Audio Transcript

45 00:09:16.950 --> 00:09:26.970
Julian Shun: Great. I see it. So yeah, let me let me introduce you. So hi, everyone. Welcome to the first seminar of the fall semester for the fast code seminar.

46 00:09:27.720 --> 00:09:38.280
Julian Shun: We have an amazing line up of speakers so far, please feel free to check out the summer website for more details. Today I'm very happy to have Franz friend Shetty

47 00:09:38.700 --> 00:09:48.630
Julian Shun: As our first speaker for the false fall semester. So Franz is a full professor at Carnegie Mellon University and the Department of Electrical and Computer Engineering and

48 00:09:48.750 --> 00:09:53.610
Julian Shun: He received his master's degree as well as his PhD degree from the

49 00:09:53.640 --> 00:09:54.150
Julian Shun: Vienna.

50 00:09:54.240 --> 00:10:02.820
Julian Shun: University of Technology and Francis research focuses on automatic performance tuning and program generation for Virgin parallel

51 00:10:03.150 --> 00:10:15.180
Julian Shun: Platforms and algorithm hardware co synthesis. He leads the spiral project where the goal is to enable automatic generation of highly optimized software libraries for important kernels.

52 00:10:15.870 --> 00:10:27.510
Julian Shun: Franz has received many awards for his work, including many best paper awards at high performance computing conferences, as well as the early career fellowship at Carnegie Mellon as

53 00:10:28.140 --> 00:10:38.460
Julian Shun: And the ACM Gordon Bell prize, among many other awards. So today, Franz, it's going to tell us about his work on the spiral project. So I'll turn it over to you for us well.

54 00:10:39.180 --> 00:10:45.150
Franz Franchetti: Thank you so much. Julian I do get to meet people, or do you also get

55 00:10:45.360 --> 00:10:46.260
Julian Shun: I'll take care of it.

56 00:10:46.320 --> 00:10:46.890
All I think

57 00:10:48.660 --> 00:10:58.170
Franz Franchetti: I know. Thank you very much for the invitation. And so I did take the liberty to actually update the talk titles that we may have to fix that up.

58 00:10:59.190 --> 00:11:12.600
Franz Franchetti: And so yeah, let me get going. So my talk to the basically the current status of the spiral project which has been going on for about 20 years and in the latest

59 00:11:12.750 --> 00:11:15.210
Franz Franchetti: Incarnation, we think of spiral as a

60 00:11:16.050 --> 00:11:20.640
Franz Franchetti: AI tool for high performance code generation or performance engineering

61 00:11:22.050 --> 00:11:27.360
Franz Franchetti: So the work that I'm going to talk about has been sponsored by a variety of

62 00:11:28.680 --> 00:11:41.310
Franz Franchetti: Funding agencies DARPA, do we when our NSF and the number of companies, including into mercury and NVIDIA and it's joint work with the entire spiral team, of course, over the last 20 years so

63 00:11:42.060 --> 00:11:49.260
Franz Franchetti: too numerous to list right now on the back page on spiral of net you can find everybody who has ever contributed to spiral.

64 00:11:50.610 --> 00:12:02.610
Franz Franchetti: So the basic assumption or the basic observation behind our approach is that if you look at a different time scales between mathematics and

65 00:12:03.720 --> 00:12:15.810
Franz Franchetti: Computer hardware and computer science and programming you see vastly different time scales, you see that algorithms and mathematics has about

66 00:12:16.350 --> 00:12:30.090
Franz Franchetti: 2500 years or more of history. And so the earliest formulas that I'm going to use in today's talk. Go back to 300 BC. And one of the things that I always find very interesting is that

67 00:12:31.950 --> 00:12:39.540

Franz Franchetti: For example, Indian mathematician, said the value of pie and for square roots to the same accuracy that we're using today.

68 00:12:39.930 --> 00:12:58.080
Franz Franchetti: In GPUs about 800 BC. So that gives you like the age of it. Then, of course, they're all developmental algorithms like Gordon elimination or the physics equations like equations of motion until they all have a couple of hundred years in them and the Fast Fourier Transform the

69 00:12:59.310 --> 00:13:02.940
Franz Franchetti: Computational kernel that spiral came from originally

70 00:13:04.440 --> 00:13:06.240
Franz Franchetti: Basically is

71 00:13:07.260 --> 00:13:17.910
Franz Franchetti: Started out at No five by gloss and eventually came into a form to make it possible to automate the performance optimization and go generation for it and that

72 00:13:18.540 --> 00:13:27.990
Franz Franchetti: In MIT gave rise to the FW project to foster through trends from the west and hit Carnegie Mellon. We did this spiral project I joined

73 00:13:30.240 --> 00:13:39.210
Franz Franchetti: I think around the 2000s, as a visiting student and have been with it ever since. So again, the time scale of algorithms and mathematics.

