SDS PODCAST EPISODE 973: AI SYSTEMS PERFORMANCE ENGINEERING, WITH CHRIS FREGLY

Jon Krohn:	00:00:00	My guest today spent $6,000 at Starbucks writing a thousand page book that NVIDIA's own documentation couldn't provide, and what he found under the hood of GPU computing will update everything you know about engineering AI systems. Welcome to episode number 973 of the SuperDataScience podcast. I'm your host, Jon Krohn. Today's guest, Chris Fregly, has worked as a principal engineer and AI systems performance specialist at AWS Databricks and Netflix, and now he's recorded his vast knowledge into his third O'Reilly book, a massive invaluable tome called AI Systems Performance Engineering. In today's episode, he distills the books thousand pages down to its most valuable takeaways. Enjoy!
	00:00:40	This episode of SuperDataScience is made possible by Anthropic, Cisco, Acceldata and the OpenDataScience Conference.
	00:00:49	Chris Fregly, welcome to the SuperDataScience Podcast. I've been trying to get a conversation with you for a decade and it's finally now happened. I heard. Yeah. Welcome to the show.
Chris Fregly:	00:00:59	Yeah, man. Thanks for having. Yeah, I heard you had one of my friends Antje on a recent podcast and yeah, she said the same thing that you were trying,
	00:01:08	I was like, let's do it. So the second I found out I sent you an email, but awesome.
Jon Krohn:	00:01:12	Antje was an episode number 963. She's absolutely brilliant. As you know, you've books with her in the past. You're a three-time O'Reilly book author, and in this episode we're going to talk mostly about your latest book. It's called AI Systems Performance Engineering, and then it's got a really long subtitle, optimizing Model Training and Inference Workloads with GPUs, CUDA, and PyTorch.

And it's not just the subtitle this long. This book is a thousand pages, including the indices stuff. It's 1,060 pages, and there's photos of you if people go to your LinkedIn profile and they see your banner at the top. There's photos of you with the book, and I love that you're holding the book sideways so that people can see what a thousand page book looks like.

Chris Fregly:	00:02:05	It's a banger. Yeah, a thousand pages. It's funny, I went back through all my Starbucks bills over the last year because I spent, basically, when I left Amazon, I just started working on this book to make sure I understood every aspect of the Nvidia stack. And yeah, I'm sure we'll get to it, but I went to Starbucks pretty much every day for a year, and my bill was over five or $6,000 I think, just on lattes alone.
Jon Krohn:	00:02:39	And you go to Starbucks to read or to write as well?
Chris Fregly:	00:02:42	To write? To write, Hey. Yeah,
Jon Krohn:	00:02:44	You write in the coffee shop.
Chris Fregly:	00:02:46	Yeah, I used to write at bars. Well, I still do write at bars. I write anywhere I can. I'm not good at sitting in a quiet room. I'm good at chaos around me. And then, yes, Antje by the way is not that way. And so she and I have very, very different writing styles, very different work styles. Yeah, she prefers,
Jon Krohn:	00:03:07	I've never actually seen him there, but for years I lived in the West Village of Manhattan and supposedly Malcolm Gladwell would frequent this coffee shop that was next door to my apartment, and for the same reason, he would kind of, Malcolm Gladwell would come in for two or three hours a day and just write. So it works for some people for one of the bestselling authors of our time. So cool strategy and yeah. Is it all of that coffee that resulted in it

being a 1000 page book? How does that, I guess first of all, tell us why you felt you needed to write this book, AI Systems Performance Engineering, and did you expect it was going to be a thousand pages when you started? How did that happen?

Chris Fregly: 00:03:48 Absolutely not. No. The book started off, I had proposed it, so it's my third time. So proposals can be a little bit looser with O'Reilly. They've already vetted me and I've already got book sales behind me and they've seen the quality of my work. So really it was scoping out what I thought was going to be mostly PyTorch, maybe one chapter on Cuda and then the rest on the hardware, GPU hardware Blackwell. And keep in mind, when I started writing this, Blackwell was more a spec and had not even been taped out yet, I don't think, by Nvidia. So I was just going off a lot of information and then, no, it started off, the proposal was 300 pages when I got, and the purest motivation was when I would be working with customers back at Amazon. I was more focused on the NVIDIA stack back at Amazon than I was training.

00:04:50 I had a little Tanium experience, but mostly Nvidia customers, and I was having a hard time getting information and I was going through NVIDIA forums I was getting, which are not that good by the way. There's a lot of unanswered threads on the Nvidia site. The Nvidia documentation's a total disaster trying to find things from Twitter from XI was getting more information from X and actual practitioners. And so that was actually the signal for the first two books as well. The first one was really about data science and specifically SageMaker and that whole ecosystem, which there's a million products within AWS, there's glue and there's EMR and there's Redshift and all that. And so the inspiration has always been, it's really hard for me to find the information and I know there's a lot of other people that are trying to find this information. So let's get this into a book form.

