: Yeah, I'm very glad to have you. I really enjoyed the paper. When we talk about evolution, most of the time we're talking about biology. We all know about natural selection and how that process leads to increasing fitness. As engineers, what can we learn from the process of biological evolution?

[0:06:01.3] JL:I think that the natural world is full of engineering marvels. People very often forget about how wondrous the inventions of nature are that surround us. If I walked into a venture capitalist room today and I said, "I have a new technology. I'm holding it in the palm of my hand." It looks like a stone. But when I throw it into the earth and add water, it self-organizes and grabs all the molecules it needs from the soil and air and energy from the sun and turns itself into an oak tree. People would be astounded.

If I said I could do the same thing with a slightly different initial seed and get a dolphin or a three toed sloth, people would laugh at me. That's exactly what nature can do. It has self-assembling nano-technologies. It can build things far more complicated and intricate than we can dream of.

We talking about squirrels, they do tightrope acts across telephone wires and the thinnest branches of trees, monkeys that leap through the canopies, things that crawl in a dirt and swim to the deepest cabins of the ocean that provides sometimes their own light. It's truly astonishing and nature does that routinely every day.

So when we do, when we use evolutionary computation is we try to harness the process that produced all of the wondrous marvels in the natural world, and the idea is that if we can set the stage and find the secret ingredients and create a process that produces a tremendously sophisticated design that's matched to an appropriate problem, then we consider that a victory. That's why we try to harness evolution, in some sense it's for engineering purposes, which is to harness the same process that produce jaguars, hawks and humans to solve our technical problems. Another reason we study it is to try to ask and answer open questions in evolutionary biology, but that's more of a scientific motivation.

[0:07:49.7] JM: Right. So the term digital evolution refers to the idea of harnessing the evolutionary properties that you've witnessed in biology and applying those to solving problems in computer science other information technology fields.

Joel, how does that idea of digital evolution work in practice?

[0:08:15.1] JL: Right. Well, in practice, digital evolution means that you're going to take some inspiration from the natural world, take some inspiration from how evolution actually works in nature and abstract from it, something, in a form of an algorithm that you could then apply to your problem and interest, or as Jeff said, maybe to abstract thing that has some relation to biology that you could use to study an open biological question. But the overall sort of thing is to take some abstract property evolution. It could be the survival of the fittest is one way to look at it, but there are many other lenses you could put on and to create an algorithm from that.

[0:08:52.7] DM: You can think of evolution as being medium independent. So we know it from the biological world. We know it from evolution of hawks and squirrels and amazing things that we see around us in nature, but when you start defining it, you end up with definitions of a process that are not dependent on whether it's DNA or whether it's zeroes and ones. So I feel

like when we say digital evolution, we just mean a process of evolution applied to a digital media, or not even applied, but present in a digital context.

[0:09:25.4] JM: So these collection of anecdotes, this is what the paper was about, is these different anecdotes that represent certain observations that there are these processes in digital evolution, these ideas like selection gone wild, or automated bug discovery, or just different ways that digital evolution results in fascinating results.

So, Jeff, can you talk about how this paper came to light, the idea that you wanted to collect these different anecdotes from different researchers into a single paper?

[0:10:05.9] JC: Sure. So when I was a Ph.D. at the Michigan State University, one of my introductions to research in a lab were our lab meetings, and the core of those meetings were deep scientific dives into research questions. Often, in the process of presenting that work someone would say, “Oh, by the way, as I was kind of setting these experiments up, here is this funny thing that happened.” That happened regularly and those stories were really engaging.

At some point in time, Elizabeth Ostrowski, who was a member of the lab said, “We should try to write these down and capture these,” and this is after when my advisor had told a particularly great anecdotes about how clever evolution had been in a particular situation. Then offhand comment from Elizabeth always stuck with me that that would have been a great thing to do, because a lot of us in the field have been inspired by and motivated and in some cases decided to do research in this area because of the surprising creativity of evolution. Often it’s these sort of anecdotes that kind of really reveal that creativity and they don’t typically make into scientific papers and journals, because by definition they’re kind of the one-off random funny thing that happened in route to finally getting your experiment dialed in where they’ve gone rid of this mischief of evolution, because usually it’s subverting your intention as an experimenter.

Then as I kind of expanded out to going to conferences both within digital evolution and actually beyond, even to reinforcement learning and machine learning, I came to realize that this phenomenon was not isolated to the lab at Michigan State, but it’s actually more of a universal property not only of evolution but also of optimization in general, and that many researches who deal with kind of intelligence in a bottle or intelligence in a computer in the form of an intelligent

artificial intelligent algorithm that they're trying to experiment with frequently have these situations arise where the intelligence surprises them and outwits them and kind of subverts the intentions that they had going into the experiment, and we love telling these stories over beers or at the water cooler at conferences and get togethers, but they don't fit the standard model and so they don't tend to get into the papers and so they're just traded orally and there's no kind of original kind of fact checking to these stories.