74 00:13:40.650 --> 00:13:51.900
Franz Franchetti: Centuries millennia. If you compare that to computing platforms and I do a lot of work with DARPA there for the very first

75 00:13:52.380 --> 00:14:02.580
Franz Franchetti: Airplane on my slide you see that a plane like the F 16 the airframe most developed to physically first fly around 1974

76 00:14:03.120 --> 00:14:18.180
Franz Franchetti: Which happens to be approximately my vintage. So an airplane is physically flying forever. Also, but in the same time, when you look at what's happened to the computer technology, you see that in 72 to into 808 started out

77 00:14:19.320 --> 00:14:20.910
Franz Franchetti: As an intelligent terminal.

78 00:14:22.050 --> 00:14:31.560

Franz Franchetti: And by now in 2020 we have 28 cores and 4.3 gigahertz and 16 racing the per core. So if roughly

79 00:14:32.610 --> 00:14:44.100
Franz Franchetti: 500 times parallelism, although their current chips are still assembly compatible down to 1972 and actually binary compatible time down to 1989

80 00:14:45.150 --> 00:14:57.570
Franz Franchetti: So this is a pretty amazing engineering feat. But the thing is, we got about 10 to the eight x compounded performance gain over the last half century or so. So let me see here is

81 00:14:58.380 --> 00:15:13.710
Franz Franchetti: The enormous timescales into enormous performance scales that we have to reverse this computer engineers. And so the question is, how do we get to the point to write software once and running across all these different platforms and all the platforms in the future to come

82 00:15:15.300 --> 00:15:18.570
Franz Franchetti: And so this is the, the big question here. And so

83 00:15:19.680 --> 00:15:30.870
Franz Franchetti: If you're think programming languages to the rescue. Look at the time scale of programming languages, would you see is that Fortran started in 19 1953 and

84 00:15:31.650 --> 00:15:53.850
Franz Franchetti: I would say, Julia is one of the latest additions in popular languages in 2012 and you see how old C c++ C has about the same vintages the F 16 t plus classes, maybe 10 years younger open NP kuda all these things have been around for ages. And so if you want to develop a system that

85 00:15:55.710 --> 00:16:03.690
Franz Franchetti: Is basically being adopted, you're really face to the problem you're facing the problem that the average programmer.

86 00:16:04.950 --> 00:16:19.680
Franz Franchetti: Is not very happy to adopt new stuff. So that's not exactly true. Of course, new frameworks, new ideas will be adopted, but really from it long term maintain ability code that really survived a very, very long time.

87 00:16:20.880 --> 00:16:22.860
Franz Franchetti: Nothing really nothing much changes.

88 00:16:25.200 --> 00:16:33.450

Franz Franchetti: So another thing to set up the talk that I like to think about is always, what does a million dollar buy

89 00:16:34.170 --> 00:16:48.630
Franz Franchetti: And so this is the slide of the talk that every talk. I basically have to update. So I did update it just an hour ago. And so what you see is that for approximately a million dollar in second half of 2020

90 00:16:49.230 --> 00:16:51.960
Franz Franchetti: You can get something on the order of

91 00:16:53.160 --> 00:17:12.930
Franz Franchetti: Four times 24 core shared memory machine was about six terabyte memory. You can get maybe 200 terabytes of flash drive, you can get maybe a box full of FPGA. And you can get about 1.6 petabytes spinning disk. And one of the latest, greatest and video de GX

92 00:17:14.490 --> 00:17:24.510
Franz Franchetti: Machines all that together fits in approximately half a wreck and approximately 10 kilowatts, and about approximately a million dollars. So I think

93 00:17:24.900 --> 00:17:35.310
Franz Franchetti: Engineers. I always like to like have a understanding of where technology is that every point. And so once a year like to go to supercomputing and walk through the halls to see

94 00:17:35.730 --> 00:17:43.830
Franz Franchetti: What is happening because if we try to optimize across different machines, you better know what the hardware vendors are doing it on every point in time.

95 00:17:45.930 --> 00:17:56.940
Franz Franchetti: So now, given that setup. The question is, what is the spiral project or the spiral effort. So it is really, it has grown into an effort to

96 00:17:58.050 --> 00:18:10.170
Franz Franchetti: catalog. The catalog machine readable catalog of algorithms hardware and program transformations to enable Automatic Programming for villa understood kernels.

97 00:18:10.500 --> 00:18:16.830
Franz Franchetti: Really across these timescales in these performance skills. So traditionally, what would happen is that you have a library.

98 00:18:17.520 --> 00:18:23.190
Franz Franchetti: And library here being actually the building with the books and you go in, you grab the particular

99 00:18:23.670 --> 00:18:32.100
Franz Franchetti: Book of interest, you get an algorithm, you get a couple of people probably a numerical analyst some performance engineers some

100 00:18:33.060 --> 00:18:39.570
Franz Franchetti: One who can really write high performance code and you write some high performance implementation of the Colonel.