Jon Krohn:	00:05:54	Yeah, it's been hailed as the missing manual for the industry, a comprehensive encyclopedia that teaches you how to optimize hardware, software, and algorithms together for the world's most powerful AI models. Let's start off with chapter one. So chapter one of the book highlights how the brute force approach to AI is now being challenged by high efficiency engineering. So you talk about, for example, deep Seek R one, which is now, it's been about a year since it was released, and it allegedly costs less than 6 million to train while comparable models. The oh one, the three out of OpenAI for example, supposedly cost hundreds of millions of dollars. And so what were the kinds of performance engineering choices that made that 10 to 20 X cost reduction possible?
Chris Fregly:	00:06:44	Yeah, and what's really interesting of course, is that due to the Chinese export restrictions, US export to China, China in theory cannot get ahold of the best chips. So they had to find clever solutions. One that stands out is they found essentially undocumented or an undocumented, I would say all of Nvidia stuff is really under documented, but they found one in particular way to bypass a particular cash or short circuit. I would say just keeping it at a high level, which led to huge, huge performance gains in this sort of limited capacity chip. And it's something someone asked the OpenAI folks if they have discovered the same thing, and Sam Altman and Company said that they're positive that deep seek or that they being open AI has found all of the same tricks that Deep Seek has found without getting into details. So these things aren't necessarily that sort of revolutionary, it's just no one talks about them.
	00:08:00	The labs have them as their secret sauce. And it just so happens that Deepsea was open source and they talked about this, a lot of other innovations though, I mean Deep Sea, what was called the open source week that I

mentioned in the first chapter. And really it was not just sort of CUDA type innovations, hardware innovations, but it was co-designing, which is the term that is starting to be used quite a bit. They figured out ways that they can modify algorithms, they can modify the actual software and they can tune it specifically for their set of hardware. Also included in that hardware is the storage system. So they actually wrote their own storage layer, which is very interesting. This is something Jensen has just started talking about more and more, I think it was last year's GTC, whereas the first time that you really hear Jensen and the NVIDIA folks talking about the storage layer, that storage layer being extremely critical, it is a bottleneck, basically picturing all the different bottlenecks, memory, access and compute is basically on a single machine or a single cluster. We've got a lot of compute. It's just all the other bottlenecks, which is like retrieving data from memory from storage,

Jon Krohn: 00:09:27 Right? Yeah. So Jensen, of course, Jensen Wong, the CEO, Nvidia, probably all of our listeners already know that, but filling in one tiny little blank there and going a little bit more into that deep seek paper and deep seek R one release, they treated communication bandwidth. You were implying it there, but communication bandwidth as this scarce resource for a developer today, an AI engineer today who feels that they're compute poor, but maybe they actually aren't, maybe they're bandwidth poor. How can they fix that? How can they optimize that on their GPU instead of just buying more GPU hours?

Chris Fregly: 00:10:03 Yeah. The most critical metric or characteristic of a GPU system these days is the memory bandwidth. So a lot of times people focus on flops and tariff flops and tensor cores and quantization. Without getting into too much detail there, you really want to look at from generation to generation, pay attention to the memory bandwidth. How fast can I get the weights of these 500 billion parameter models, 1 trillion parameter models? I need to load those into the GPU into those registers, into those caches. And that's 100% dictated by the memory bandwidth. And so one thing I always encourage teams that I work with, customers that I advise, companies I advise is nail that layer down. You have to do tricks. You have to overlap and keep the compute busy while you're loading data in. I mean, these are relatively well-known concepts these days. It's just taking it from concept down to the hardware level. There's a lot of layers in between that are not well documented, and that's what my book focuses on.

Jon Krohn: 00:11:33 Really Cool. I feel like to probably make the most of performance engineering for AI systems, you'd probably would, I mean, it would be tough in a podcast episode like this in an hour to convey to people all the things that they need to know or even the key things that they need to know. Because I imagine a lot of it is specific to the circumstances that they're in. Would you say that's correct?

Chris Fregly: 00:11:58 Right. At the end of the day, everything is workload dependent. You're typically fixed in the hardware that you can get, the budget that you have. This was, and in particular, deep seek didn't even have the same GPUs that the Western labs have. And so trying to, and I'm just looking at the book here. Yeah, after you write a thousand pages, you can't even really remember where anything is. So the secret to this book, by the way, real quick, is that this is actually three different books and multiple times I had. So yeah, I said there was only going to be one or two chapters on cuda. Now there's I think eight chapters on cuda, and the reason is I couldn't find any single resource. There's a lot of Cuda books out there, but they're written by people for specific use cases like fluid dynamics and things that just aren't really relevant to the LLM world.

	00:13:02	There's certainly overlap in the underlying concepts, but being able to tie, here's a model, here's a deep seek model, or here's G-P-T-O-S-S, which is the open source version out of OpenAI, how do I map that in a mechanical sympathetic manner? So this is one term that I've loved for years and years since my old Java Day is trying to optimize JVMs and things, is this concept of mechanical sympathy. This also ties into a piece of advice that I always give to my friends and specifically and specifically my friend's children, is if you're in a CS program right now and thinking that your skills are not going to be relevant, really dive deep into the hardware and try to understand your surroundings. Now the hardware is going to change, of course by the time they graduate and stuff, but at least don't stop at the software layer.
	00:14:03	Think of this hardware, software, co-design. The algorithms haven't changed that much. There's been tweaks over the last few years, different layer normalizer and different parts of the transformer. But what is sure to change is the hardware's going to get faster, the memory bandwidth's going to change, and you really need to be able to think in terms of that whole stack. And that's why this book exploded into a thousand pages. It was because every layer that I would uncover, I'm like, oh man, this is going to be way better if I can explain how all this works together.
Jon Krohn:	00:14:43	So one of the sub books in your a thousand page book, AI Systems Performance engineering is cuda. What are the other two?
Chris Fregly:	00:14:51	So think of it. So there's I think three or four chapters on PyTorch. So think of it like the three parts of the co-design are the hardware, the software, and then the algorithms. And so really the first few chapters kind of set up the hardware, set up the operating system, because all these things matter. One small change at the operating