Sometimes you don't know if the fish has gotten a lot bigger over the telling of the stories. You can't remember the details exactly right, and it occurred to us that maybe if we don't get serious and try to write these all down, many of these stories would kind of pass on and the antiquity as researches leave the field.

So there was a related paper recently that had an interesting approach, which was as opposed to going out and kind of interviewing every single person who's done work at a certain area and then one person or one small team of people kind of collating all of the results and writing them up and having to read an article that's written by three people, in a modern world you can kind of crowd source these collaboration amongst scientist. So we decided to follow the model of that paper and Joel and Dule and I put out a call for anecdotes where we said, "If you had one of these amazingly hilarious events happen in your lab, where evolution or any kind of optimization has surprised you, send us in a description and then you can join on the paper and we'll have kind of the firsthand story from the original source and have it all compiled so that posterity has a written record of these really interesting anecdotes."

[0:13:23.1] JM: So you crowd sourced these different stories about digital evolution, and the first class of evolutionary results that you explore is this selection gone wild. This is when evolutionary principles that you put in place in your simulated model lead to counterintuitive results. So you put some evolutionary pressures on a system and you let it run wild and then it does strange stuff.

There was a few examples that you talked about. One is, I believe it was a creature that had a bunch of different variables that it could evolve over each evolutionary period and it was supposed to learn to do a summersault efficiently, which is where you kind of flip over. But then

because the way that it was optimized, the variables were input in a strange way. The creature instead learned to grow very tall and then just fall down to summersault instead of jumping.

So intuitively we would think, “Okay. In order to summersault, you need to jump off the ground to flip over,” but actually this creature just learned instead if you grow really tall legs, you could just fall over and flip, which is hilarious. There was another one where this program repair application, there was a program that was created that was supposed to sort a list of numbers, and it was a broken program. So there was a broken program and the program was supposed to – Through evolutionary iterations, was supposed to learn to sort a list of numbers and instead of learning to sort the actual list of numbers, it just returned an empty list. So that was its evolutionary adaptation, which is didn’t break the rules. It was very strange and kind of annoying, but it didn’t break the rules.

Dusan, do you have any particular results, any lessons that you took away from the selection gone wild cases?

[0:15:13.5] DM: I guess in general, thinking of a selection gone wild, is that we think we know what we select for, or we think we understand what the evolutionary pressures are, and then you get something completely different that still satisfies your question or still passes whatever test you put in front of the evolutionary algorithm, in front of your digital evolution. So I guess the overall lesson to me has been that we need to be more careful about what we ask for and that we need to understand better how to define these questions and these evolutionary pressures if we want to get what we want, otherwise we’re going to end up with funny stories, but not necessarily the results or the outcomes we were looking for.

[0:16:03.1] JM: Indeed, and I think one way this applies to our modern software engineering practices is that choosing reward functions can guide how a system evolves. Jeff, have you seen any examples in practice where choosing a reward function sub-optimally leads to sub-optimal results?

[0:16:29.2] JC: I would say it’s almost a daily occurrence in doing research in this field. As Dule was just saying, evolution very frequently will do exactly what you asked it to do, which is almost never what you wanted to do. The lesson of this research is that we don’t know how to specify

what we want very clearly, and this is actually an old problem. There's many myths dating back to antiquity in many sitcoms that are all based on the premise of asking for something and having somebody do that when that's not really what you want.

Frequently, the myth of the genie is you get three wishes and all of your riches turn out horribly poorly because you literally said something different than you actually want. So what happens frequently with these evolutionary algorithms is that they optimize the letter of the law, or the letter of the request and not the spirit of the request. So almost every example is kind of a fallout of that.

You already mentioned the case where evolution was challenged to return a sorted list and we thought we knew how to specify a measure of that, which is how many things in the list are not in order? An evolution returns an empty list, because nothing in that list is out of order. Another case is a game of connect four, where these relational algorithms are pitted against each other and we thought we know how to measure the score in a game, which is you beat your opponent or they run out of time to respond, because it's a timed game.

What evolution learned how to do was create a memory, a bug in the opponent's code by setting a piece so far out on the edge of the board that the opponent has to allocate enough memory to think of this almost infinitely large board and that crashes the program, it never responds, and then evolution wins by default.