101 00:18:40.020 --> 00:18:48.060
Franz Franchetti: Probably as a math library like the intellect kale or Dean to like BP or DSL or something like that in order to

102 00:18:48.810 --> 00:19:01.200
Franz Franchetti: Make the software reusable and have really high performance. And once you're done. If you are a supplemental or IBM, you can start over, because by the time you're done. The next process or has been released.

103 00:19:01.710 --> 00:19:12.090
Franz Franchetti: And so what we've been trying over the years is to basically automated approach so that the people doing that work really can go on to higher levels of the stack to

104 00:19:12.480 --> 00:19:22.830
Franz Franchetti: implement new functionality, instead of re optimizing the matrix multiplication, Colonel. The FST Colonel the convolution colonel and all these kinds of things. Every single time.

105 00:19:24.180 --> 00:19:33.750
Franz Franchetti: And so well. This started out as a automatic performance tuning approach that was inspired by FW, and at last, and five pack.

106 00:19:34.740 --> 00:19:45.810
Franz Franchetti: Maybe 20 ish years ago has turned into a huge effort to really build a catalog of the known algorithms of across a wide, wide range of numerical computations.

107 00:19:46.410 --> 00:19:54.630
Franz Franchetti: And beyond those are including graph algorithms at the point here then is that you really write down and all at once.

108 00:19:55.230 --> 00:20:04.440
Franz Franchetti: Liking the library, you basically make. It's the equivalent of making the library machine readable to the system and you capture the knowledge that algorithm optimizes and

109 00:20:05.400 --> 00:20:16.470

Franz Franchetti: Program optimize and performance engineers. Have you tried to capture all that optimization knowledge and then you will let the system rights to software instead of people writing the software.

110 00:20:18.060 --> 00:20:25.710
Franz Franchetti: And the goal is to get comparable performance or hopefully a new platform to outperform the humans because

111 00:20:26.790 --> 00:20:38.550
Franz Franchetti: High level of abstraction itself is not necessarily the problem getting the high level of abstraction and forwards compatibility to new machines and high performance at the same time is the hard question.

112 00:20:39.270 --> 00:20:56.880
Franz Franchetti: So that's what we have been after in spiral over the last 20 years. And so in order to give you a little bit of an idea of how we are approaching that there are three main pillars in the talk. The first pillar is operator language on how are we representing algorithms.

113 00:20:58.050 --> 00:21:15.930
Franz Franchetti: The second one is, how are we bringing the hardware into the mix. And the third one is, how are we going to make the whole approach useful and topical for users who do not want to learn new languages, and then we'll have a little bit of a summary.

114 00:21:17.460 --> 00:21:19.470
Franz Franchetti: So the basic idea.

115 00:21:21.330 --> 00:21:32.460
Franz Franchetti: Of spiral is let's capture mathematical computations, not as programs, but as mathematical specifications.

116 00:21:33.150 --> 00:21:49.050
Franz Franchetti: That tell us the semantics in it in terms of input output behavior. But really, it's really just a specification and other program that tells us how we do it. And so in mathematics and mathematical function or the higher the more complicated equivalent

117 00:21:49.620 --> 00:21:51.450
Franz Franchetti: Operator is

118 00:21:53.400 --> 00:21:54.180
Franz Franchetti: The best

119 00:21:55.830 --> 00:22:01.650
Franz Franchetti: Way of doing that because it's basically stateless, it can handle high dimensional data.

120 00:22:03.270 --> 00:22:20.880
Franz Franchetti: And so that is the core abstraction of using and it really tap can capture almost any computation. If you look at it the right way. And so the simple list example that we have here is a schema product which maps to actors to that scale a product

121 00:22:22.410 --> 00:22:33.510
Franz Franchetti: But if you will not get state in, you have to explicitly pass it in and have it come out again and higher dimensional data is linear, rational, this kind of thing so

122 00:22:35.850 --> 00:22:52.560
Franz Franchetti: An example would be a robotic problem here, which is you have a robot. That's the blue dog here and the robot, we want to compute and amazingly do something like a emergency system that the robot never

123 00:22:53.280 --> 00:23:03.090
Franz Franchetti: Runs actively into an obstacle until they can write that with Newtonian mechanics and all these kinds of things geometry. That's why we had Newton and Euclid at the early slides.