		system can make or break and cost you millions of dollars. So I've got a whole chapter dedicated to that. And so yeah, basically all of those different layers. So hardware, software on the software side, there's basically a whole book that's in here, the last I think six or seven chapters that are on inference. I do cover training. However, I decided to really, really dive deeper into inference because of where everything's at with the current, I'm trying to keep this book relevant for the next couple of years. And so it's really, it's inference hardware
Jon Krohn:	00:16:09	And inference is where you can correct me if I'm wrong on this, but I believe about when you think about how much the workloads that A given GPU is handling in its lifetime, 99% of that is going to be handling inference, not training.
Chris Fregly:	00:16:21	Absolutely, yeah. Even for post-training, the reinforcement learning, a huge portion of that is actually doing the inference part of it and then just a little bit of weight adjustments and Absolutely.
Jon Krohn:	00:16:38	Yeah. And so this co-op optimization that you've been talking about of hardware, software, and algorithms, those three pieces together, it's a concept that you refer to in your book as mechanical sympathy. Yeah. So I dunno if there's anything more to say about that term.
Chris Fregly:	00:16:56	Yeah, it's a term. Martin Thompson, a famous computer scientist software engineer many years ago, described it. He was making this analogy to Jackie Stewart, who is a British Formula one champion, and Jackie Stewart was famous for not just being a driver, but also able to construct and build the cars. And that was really something very unique to things back then. And yeah, I would say as even today, I don't think a lot of drivers are really capable of turning the wrench and making these adjustments. They can give the feedback, which is what

the constructors need to make changes throughout race and things, but they're not necessarily the ones that are able, and it's a huge advantage when you're able, so a really big, big sort of emphasis. And this comes through with the GitHub repo for this book, by the way, which has about 700 examples and I think is about to reach maybe 800 by this weekend.

00:18:11 I'm constantly updating the GitHub repo, which is documented in the book, but something I very much emphasize that I don't see teams doing is paying attention to the GPU metrics. And this is more than just temperature, which can certainly affect things more so than CPUs, but look at the low level counters. It's something Nvidia has tools, but they're very cryptic. They're command line tools. They've got a ui, but it's janky and it's outdated. And I know that they're working on something better, but actually one of the startups that I advise called Xtra, they've sort of cracked this and they're sort of like, if you're familiar with AppDynamics, they are sort the runtime, the kind of modern version of AppDynamics where it can look at the CPU, the GPU, it can look at PyTorch, give you exact stack traces. This is exactly what I was about to build by the way, after writing this book, was realizing because part of me writing a book is figuring out what are the boundaries of this space and can I find people that can fill in these things? And so it's a very untapped market and being able to in real time, make adjustments, turn the wrench. And so yeah, profiling down to that level is extremely important. And not just stopping at the Pie Torch Profiler, which is what all the PyTorch community, it's so funny because it's just one or two more commands away, but the PyTorch folks seem to just stop at the PyTorch profiler, but that is not enough to really, really dive in and make these changes.

Jon Krohn: 00:19:57 To give our listeners who aren't aware, who maybe aren't hands-on practitioners of ai, PyTorch is by far the most popular library today for putting the pieces together of an AI system. And Chris, what's the PyTorch profiler? What do you see in that and why isn't it enough?

Chris Fregly: 00:20:16 Yeah, PyTorch Profiler, the reason the PyTorch community likes it is because it's Python friendly. It's using terms that are familiar to PyTorch users. It's showing you where time is being spent in your training job or in your inference or basically any PyTorch job that you're running code base that you're running. But it really only gives you a few different aspects of, it's the tip of the iceberg when it comes to trying to actually really understand the system and really achieve mechanical understanding of the whole system. And it'll show you sort of high level that the CPUs being used here and that the GPU now comes in, and there could be better overlap here where you could be moving data at the same time, but there's probably 50 to 60 more critical or just as critical metrics that are needed to really move things forward. And so that's why I haven't really seen a whole lot of progress in terms of performance or that type of progress is really limited to the Frontier Labs or folks like Deep Seek that are taking the time to really understand what's beneath the covers

Jon Krohn: 00:21:39 Kind of, you mentioned there kind of a dozen or so metrics that are critical that are difficult to see. What are the key ones that we might not be aware of and how could we possibly see them?

Chris Fregly: 00:21:50 Sure, yeah. The GPU does expose a lot of counters and there's really the two or three main tools are at the kernel level. So Kernel is, it's the program that the GPU actually runs, right? And when you program to A GPU, you have to realize that part of it's happening on the CPU to kind of prepare the data. Then the data has to ship over to the GPU. And now the GPU has a whole bunch of constructs and hardware optimizations that have been built up for the last 10 years specific to ai. So there's things like the streaming multiprocessor, think of it like a core or a set of cores that you would tune for parallelism on the CPU. Well, the GPU has that same thing called streaming.

00:22:53 The amount of time being spent on those streaming multiprocessors a percentage that's called the occupancy, and it's basically all about the utilization of the hardware. But the other really key critical thing is that there's separate pipelines. There's instruction pipelines that power all of these different portions of the GPU. So the GPU has specialized chips for soft max, which is used super heavily in transformer world. And so what we've seen is since transformers 20 16, 20 17, once Nvidia found out that this is what researchers are using, they started building in these specialized function units there, which short, so that acronym is then SFU for specialized functioning unit to handle those transformer specific operations. And then of course there's also tensor cores, which are specialized cores that can do floating 0.8, floating point 16. So specialized operations that of course are common in transformers. Those all have separate pipelines trying to move memory in and out. That's another pipeline. And so you really have to understand how can I best parallelize all of these things and make sure that they're all fully utilized all at once.