Another one of my favorite examples is the one that launched this entire paper, which was the anecdote out of Michigan State about 10 years ago, and that is my advisor is trying to study what happens when evolution cannot have beneficial mutations? So most mutations are harmful in an evolutionary process, but every once in a while a mutation is helpful, and that's what fuels all evolutionary adaptation. But he wanted to study, "Okay. What happens if there are no positive mutations?" So he decided to create an experiment in which he basically checks if any offspring was better than his parent, and if it was, he would just delete the offspring and revert to the parent.

What happened was that evolution effectively flat-lined for a while. It wasn't getting any better. But all of a sudden fitness started going up again. It started improving again, and he's like,

“That’s impossible. I asked it to not allow any improvements. What’s going on?” He dug in and what he realized was that he was testing whether or not this programs were gaining skills, because he was giving them a test. He was giving the offspring a test to see if they’re any better than their parents.

What the offspring learned to do was to play dumb on a test. They’re like, “Oh! If I pass this test, if I do well, you’re going to kill me. So I’m just going to play stupid. Fail the test. That’s good. I get to be let out into the world and then I could advantage of the fact that I’m actually quite smart and I was just playing dumb on a test.”

[0:19:28.7] DM:I want to tell something. When you list all these examples, it may sound like these are bad things because we didn’t get what we asked for. But they’re very interesting exactly because they surprise us, exactly because we didn’t get what we asked for. But also it’s evolution finding a simpler way to do something. So it usually tells us that we overthought things and that we thought of a more complicated solution and evolution somehow found a simpler one. That’s part of the excitement and the amazement of it.

[0:20:01.8] JM: Absolutely. I mean, there are certain practical applications here, and when you think about the idea of human evolution, it’s based on mutation, and in most other contexts, the word mutation would have a negative connotation. But in evolution we know that mutation is pretty much a bipartisan thing. Sometimes it results in advantages to the species. Sometimes it results in things that make you more prone to developing sickle cell anemia. It can really go either way, and I think we see the same thing in digital evolution.

[0:20:37.5] JC: Yeah. In fact, it’s funny you mentioned that, because another case of evolution surprising us has been that evolution itself figured out that mutations aren’t always a good thing. Frequently, we as researchers are trying to use evolution to come up with a solution, and that means trying out a lot of things, in the words of Silicon Valley, to fail fast. So you want evolution to try a whole lot of things, which means trying having a lot of mutations and then getting rid of the ones that don’t work and grabbing the few that do work.

But the thing is, is that frequently we tell evolution to do something like increase your average performance overtime. Well, on average, mutations are harmful. So if you give evolution the

ability to control its own search process, which is to control its own rate of mutation, most people thought they would be a very good idea because evolution would then find exactly the right mutation rank to get the ultimate engineering solution.

But what we found, Dule and I in particular and some other researchers, is evolution almost always just turns off mutations. So basically it refuses to look for anything better, because you told that on average to try to improve its performance, and the best way to do that is turn off search completely and stay with a current performance, whatever it is.

It's one of these cases that in the short term average performance is hurt by mutations, but it's essential in the long term for adaption. So, yet again, you naively ask for something, like improve your average performance and you realize that in some subtleties you didn't think about in terms of a schism between short term and long term performance and evolution ended up optimizing for one in what you ultimately cared about with another.

[0:22:15.6] DM: In a way, we talk about debugging in a different sense, but you could say that evolution is kind of debugging our thinking process, that it's finding holes in arguments and it's correcting them.

[SPONSOR MESSAGE]

[0:22:34.1] JM: Users have come to expect real-time. They crave alerts that their payment is received. They crave little cars zooming around on the map. They crave locking their doors at home when they're not at home. There's no need to reinvent the wheel when it comes to making your app real-time. PubNub makes it simple, enabling you to build immersive and interactive experiences on the web, on mobile phones, embedded in the hardware and any other device connected to the internet.

With powerful APIs and a robust global infrastructure, you can stream geo-location data, you can send chat messages, you can turn on sprinklers, or you can rock your baby's crib when they start crying. PubNub literally powers IoT cribs.

70 SDKs for web, mobile, IoT, and more means that you can start streaming data in real-time without a ton of compatibility headaches, and no need to build your own SDKs from scratch. Lastly, PubNub includes a ton of other real-time features beyond real-time messaging, like presence for online or offline detection, and Access manager to thwart trolls and hackers.

Go to pubnub.com/sedaily to get started. They offer a generous sandbox tier that's free forever until your app takes off, that is. Pubnub.com/sedaily. That's pubnub.com/sedaily. Thank you, PubNub for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:24:17.3] JM: Speaking of bugs, there is one class of evolution that you talked about called automated bug discovery, and this is where you have a simulated environment and you try to search within that environment for bugs, but if you don't constrain the environment correctly, like if it has a bug in the simulation code, you have surprising results. For example, you could simulate the physical world and you have a creature evolve in the physical world, but if you designed the physics engine wrong so the creature evolves in ways that defy real-world physics, well, that's not exactly what you were looking for. Dule, what can we learn from the anecdotes of automated bug discovery?