Jon Krohn: 00:24:29 Really cool. I don't have any experience with that level of depth. And in fact, that leads me to my next question because most data scientists, AI engineers, we stay at the Python level. At what point does an AI system scale make it mandatory to go down the stack into Cuda C plus Triton or I don't know, SFU

Chris Fregly: 00:24:52 For sure. Yeah, and one of the things I've been really trying to do with my GitHub repo is I've got MCP tools that if you're inside of cursor inside of cloud code, you can use these CP, so right, but the issue is that you have to be directly on Blackwell B 200 machine to run these tools. And so you have to shell in, so my particular development flow is I shell right in with cursor or with VS code with the different plugins, code X and cloud code I shell in directly to the B 200 that I'm actually going to run this on, and I start chatting basically with the system and say, okay, tell me about what's your memory bandwidth, run this particular PyTorch code or this inference job and tell me where's time being spent and then tell me which lines of code. And so I'm able to actually chat with my system.

00:26:01 And so that's super powerful. And it can also detect things like when the GPU is getting warm, and so I could lock the GPU clocks, for example, to get a consistent benchmark. One thing about the GPU ecosystem is it's these things are purposely running way hotter than a CPU would, right? If you recall maybe being a kid, you were building your own machine or you have friends that were building a machine and they would always try to overclock either the CPU or the GPU, so the NVIDIA folks have decided just overclock it themselves. But if you're trying to run a benchmark or trying to see if a change that you made had an impact, you have to basically cap that overclocking because if you go up too hot, it's going to throttle and then your change will appear to not have impact because it basically got too hot and then stopped processing as quickly. So yeah, it's still very hard even to go from my book, which has all of this information in one place to a day-to-day practical thing, which is where the GitHub repo, which is where these different startups are really focusing on.

Jon Krohn: 00:27:28 And so earlier in this episode I said how it would be very difficult in 60 minutes and an hour long podcast episode to convey all the things that somebody needs to know in order to optimize their AI system. Your book actually ends with a 175 item optimization checklist, which kind of goes to prove my point that if we took a 60 minute episode, it would, we'd be able to spend 20 seconds on each of those items in the checklist, which obviously is not going to convey any meaningful information. But if a listener had to take a handful of those, maybe three of those as the kinds of optimizations that would give them the best return on their time investing their time in that, is it easy to kind of say that these handful would be, are the kinds of key things that they need to be doing?

Chris Fregly:	00:28:20	I'll give you a little tip, which is if you have the PDF of the book, which obviously I do, I give that to cursor, I give that to Claude code. And when I'm analyzing a system, because these models don't have all of these things at their fingertips, there's a lack of training data, for example. So one thing I do is I actually put the book in context and it's a thousand pages. So you need a big model that has a big context window and ask it what are the top 10 things I should be looking at? And it'll start to run its checks and everything, assuming that you have the right tools installed, which is the Nvidia insights and the Nvidia compute. So that's kind of longer term what you would do, which is try to give enough context. Now, I think this will change. I know that the labs, I know firsthand that the labs are training more and more on hardware specific for A MD because they themselves have this interest in their models getting better at optimizing kernels, but to be very systematic about these things. I'm certainly going to talk through a few of these, but make sure there's some sort of before and after profiling best practices thing, things like locking the GPU clocks and making sure it doesn't hit peak, but then there's really having a methodical way of going through.
	00:30:13	So in my GitHub repo for example, I've got this harness that will do things consistently before and then after, and if there's any inconsistencies, it'll fail the comparison and say, Hey, this is not apples to apple. So it's extremely

important that you have a very, very methodical way of comparing changes, but things to keep an eye on. Well, yeah, I'll start small and then kind of go bigger on a single GPU. Obviously you really want to understand if you're on Blackwell has extremely different characteristics, then Hopper, then we'll have in the future Vera or I think it's Ruben is the next generation and then Fineman is the next generation after that. But separate from the GPU, there's a lot of things that can go wrong at the actual operating system level and with the GPU driver for example, there's the famous Cuda toolkit and then the NVIDIA driver.

00:31:20 And yes, anyone that's tried to enter this world has spent I'm sure countless hours trying to get these things synced up and the versions all set up properly. Once you get past that, and I do want to emphasize that you should always be on the latest. I don't always upgrade for the sake of upgrading in my life just sort of generally, but I've noticed in the GPU world, it's extremely critical to stay up to date. That does cause problems because sometimes the PyTorch doesn't keep up with the latest CUDA 13.1, but you really, really have to be aware of the versions in this world. But yeah, I would keep an eye on memory fragmentation, extremely important. You want to disable swap for example. These are things that if you just get a basic Linux system, these things are all turned on because they're optimized for CPUs and just sort of general purpose workloads and a lot of different tuning that goes in.

00:32:26 This was actually one of my favorite things about the sort of SageMaker right ecosystem is that there was a lot of work put in. Yeah, so SageMaker is the Amazon or it's a service within AWS that offers you GPUs and specifically there's a service called SageMaker Hyper Pod, which is pretty bare bones in a good way. It's got Kubernetes installed, it's got all of these things tuned. You've got a lot of people focusing on this. There's a lot of customers that are tuning that are helping Amazon with the right configuration for all of these things. You almost always want to occupy the entire GPU, so you don't want to share like you would in A CPU lands where you can split it up and virtualize it. You want to get as close to bare metal as possible. Containers have gotten a lot better throughout the years with GPUs, so we're seeing a lot more folks use Kubernetes and Docker for these kinds of things.

00:33:34 But I would almost encourage to stick with bare metal IO optimization I think is the number one killer here. This is the lowest hanging fruit. Use the fastest discs that you can get. Use local disc. Don't try to use any network file system. It's convenient to use a network file system, but it will almost always cause you problems. I had said earlier that Jensen, the CEO of Nvidia has been focusing a lot on storage recently, probably not as much as I think he should be going forward, but he does have a lot of partners that are working with, and there's something called GPU direct storage, which can load data from DISC right into the GPU without going through CPUs, without creating copy buffers and those kinds of things. So that would impact the sort of data pipeline. And when you scale of course to multiple nodes, multiple GPUs, which is these days just a given, you really can't do anything that complex on a single GPU anymore.