[0:24:59.1] DM: That we are not so good at programming as we think we are. Speaking of frequency of things and how often we see – Jeff said it's every day, and I think anybody who try to code up a simulation of anything had this problem that something weird and strange happens and you go back and you look, "Oh! Well, actually, I put a minus sign where a plus sign should have been, or there is this edge case in my physics engine that I did not take care of."

So I guess it teaches us how to be a little bit more careful encoding, but also it's really like a – It acts like a debugger, and overtime as researchers, we make may better at being careful, but there are always bugs and evolution will find them quicker than we can.

[0:25:48.1] JM: Simulations are widely used in real-world applications. One that comes to mind would be if you wanted to simulate a human heart in order to device to a customized stint for the

human heart, why would that be a situation where the consequence of a poorly-defined simulation would have dramatic consequences?

[0:26:13.0] JC: One of the things that we've seen over and over again in this field is that evolution is a very clever and capable optimizer and it will optimize the exact conditions that you asked it to. So if you're completely evolving a solution in simulation, it will customize that solution to the laws of physics in that world, and if those laws of physics were differently than in the real-world, then you might get a solution that works in simulation but doesn't transfer to the real-world.

So there's an entire almost field of research into getting solutions based on AI or evolution algorithms that work in simulation and then transfer to the real-world. In fact, we had a paper that was on the cover of nature that did exactly that. It harnessed an evolution and an algorithm offline ahead of time to evolve a lot of solutions, but then required in a different machine learning algorithm to customize those solutions to the real-world, because they don't just work out of the box, because evolution is too good at what you ask it to do, and that is only perform well in simulation and not also in reality.

[0:27:15.3] JM: So now we're getting in practical applications. Dule, if I wanted to use the digital evolution in a pragmatic way within a machine learning pipeline, for example, how could I make use of that? Where would I put it into my machine learning process?

[0:27:37.2] DM: So I think there are many ways you could put an evolution. You can use evolution to search for better parameters for your machine learning. You could use it to debug your process and to see if you get what you aimed for. I think, as Jeff mentioned, one way to think about evolution is an optimization process and it's not the only thing. It's not how we define it in biology, but it is one of its features.

Basically, anytime you need to optimize something, may it be performance time or precision or how well it works, and in different situation, maybe it's versatility. Setting it up in an evolutionary context where you try a bunch of things, you select the best ones, you mutate them a little bit and you try again will lead you to have better results.

[0:28:29.5] JM: So these anecdotes from your paper, they hadn't been discussed or collected until you wrote this paper. Why is that? Why hadn't these been explored in finer detail before?

[0:28:41.0] DM: One major reason for that is the way we do science, which focuses on positive results. So we focus on things that work. Also, many of these anecdotes were sort of things that happened during a long research process of looking into a particular question. There was a bug. There was something weird that happened. We fixed it. The point in time when we're writing the paper, this is not interesting in a classical academic sense. What's interesting is what was the hypothesis? What kind of methods did you use? Were you able to disprove or to support that hypothesis? But the intricacies of how things went wrong at month five out of a two-year long project are not what makes it into papers.

This was exactly also the motivating factor for us, is because we knew that these things are out there, but that there was no room for them in a classical academic way of telling stories about their research, about writing these papers. So we wanted to find a way to preserve them, to share them with our community, but also more broadly with people who are interested in evolution in general or digital evolution more specifically.

[0:29:55.2] JM: Dule, do you have a favorite example of when evolutionary algorithms produced a result that exceeded the expectations of anybody involved?

[0:30:07.5] DM: There are many that I'll have to do a quick choice. But let's say there is an example from a work that we did in our lab here in Paris with a fantastic postdoc [inaudible 0:30:17.2] now postdoc elsewhere, and we were studying evolution of cooperation. So why individuals might cooperate with each other even when whatever they're doing actually caused them directly. So this is one of the big evolutionary questions. There's a lot of work on it in all sorts of systems, so including digital ones. We managed to evolve digital individuals that cooperated, and as a way to analyze them, we actually increased the cost of that cooperation. We expected, "Okay. So they're going to stop cooperating, because it's too costly. The overall benefit is not big enough to offset the direct cost."

However, we saw that consistently, some of our evolved digital organisms continues to cooperate, and we weren't sure why it was and spent many months trying to sort of figure out

what happened. It turned out that in their digital encoding and the way their genomes were set up, they put their genes for cooperation literally on top of genes for metabolism, for things that are directly rewarded.

So they managed to find a way to hide their cooperation genes and protect them from mutations, because any mutations that those genes have a high chance of having a direct fitness cost due to these metabolic genes that overlapped. Basically, without looking for it and we managed to find another way of maintaining cooperation that seems to also exist in nature, although it's unclear what is its significance, how often this happens, but there's certainly overlapping genes in nature and some of them maybe cooperation genes.

To me, this was not a simple example, maybe not one of these quick sexy things, but it uncovered a very interesting mechanism and it certainly went beyond what we expected when we set out to do this study. In a way, it's also an example of an anecdote that did it make it in the paper because we did write it like this and we told a story of, "Look. We tried this and it didn't work out, and then we discovered something else."

[0:32:27.3] JM: Jeff, do you have anything to add to that answer?

[0:32:29.6] JC: Yeah. So another example that I loved where evolution exceeded our expectations was it was in the case where we thought evolution couldn't solve a problem at all and it did so quite cleverly. In our nature paper, one of the things that we asked evolution to do offline in simulation ahead of time was find a variety of different ways to walk, and this was a six-legged robot and we defined different ways to walk as the different percent of times that the feet of the robot would touch the ground.

So we wanted all possible combinations of using all six legs a lot, using these five legs a lot and the sixth leg not that much, etc., etc., and there was one corner case and that vast array of possibilities, and that is try to walk without ever having any of your feet touch the ground. We thought it was impossible for the robot to walk without using its legs, and when the first author pulled up the data, he looked in that cell which we thought was going to be empty. In fact, we almost considered not even including the cell, but it was just easier to leave it on the code, and

sure enough there was a pretty good solution on that cell and he actually thought, “Oh, no! I have a bug. I’m going to have to rerun all these weeks of experiments,” and he was really sad.

So he pulled up the video of the robot walking in that cell and it turns out that the robot was walking quite happily by flipping on its back and walking in its elbows. We’ve met the letter of the law, which was walk without touching your feet to the ground, but also accomplished the task, which was move as fast as possible. So it’s just quite amusing to see this video of this robot almost winking at you, flipping over on its back and then happily crawling along on its elbows instead of in its feet.

[0:34:07.5] JM: Indeed. Is there anything else that digital evolution can teach us about real-world biological processes?

[0:34:17.3] JC: Sure. So both Dule and I come from a lab that tends to use this technology not just for engineering advances, but also because we’re trying to ask an answer, open questions in evolutionary biology. So, for example, I mentioned this earlier, but a lot of people thought that evolution, if it was going to control its own mutation rate, would do a good job of picking a mutation rate that was beneficial for long term adaptation, but Dule and I have a paper. We basically created this simulation of evolution and we showed, in fact, now evolution will turn off its mutation rate if given the possibility and turn off its own ability to adapt, because it’s short-sided, and evolution doesn’t have foresight. It doesn’t care about the long term benefits of evolution search. It’s just immediately trying to improve its average fitness, and there are many other examples where we ask questions.

Dule alluded to some work that he has done and I’ve also done some work on the evolution of altruism. So we allow these organisms to be nice to each other, or not, and we see under what conditions does altruism evolve. We also see – Dule’s done a lot of work into under what conditions do you see sexual reproduction emerge as supposed to asexual reproduction. Basically, almost any question you can think of in biology. It’s an abstract question about the dynamics of evolution and it’s not dependent on the specific molecules involved in an evolutionary process.

You can create a simulation that probes that question and often times do it with a lot more speed and ability to gather data and control experiments very precisely because it's all happening in a computational world. But as this paper points out, you might think you have perfect control, but because you're dealing with this creative adversary and evolution, it sometimes has its own motives. You very frequently think you've got everything dialed in and it's going to go exactly according to how you think your plan and evolution has its own idea for what it wants to do.

[0:36:11.9] JM: The practical applications of the results of this paper are not just limited to the idea that we could use digital evolution to develop more effective models and explore a problem space more effectively. There's also implications for AI safety, and I think that's because it's hard to anticipate how you can have a system evolve if you give it a wide open space. What are the implications for AI safety?

[0:36:48.0] JL: So I think that the lessons for AI safety are significant, and they are that we are not as clever as we think, or there's a famous quote that says, "Evolution is clever than you are." We often think we know what we're going to get when we launch an optimization process and often times we do not anticipate all the ways that it can go wrong, and this is not just confined to dealing with evolutionary algorithms or even machine learning more broadly. It's also true of any incentive structure. Whenever you create some sort of game or an incentive system and which people are supposed to do something according to a set of rules, if you're dealing with intelligent agents that are optimizing for their own best interests, they will often time find loopholes.

So whether or not it's tax law and people finding loopholes there, or international trade regimes or environmental protections and people finding loopholes, or teachers in a classroom thinking that they're going to create some cooperative dynamic because they allow peers to grade each other, finding out that their peers are shanking each other's grades to try to raise their total grade because things are graded on a curve.