00:34:45 So then you start to optimize the network and then you have to make sure that you're using direct GPU to GPU communication and use things like NV link, the NVIDIA interconnect that's been optimized for the, so one piece of advice is when you're looking around for a GPU cloud provider, I guess these days, people call them neo clouds, like core weave and stuff, try to find one that has bought into the entire reference architecture like buy Nvidia. There's a group of, I guess cloud incumbents that have their own networking, that have their own stuff that's been built up throughout the years and they are patching and bypassing the sort of natural way that the NVIDIA GPUs were designed. And that's actually in my opinion, the highest value proposition for these neo clouds is they're not tied to all of those sort of CPU legacy ways of doing things.

00:35:57 They have bought into the Nvidia ecosystem. In some senses they're favored by Nvidia, I would posit, which seems natural because you're going to favor customers that buy a lot more from you. Because keep in mind, one of the most pivotal aspects or the most pivotal acquisition like buy Nvidia was this company Mellanox. So they bought this company Mellanox back in I think 2019. I think it closed in maybe 2020, their purely networking. And so now Nvidia, since 2019 has gone from just a GPU company to an AI systems company. And so if you buy into that whole thing, there's a lot of efficiencies that happen even in the network hardware can actually do calculations with these mellanox routers that like Nvidia purchased.

Jon Krohn: 00:36:55 Fantastic. Thank you for all of those tips giving us, distilling for us the most valuable of the 175 items that have provided in your book. I think that was

Chris Fregly: 00:37:06 Four of them,

Jon Krohn: 00:37:07 And that's great. I mean, what a nice thing to be able to do in this episode. And yeah, of course, check out Chris's book, AI Systems Performance Engineering to get the full list of 175, a full thousand pages on how you could be optimizing your systems. Let's shift gears now a little bit, Chris, to talking about inference specifically. We talked earlier in this episode about how inference is most of what's happening at scales with AI models. Training is an important part, but inference is most of what we're doing

		with GPUs these days. And so you talk in the book about multi-node inference and disaggregated architectures. What the heck are those?
Chris Fregly:	00:37:49	Yeah, welcome to another chapter that was supposed to be a single chapter or topic that was supposed to be a single chapter and
Jon Krohn:	00:37:55	Up. Yeah, so this is one of the three books, one the three sub books
Chris Fregly:	00:37:58	In the book, exactly. Inference. There's a lot going on in inference that folks probably on the surface don't realize. So stated simply with inference, we're doing just a forward pass where we've got the weights, the weights, which are the parameters in this. Let's say it's a trillion parameter model, we just need to compute the answer, boom, just go forward. So on paper, this seems extremely simple in reality, transformer based inference, there's a lot of places for things to slow down. And so there's a lot of caching that needs to happen so that we're not recomputing things on every, so think of when you go to chat PT or to Claude, you type in a query, it is going to look at all of the context that you've passed in. It's going to compare everything to everything else that's in there. It's going to find things that are relevant, and it would have to do that the same way every single time on every single query, for example. And yes, even within the exact same query, it's doing that many times generating the next token and the next token. So there's something called a KV cache. Funny enough, I was going to put KV cache on my license plate. I just shipped my car from another place and somebody in California has already taken the word KV cash on their license plate. So I'm like, that's how popular KV cash is.
Jon Krohn:	00:39:37	IYou can probably in any other state in the US still get that

Chris Fregly:	00:39:40	Totally right. So someone beat me to it, but the, so think of it just like any sort of cache system where you're computing, but on a single query, these caches can be humongous. I mean, especially with people that are uploading documents that are trying to scan things, there's a lot of computations happening. So now you've got a case, so there's building the cache, which is step one, and then there's actually generating the next token, which is step two and starting to actually generate the response. So those are called prefill is the first step where it's generating all of these computations that's extremely, extremely compute heavy. And so that's where you potentially want a node or a set of nodes in your cluster that are configured differently. So memory bandwidth, it's still important, but it's not as big of a deal as the second step, which is actually generating those tokens because that second step has to move all the weights back and forth from your GPU ram, sometimes CPU RAM based on how big the GPU REM is sometimes on disc.
	00:41:10	So these are very different configurations, and so you'll often hear people talk about disaggregated pd, which is disaggregated, prefilled decode. I would guarantee you, if you are going for an interview in 2026 and need to study something for your interview, it is going to be about this prefilled decode. It's going to be about KV cash. So if you're going to talk to any of these big labs in any sort of engineering capacity study, I can't remember what chapter it is in my book. It was supposed to be a half a chapter actually, and it ended up being I think three or four. So there's trying to understand it and then there's trying to optimize it.
Jon Krohn:	00:42:03	Prefu decode, disaggregation and KV cash. There you go. There are your interview tips of the episode. Speaking of tips and ways that people can be getting ahead more quickly, you mentioned to me prior to us recording that you've been loving using AI coding assistance, and so