Over and over and over again we launch these kind of natural experiments with intelligent agents an it's very difficult to predict ahead of time the result of the dynamics of loss of intelligent agents that are trying to optimize exactly what you ask for, which again may not be

what you want. So there's kind of – I think the overall lesson from engaging with an intelligent optimization algorithm on a daily basis is humility, and not to be too heuristic and think that you will get it right the first time.

In fact, I think everyone who deals with intelligent evolutionary algorithms, machine learning and probably human optimizers has learned that they should plan for iteration and failure. The first couple of times, weird things are going to happen and you're going to be impressed and surprised with how the system and the agents reveal to you that what you asked for it's not exactly what you want and you'll have to refine the process and ultimately you will get there.

But what you shouldn't do is, for example, launch a process, assume you've gotten everything right, when the consequence for going wrong are serious and catastrophic, and that's where AI safety comes into place. You definitely want to think as much as you can ahead of time about what might go wrong. Use the lessons learned. We've all developed a set of heuristics , kind of an instinct for how evolution is going to misbehave and its typical tricks and how we can kind of head them off at the pass.

Ultimately, what we have is a sense of humility and modesty and assumption that we're not going to get it right, and therefore things will go wrong and we want to make sure that it's properly sandboxed such that when things go wrong there aren't consequences for humans.

[0:39:34.2] DM: Without a doubt, exactly along the lines of what Jeff said is – So we've been doing this in the digital media for several decades, and some of the examples go back to the, at least, 80s and even earlier. So overtime we developed ways to – Prevented ways to make sure things don't go horribly wrong to sandbox it, but then also look at evolutionary biology. People have been aware of these things in the lab or in the field for a long time. We've tried to work with evolving systems for hundreds of years and things go wrong. You introduce rabbits somewhere where you shouldn't and then they eat everything.

We're not new to the idea that evolution outsmart us and maybe the message is that this happens no matter what the situation is. It works in nature. It works in the lab. It works in our computers and in the context of digital evolution. The same thing will happen with AI. So we can

maybe look for experiences from these other domains and apply them to AI work to put those safeguards in that are necessary.

[0:40:43.7] JC: Yeah, I love that answer, especially because many people think that because it's a computer program that it couldn't possibly surprise you. They say all the time, "I don't understand. How is it surprising you when you wrote the program?" What we basically have to explain to them is that we set up the very, very basic rules by which an evolutionary process gets off the ground. Then it really is evolution taking it and running with it from there and we are frequently and constantly surprised with what it comes up with.

So the fact that you wrote the kind of initial rules of the game doesn't mean you know how the game will unfold, and that's what we see day-in and day-out with evolution, and that is a message for people who are not used to thinking about evolutionary algorithms and machine learning algorithms being remarkably creative.

The proof there is that, frequently, evolution does come up with solutions that we never would have dreamed of, and in some cases we don't understand, and if that's not proof that evolution can outsmart us and do things on its own that we didn't program in, I don't know what is.

[SPONSOR MESSAGE]

[0:41:51.9] JM: We are running an experiment to find out if Software Engineering Dailydaily listeners are above average engineers. At triplebyte.com/sedaily, you can take a quiz to help us gather data. I took the quiz and it covered a wide range of topics; general programming ability, a little security, a little system design. It was a nice short test to measure how my practical engineering skills have changed since I started this podcast. I will admit that though I've gotten better at talking about software engineering, I have definitely gotten worse at actually writing code and doing software engineering myself.

But if you want to check out that quiz yourself, you can help us gather data and take that quiz at triplebyte.com/sedaily. We have been running this experiment for a few weeks and I'm happy to report that Software Engineering Daily listeners are absolutely crushing it so far. Triplebyte has

told me that everyone who has taken the test on average is three times more likely to be in their top bracket of quiz course.

If you're looking for a job, Triplebyte is a great place to start your search. It fast tracks you at hundreds of top tech companies. Triplebyte takes engineers seriously and does not waste their time, which is what I try to do with Software Engineering Daily myself, and I recommend checking out triplebyte.com/sedaily. That's T-R-I-P-L-E-B-Y-T-E.com/sedaily. Triplebyte, byte as in 8 bits.

Thanks to Triplebyte for being a sponsor of Software Engineering Daily. We appreciate it.

[INTERVIEW CONTINUED]

[0:43:50.4] JM: This episode is a good compliment to the one that we published today actually with Dario from OpenAI. He was talking about AI safety, and the emphasis he proposed is that AI safety in the context of accidents is a concrete problem to address. So we can talk about whether or not AI safety is going to be relevant in the domain of malicious activity. We can talk about that. But where it's definitely relevant is accidents. If you develop an AI system in the context of a wide open decision space, there will be accidents made. Do we have any approaches to preventing accidents in the realm of these fast-evolving systems?