		tools like Cursor Cloud, code Codex, how has your workflow changed as a result of these kinds of tools? How does that impact in particular AI systems performance optimization and yeah, what do our listeners need to know about where this is going?
Chris Fregly:	00:42:42	For sure. I tweeted about this a couple of weeks ago. If you're manually writing code in this year 2026, you are way behind. And I would not have said that if I didn't spend all of last year watching this sort of evolution. I started off in my legacy ways where I was writing every line of code and I vowed for 2026 to only use these coding assistance and to see how far I could get. Now, I've done a couple of projects for some friends and for some portfolio companies just to make sure that, so the short of it is you can fire off about 10 to 15 different things, like different aspects of either features that you're trying to build or bugs that you're trying to find. I'm personally using these tools right now to do a lot of optimizations and to look at different aspects.
	00:43:48	And so I'll say, take a look at the occupancy percentage for this particular kernel that I'm trying to optimize. At the same time, I'm having another GPU that has the same code that's analyzing a different part of it. And so think of a four GPU system and you can run separate experiments, each one running on a different GPU or even four separate GPUs, like separate nodes that have their own memory and stuff. But yeah, my personal workflow, I'm going to be a little controversial and say I prefer Codex. Yeah, I prefer the open ai one thing, and again, very, very workload specific. I would say. I think if I was doing UI stuff, I would probably maybe prefer Claude right now. Claude's great with the ui. It's kind of my little dirty secret that I use Codex. And so while all the other developers are flooding the Anthropic GPUs and the inference stacks, I've got my little group of people that just use Codex and I still have really good performance

		until people realize that Codex is actually better for other stuff.
	00:45:03	But for right now, I'm enjoying good performance and fair amount of token. But the one thing I would recommend for AI systems performance would be on the machine. Don't try to do this stuff on your MacBook and hope that it works. Or even on a personal like Nvidia, GPU, like an RTX, yeah, I don't know, 50. I've literally never used any of the personal GPUs because to me, the profile is so different. The hardware is so different, the memory bandwidth completely different. So if you are working on an AI project and you are ultimately going to deploy on a Blackwell or on a hopper, you have to be on that machine during development. Don't try to take any shortcuts. I recently tried the DGX Spark, which is the little mini one that got a lot of buzz the end of last year. There's so many things disabled on that that none of my benchmarking was even working. It's just completely missing a lot of core hardware components and then software components on top. The driver isn't the same. It's a lot of different things.
Jon Krohn:	00:46:16	Nice. Really appreciate your insights on what you're doing with the coding assistance today. When you are working on those, Chris, do you still review all of the code before it goes to production?
Chris Fregly:	00:46:27	I did. Yeah. I was once you, or once the question that you're
Jon Krohn:	00:46:34	Asking, in
Chris Fregly:	00:46:35	Fact, I was just working with with someone this weekend and trying to get them a little bit up to speed. It's a good friend of mine and I'm like, look, you're doing things, you're going to be outdated within by St. Patty's Day. That was the joke specifically. And I was watching their

workflow and know these assistants, they like to write little snippets of Python code or Bash scripts to get stuff done, and this person was literally reading every single line of the Bash script. It's happening during the thinking process. And they would go up and they would expand it and they would be looking and they'd try to find the script on disc, and I'm like, dude, you have to let go.

00:47:28 I did, and it's exhausting and I've learned to step back and there's temporary scripts that these things write to write the final code, and you could do that and you could be very disciplined and review all the lines of code. You could write tests, you can have the LLM actually write the tests as well too. I've actually even sort of let go of unit tests, and this is very controversial for all of my test driven friends and folks that are listening. I focus on the evals, so I clearly set up and provide examples that I know are correct. Those are sort of the collaboration set. And then I use the LLM to actually judge the quality. I have it constantly running, so I've got evals that are always running in the background in a separate tab while I'm building the software. So short answer is no.

00:48:38 I stopped doing that and it really slows things down and it's hard for people to do, but I'm shipping code a lot faster with kernel optimization. You have to be very careful. These models want to hack the reward or it's called reward hacking and just get the fastest thing possible. If that means zeroing out everything to make the computations a lot faster, it's going to do it. One thing I spent a lot of time on with my GitHub repo is these correctness checks and correctness verifications. These pop up, yeah, every single month someone released some small little startup releases, something that they have achieved 50 x speed up, and the first thing I do is take that code and I put it into my harness and boom, I see that they're not using Cuda streams properly or that they

		have that. It's not their fault, it's just that they've given the LLM too much freedom.
	00:49:46	And so we have to get better about what it means to review. And I don't think even going through all the code that you would be able to catch these things because I assume that they went through and looked at the code, but not unless you're actually running the code on the machine and you've got all of the profiling set up where it can show you how many streams are being used and all the different aspects. If you don't have correctness checks, if that's in the form of evals for your end user application or performance metrics and real correctness checks within your performance harness, then you're not doing it the right way.
Jon Krohn:	00:50:27	Yeah, really cool. I guess I need to stop looking at the code. Yeah.
Chris Fregly:	00:50:31	Do you still do that?
Jon Krohn:	00:50:32	Yeah. Living in the stone age. So my last semi-technical question for you before we start wrapping up, there's so many interesting things about you. We actually, we've really just scratched the surface here. We focused on your latest book, which again, if people like this episode, you've got to get Chris's latest book, AI Systems Performance Engineering. But in addition to that, you've done tons of other interesting things. So your previous two books were both both on, they were on AWS books. It was Data Science on a WSI think that might've been with Ante bart. That was the one that was with her. And I think you ended up at AWS because you were the chief product officer at a company called Pipeline AI that was acquired. And you've also, you've worked in principal engineering roles at Databricks and Netflix, including getting an Emmy at Netflix for the work that you've done.