[0:44:42.9] JL: Could you define what you mean by accident?

[0:44:45.3] JM: By accident I mean a system that does something that is negatively harmful that we wouldn't have wanted.

[0:44:55.1] JL:I would just point out that most researches currently do not launch optimization algorithms out there in the wild that just kind of – The typical way that it's done is that you launch a learning process or an optimization algorithm in some sort of controlled environment or sandbox and then it might produce a solution, and then you have a whole second process by which you try to see whether or not that thing that was produced is safe and is respecting the values that you have in terms of deploying it.

But it is very much an AI safety issue if people start to deploy these optimization processes in the wild, and that is increasingly being done. So that's kind of where these are lessons to people who are thinking of doing that, that there may be things that go unexpectedly wrong and you should be humble about your ability to anticipate them.

[0:45:50.8] JM: Yeah. I mean, even the domain of social media. We would have never thought that such an area could be so damaged by accident. I would argue that the creators of the algorithms for social media, what they created or what happened in actuality with people being able to, I guess, hijack. I don't know if that's the right word, but take advantage of those algorithms. That was an accident. That was not on purpose, but it made it out into the wild nonetheless.

[0:46:24.4] JL: Yeah. Just working with intelligent agents that have their own agendas and their own goals is kind of a difficult thing to anticipate all the ways that things might go wrong, and whether or not those intelligent agents are AI algorithms optimizing according to some objective or human optimizing according to whatever they are trying to accomplish, the highest level takeaway, I think, is that who knows what's going to happen? These are complex dynamic systems with lots of interacting parts and intelligence adopt and it finds a way.

[0:47:04.0] JM: Jeff, I asked Dule this a little bit earlier, but I want to phrase it to you as well since you're working in a heavily applied space. When you think about machine learning, there are places where you could insert notions of digital evolution. How can you apply digital evolution to get better practical results?

[0:47:29.6] JC:I would say that thousands of Ph.D. dissertations and papers have been dedicated to that question. It is a rich area of research how best to do that, and many of, if not all for the authors on our paper, and there are something like 30 to 50 or something, have dedicated much of their careers just studying those questions. There are high-level answers and there are low-level answers. There are thousands of different tricks that you can use to try to harness evolution, to help it accomplish your goals and to catalyze its ability to innovate.

It's very difficult, I think, to give an overall summary, except just that it's exceptionally complicated. On a daily basis, we take ideas from statistics, from neuroscience, from

evolutionary theory, from game theory and we try to blend them and mix and match them and study them to try to get evolutionary processes to innovate and solve engineering challenges. Also, as I mentioned, to try to study some of the most fundamental aspects of science.

For example, one of the most amazing open questions and challenges in science is where did all the complexity that surrounds us come from? How do you launch an open-ended process that continues to innovate and eventually produces three toed sloths, jaguars, hawks and human beings and continuously innovate? Imagine if you could do that in the space of novels, or screenplays, or paintings, or software code. Imagine if you could create an endlessly innovative process that would do nothing but write new interesting types of code that solves new process, new problems or creates new art forms?

So we have a lot to learn from nature. We have learned a lot, but we have a lot more to go to try to unlock the secrets that made the complexity explosion that surrounds us and impresses me every single day a reality in silico

[0:49:31.0] JM: To close off, what are each of you working on? Why does digital evolution interest you?

[0:49:37.8] JC: Well, since I just gave that answer, I will dovetail on it. A lot of reasons, things I just mentioned, are motivations to me. I'd like to understand where we came from in terms of what are the processes that gave rise to what Darwin called the "entangled bank", which are all of the myriad of amazing creatures solving problems in very different ways and how did that process kickoff and sustain itself and how did it ultimately lead to humans? Also, how does thinking happen? How does intelligence emerge? What is the generality of intelligence?

In some extent, boring interstellar travel, we don't have intelligent alien species to interact with such so we could learn about the space of intelligence. How different can you be, but still be an intelligent? What is alien culture and art and political science and morality look like?

Well, one way to study those questions will be to invent artificially intelligent life within the computer or maybe sets of very, very different types of AI, and then we get to learn from and interact with those alien cultures by creating them in a bottle or in our computers. To some

extent, being an AI scientist is a dream job and that you get to study some of the deepest, most profound questions of are origins and ultimately the set of all possible intelligent agents that might exist.

[0:51:02.5] DM: Yeah, I'll try to give a slightly different answer. When I look at biology, there is large parts of it that try to answer the question of how something works. So how a particular set of molecules interacts with another set of molecules, how genes are expressed? There is a lot of mechanics, a lot of biochemistry, a lot of biophysics. For me, studying evolution is asking and trying to answer the questions of why does it work like this?