	00:51:30	But the thing that I want to talk about, the only thing that we have time to talk about now is your role as an investor. So you've been an investor or advisor in companies like xai, which is now at the time of recording was just acquired by SpaceX. You invested in Groc, GROQ, which makes LPs language processing units, and that firm was acquired by Nvidia. And so yeah, I'd love to understand your thinking on what's going on with hardware. Do you believe, for example, that the future of AI performance lies in specialized LPU hardware like rock builds or in better software orchestration on top of general purpose GPUs? And you can also answer this question more broadly with just generally where you think hardwere is going and where people should be putting their money.
Chris Fregly:	00:52:25	I'm in a fortunate position where I do have extra cash and I can take some chances. These are all secondary market shares. They're not public companies or some are about to IPO and things on the investor side of it. I look for, this is very trite these days, but you have to look at the founders. Elon, he's an absolute nut job of course, and he's very controversial and my sister hates Teslas because of him and all that kind of stuff. So I can't even talk about some of these investments when I'm home for Christmas and things, but you have to look at their ability to fundraise Right now. The secondary market is where all the action is, and so if you're still investing in Apple and things, you're behind the times. This is not where the money's currently being made, especially because there's been such a dry spell with these IPOs. So hopefully that changes. Yeah, I'd say my best investments so far have been Databricks. Of course, I was an early employee there back in the day, and I know those founders very well. You have to look at folks that can raise money, that can convince people that they're on the right path that have their stuff together.

00:53:59 Where are things going? There's so many different types of chips coming out. In fact, I'm about to go to lunch with some folks at Etched, which is another little player and similar to Grock with a Q. It's good to keep an eye on these folks to see sort of generally where things are moving. I mentioned before that the biggest bottleneck is this memory bandwidth. This is something, this is the main reason why the NVIDIA folks purchased GR with a Q or did their little weird acquisition licensing deal is the LPU has all of that memory on the same chip right next to the actual accelerator itself. And so that then decrease. And that's really good for inference because like I said, when you're doing inference, there's that decode step that has to move a lot of weights around. That's very, very memory bandwidth bound. And so it's a really great compliment to the NVIDIA chip, which suffers from this memory bandwidth limitation.

00:55:11 And it's just a fundamental architecture design that dates back to their graphics days. So instead of building a cycle of five or six years out where they have the chip, they ended up, and also with XAI, it's interesting because I passed up on SpaceX a few years ago. I didn't see the leap that Elon is currently talking to. I didn't have enough vision for that. I saw that I liked XAI in that crock with a K, which I actually personally don't really use that much. I should because an investor, but I liked, liked the work ethic of those guys. I actually met with them quite a bit last year while I was writing the book. They're big backers of SG Lang, which is like VLLM, which is an inference engine, but it's SG Lang. It's another project that kind of came out of Berkeley around the same time VLLM that does things a little bit differently. So we had some conversations of those conversations actually are part of the book now. What's the specific question again? I talked my way out of that.

Jon Krohn:	00:56:31	Yeah, I had a question around, it's basically, where's this going?
	00:56:37	Where are the opportunities for investing? And of course, we've also got to give the disclaimer that we are not investment advisors, but I think it is just something that people are interesting. You've got a really interesting being in the Bay Area with the kinds of companies you've invested in. You have an inside baseball perspective on where things could be going. You've already given us some of the tips around things like your advice is that the secondary market is where there's a lot more money to be made than in the public markets, your apples, for example. But yeah, and you also kind of already answered the question about whether the future lies in specialized LPU hardware or more general purpose GPUs. So I think you've covered that, Chris,
Chris Fregly:	00:57:17	The folks over at together ai, the person, I think he's the chief scientist, one of the co-founders of this company called Together AI, is a big fan of general purpose GPUs in that it gives you more flexibility to support different types of algorithms that can, not all the world is going to be transformers. I mean, this is going to change. It just has to change. We need a better, and all of these chips that are specialized for transformers may not be the best chips going forward, which was quite honestly a tiny bit of concern when I made the Crock with a Q investment is, oh crap, if something changes and another type of model starts to dominate another type of algorithm, where is that going to leave folks like Rock with a Q? So fortunately, I've got shares on both sides of it. I've got public Nvidia and I've got private things as well too, so you have to kind of have a balance there. But yeah, and I'm the worst investor in the world. Sold Netflix when it was low and I bought it when it was high and then it went low again and I lost a bunch of money and things like that. So don't take advice from me.

Jon Krohn:	00:58:42	Great
Chris Fregly:	00:58:43	To be getting lucky.
Jon Krohn:	00:58:45	Buy high sell low is the takeaway advice from Chris, frankly today on investing. Great.
Chris Fregly:	00:58:51	Yeah, certainly look at the addressable market. If you're familiar with, Sequoia has a very famous pitch deck, and I'm very familiar with this pitch deck because I spent many years fundraising for my startups. And when I go into even a job, if I start talking to folks about potentially joining even as an advisor or any type of role, if I decide to go back into engineering full time or whatnot, you want to stick, there's a famous, I think it's 12 or 13 slides, and it seems so silly to only have 13 slides to make or break millions of dollars, but think of try to find that slide deck. It's out on,
Jon Krohn:	00:59:41	And it's the subject of countless blog posts out there. Having the experience of building these decks myself, you're not going to have a hard time finding guides to take your dozen page deck. And I think the key takeaway, maybe this is exactly where you're going. I don't mean to take the words out of your mouth, but if you can't explain in that short number of slides why this is a compelling investment, it's probably not a good investment
Chris Fregly:	01:00:05	And value proposition. To me, that is the most important slide. I mean, yeah, obviously team and like assuming team and all that stuff's there, but if you can't clearly explain why now and why am I the right fit for this and what's the real value, that's always to me, the hardest slide is like, yeah, how do I quantify this? But yeah, I mean, always look at the addressable market and also these days it comes down to the business model and the pricing gone are the days of this seat SaaS-based seat pricing or now it's all about value, so value-based pricing,

and they call it outcome based pricing, which is very obvious for things like customer service and stuff. But yeah, trying to think about that. So you really want to look for founders that are in tune with that, and especially repeat founders that might not have traditional, have been part of traditional pricing models, business models. You really got to suss that out.

Jon Krohn: 01:01:15 Great. Thanks for all of those inside baseball tips, Chris on investing. Yeah, so now we'll start wrapping up the episode. You already know you've been warned by Antje Barth that I always ask for a book recommendation at the end and don't let you use your own book, and so I think you have come prepared.

Chris Fregly: 01:01:33 Yeah. One of my favorite books is called Founding Sales, and these days it's a relatively old book. It's from August of 2020, founding sales, the early stage go-to market handbook, and I'll tell you why this is an important book and was so critical to me. So the author goes through his sort of evolution in learning about go to market, GTMI used to laugh when my friends that came out of business school would talk about go to market. It was so far from the things that I would think about on a day-to-day basis, and then I became a founder, and then I was like, oh, crap, I have to figure this out. So there was a point in time, and I still, I'm fortunate that I live somewhat near Stanford, so I take a lot of weekend classes and I don't have any kind of executive degree or anything.

01:02:28 I just have an undergrad from Northwestern, but I can soak up what is valuable to me. So really thinking through what does go to market mean specifically in a SaaS environment, and not all the things will be relevant to LLMs. Of course, this is an older book, but it really gets you to see the pain points that he had gone through and the different pivots that he made and how his competition really affected his route through this. So if you're, I point this out because I think this audience is mostly technical, and if you ever do think about becoming a founder, yeah, check out founding sales because it's a nice sort of engineering friendly way, logical way to think through how to come up with your real true value proposition for your ideas, and you'll read something and say, oh, that's exactly what I was thinking, and now I see how it's bad, how it's not really going to convince anyone to use this product or to give me money.

Jon Krohn:	01:03:38	Nice. That's a really great tip, and I like how it follows along nicely with actually the arc of this episode as well, where we started off really technical, really heavy in the weeds on AI systems optimization, and then we kind of ended with commercial discussions and a commercial book recommendation so perfectly on theme. Chris, in the end, you didn't know that's where I was going with questions, so pretty cool. Yeah. Then my final question before I let you go is how can people listen to you after this episode? Obviously your book, AI Systems Performance Engineering is the way to go. I'll have a link to buy that book as well as to the GitHub repo for the book. Beyond that, how should people be following your thoughts after this episode?
Chris Fregly:	01:04:20	Yeah. I'm c Reley, C-F-R-E-G-L-Y, pretty much everywhere. I recently discovered. I have a second cousin named Chris Fregly, who does lots of van conversions. He's in tech. He's a Kubernetes guy, and we actually live pretty close to each other, so we've been connecting. He's half my age, which is real fun to get his perspective on the tech industry, but we're both in San Francisco, but yeah, I'm the original. Yeah.
Jon Krohn:	01:04:53	Okay. I see you're saying watch out for that guy, don't follow him. If there's van conversions, you've gone, it's the wrong guy.

Chris Fregly:	01:05:01	Totally. That's the most unique thing is I don't do van conversions. I prefer to live in a house condo. Yeah. So cre on Twitter, cre, GitHub or Yeah, Chris, regularly LinkedIn.
Jon Krohn:	01:05:18	Perfect. Yeah. We'll, of course have links to all of those in the show notes. Chris, such an honor to have you on the show after the kind of decade of being aware of you. You're such a rockstar. Thank you for taking the time. And yeah, next book comes out. You've got an open invitation.
Chris Fregly:	01:05:32	I am swearing off book writing. Yeah. Which I said after the first one, after the second one, but yeah, we'll see. Hey, John. Yeah, man, great chatting.
Jon Krohn:	01:05:44	What a great episode today with Chris Fregly. In it, he covered how deep CR one achieved 10 to 20 X training cost reduction over comparable Western models by co-designing hardware, software, and algorithms together. A recurring topic Throughout the episode, he talked about how memory bandwidth, not flops or tensor chords, is the single most critical metric to pay attention to when evaluating GPU performance from generation to generation. He talked about mechanical sympathy, a term borrowed from formula one. That means understanding the full hardware software stack and Chris argues it's the most valuable skill for anyone studying computer science or related jobs today. He talked about how the PyTorch profiler only shows the tip of the iceberg. There are 50 to 60 additional GPU level metrics involving streaming multiprocessors occupancy, specialized function units and instruction pipelines that are essential for real optimization, and he spilled the beans that he's abandoned, line by line code review in favor of continuous evals and correctness harnesses, and says that if you're still manually writing code in 2026, you're way behind.

01:06:49 As always, you can get all the show notes including the transcript for this episode, the video recording, any materials mentioned on the show, the URLs for Chris Re's social media profiles, as well as my own at superdatascience.com/973. Alright, thanks to everyone on the SuperDataScience podcast team podcast manager, Sonja Brajovic, media editor, Mario Pombo, our partnerships team Natalie Ziajski, our researcher, Serg Masís writer, Dr. Zara Karschay, and our founder Kirill Eremenko. Thanks to them all for producing another super episode for us today for enabling that super team to create this free podcast for you. We're deeply grateful to our sponsors. You can support the show by checking out our sponsors links, which are in the show notes, and if you'd ever like to sponsor the show yourself, you can get the details on how at john cron.com/podcast. Otherwise, share this episode with people who would want to listen to it, review it on your favorite podcasting platform or on YouTube, subscribe. Obviously if you're not a subscriber, but most importantly, just keep on tuning in. I'm so grateful to have you listening and I hope I can continue to make episodes you love for years and years to come. Till next time, keep on rocking it out there and I'm looking forward to enjoying another round of the SuperDataScience Podcast with you very soon.