To understand why we have certain features is to understand how they evolved, and there are many ways you can try to answer those questions. I like the digital evolution in part because it tends to be quicker than anything else that we can do. To my students, I often show a picture of Darwin and jokingly say, "I want to answer this evolutionary questions, but I don't want to look like this guy before I'm done," because it's usually a picture of him very old with little hair and a long gray beard.

So there is this appeal of being able to do things quickly, of being able to do things in a very controllable way in spite of what we just talked about of those things go wrong and how they surprise you. It's still a lot more controllable, anything that you can do in the lab or in the field. Finally, I completely agree with Jeff. There is this appeal of strangeness of something different, of a different instantiation of evolution or of a different instantiation of life. The same we're excited about science fiction is because it gives us different worlds. Our digital evolutions gives us different worlds in which to study evolution and try to understand it better.

[0:52:44.2] JL: Yeah. One of the things we wanted to capture in this paper is that when you get to interact with an intelligent force that is within your computer, there is a certain joy and exhilaration and not knowing what's going to happen. So ever since my first days as a Ph.D. student through today we will dream up an experiment, dream up a world, dream a challenge for AI or evolution and we will hit a button that launches a process, and then we'll go sleep. There is moment in the morning where you wake up and you realize that there's a present to open up and see what's inside. You're going to turn on your computer and you're going to see what did evolution come up with this time? How creative was it? How did it solve that problem? Did it

work? Did it not work? Often times, you're completely blown away with what you find and there's just this kind of smile and smirk and of admiration for a creative force that you don't have full control over.

[0:53:42.3] JM: Okay. I know we're up against time. Jeff, I wanted to ask you one more question because you are so ebullient about your job, the dream job, and I agree with you. I love studying this stuff and talking to people such as the both of you and Joel is a dream job for me as well. There are naysayers today. There are people who talk about deep learning being overhyped, machine learning being overhyped. Do you have a perspective on that? Are we on the verge of another AI winter?

[0:54:15.5] JC: So what I love about this job is that we are simultaneously trying to invent new machine in the algorithms that are going to take us in the next phase of AI, but also that the current AI is working tremendously well. So even putting aside whether or not it ultimately will take us to the next phase or to AGI, the current AI that we use on a daily basis is extremely helpful. It's extremely helpful Uber, throughout many different areas of its business. It's helpful for Google and Facebook and Twitter and, actually, any company that's using machine learning and deep learning is probably finding that as tremendously effective.

So after decades of AI not working, for the first time, really, it's working tremendously well across a number of different domains. I do think that deep learning is not overhyped. I think that it has tremendous benefits and it's really moved the needle. I also personally feel that it's going to probably be part of the solution that takes us all the way home. I think other technologies will get woven in with it but it's definitely a piece of the solution.

To some extent, whether or not deep learning itself is overhyped gets a little bit into the semantics of what you mean exactly by deep learning, and I think in the end it's going to be part of the solution and part of the things that some people think are missing are going to be woven in. But it's also a research problem. It's an open area. We don't fully know the answer. It may require complete paradigm shift to get to more sophisticated formats of intelligence. But my money is on something like deep learning that is enriched with other techniques and other ideas and more complicated environments in which to operate that ultimately leads to the next major sea-change in AI research.

[0:55:59.1] JM: Well, Dule and Jeff, I want to thank you both for coming on the show. I know we lost Joel at some point, but I will follow up with him and see if we can do something else. Thanks, guys. I really enjoyed the paper. It was very thought-provoking.

[0:56:10.8] DM: Thank you.

[0:56:11.4] JC: Thank you.

[END OF INTERVIEW]

[0:56:15.1] JM: GoCD is a continuous delivery tool created by ThoughtWorks. It's open source and free to use, and GoCD has all the features you need for continuous delivery. Model your deployment pipelines without installing any plug-ins. Use the value stream map to visualize your end-to-end workflow, and if you use Kubernetes, GoCD is a natural fit to add continuous delivery to your project.

With GoCD running on Kubernetes, you define your build workflow and let GoCD provision and scale your infrastructure on-the-fly. GoCD agents use Kubernetes to scale as needed. Check out gocd.org/sedaily and learn about how you can get started. GoCD was built with the learnings of the ThoughtWorks engineering team who have talked about building the product in previous episodes of Software Engineering Daily, and it's great to see the continued progress on GoCD with the new Kubernetes integrations. You can check it out for yourself at gocd.org/sedaily.

Thank you so much to ThoughtWorks for being a longtime sponsor of Software Engineering Daily. We are proud to have ThoughtWorks and GoCD as sponsors of the show.

[END]