124 00:23:03.510 --> 00:23:12.690
Franz Franchetti: And usually use right it's like we have up here, which is just an equation. The problem is that that equation is the equivalent of mathematical assembling

125 00:23:13.170 --> 00:23:25.620
Franz Franchetti: It basically has additions modifications subtraction. A couple of other things comparisons and you can express anything you want, but because of the generality, you lose all the

126 00:23:26.070 --> 00:23:42.060
Franz Franchetti: Abstraction and the structure of the computation. And so what we do is the restated as an operator. They were basically lift the mathematical expression into a type system. The Act, the type C at a higher level operations like

127 00:23:43.080 --> 00:24:00.600
Franz Franchetti: Lambda functions and whatnot, you'll see later on. And we use the mathematical abstractions like norms distances polynomials and things like that to describe the computation. So this is much harder to parse for a human, but it basically has much higher level information with it.

128 00:24:02.490 --> 00:24:14.100
Franz Franchetti: Now, what we then have to do is we have to formalize the math library and here it's actually something like linear algebra one and two and not leave em.

129 00:24:15.420 --> 00:24:32.700
Franz Franchetti: And formalize that and write it out and this operator formalism, and that gives us a language that at the same time is basically vault math books have and it's a computer language that has type information. And so here are a couple of examples.

130 00:24:33.720 --> 00:24:42.180
Franz Franchetti: That you're using here and distribution systems here, the distance with the maximum of x and y is

131 00:24:43.740 --> 00:24:53.400
Franz Franchetti: What we would like to use as an example, until you see all of those operations here give you a specification, but you would usually not computed like that.

132 00:24:54.090 --> 00:25:03.330
Franz Franchetti: And so then we build a computer language automated. So we have operations which would become higher order operators and you see that the of invariant here because

133 00:25:03.720 --> 00:25:16.380
Franz Franchetti: Whenever you compose or whenever you apply operation to operators to get a new operator and there is a little bit of syntactic sugar like shorthand notation for infants and so forth. We can nicely write up

134 00:25:18.330 --> 00:25:29.340
Franz Franchetti: A big nice language that captures linear algebra and Otter maybe standard out abstract algebra and things like that for us in a computer algebra system.

135 00:25:30.570 --> 00:25:39.900
Franz Franchetti: Now this is the high level and this high level is arbitrarily extensible. As long as you provide explanation. It's kind of like a user library as long as you provide a library that explains

136 00:25:40.410 --> 00:25:49.110
Franz Franchetti: The, the high-level concepts to the system, you're good. It's just that eventually everything has to be explained recursive leading in terms of basic operations.

137 00:25:49.620 --> 00:26:08.160
Franz Franchetti: And that is the equivalent of something like a functional language inside spiral that has the equivalent of maps mapping zips fools on fools and things like that. However, they are still mathematical object. So they are at the same time math objects and they are also a functional language.

138 00:26:09.180 --> 00:26:16.920
Franz Franchetti: And in that functional language, we can write the thing that we had before for the for the robot not painting into the wall. It's this little program.

139 00:26:17.520 --> 00:26:28.320
Franz Franchetti: If you were to rewrite that one in a functional language, you will see, well, it's just a couple of maps in there are a couple of producers and thing and a couple of points was operations which maps.

140 00:26:30.030 --> 00:26:42.330
Franz Franchetti: So the only thing though is that this way of writing a program is not meant for humans. This is meant to be an internal representation that automatically is going to be derived by our system.

141 00:26:43.770 --> 00:27:01.680
Franz Franchetti: And so we need two things we need a entry point into the system where we, for example, see well really safe distance can be expressed as a polynomial evaluation compared to a distance evaluation and that can be written by hand or that can be done with some

142 00:27:02.940 --> 00:27:07.440
Franz Franchetti: Theory improver like the camera theory improver Andre plots of was part of that to

143 00:27:09.270 --> 00:27:18.540
Franz Franchetti: Use differential equation and variance and things like that. And then there's the one time, effort to have a mathematical library, which is really formalizing

144 00:27:19.110 --> 00:27:32.820
Franz Franchetti: elementary level algebra and basic functional programming in our system and you see here these. It's just a handful of rules that break down higher level concepts like a distance into eight norm.

145 00:27:33.690 --> 00:27:41.490
Franz Franchetti: Or as Kayla product into a map and reduce or a polynomial into partially below that killer product and things like that.

146 00:27:41.970 --> 00:27:52.650
Franz Franchetti: So this is where a math and math and computer scientists have to sit down together and figure out if familiarization often knowledge off the

147 00:27:53.460 --> 00:27:56.190
Franz Franchetti: underpinnings of all the algorithms, who would like to write out

148 00:27:57.120 --> 00:28:07.440
Franz Franchetti: And once you have that you're good to go and just want to point out that this is the inspired and very similar to how originally a symbolic integration was done.

149 00:28:07.800 --> 00:28:13.140
Franz Franchetti: They generally have lost and pick rule based di system that does function substitution. It has basic

150 00:28:13.920 --> 00:28:20.880
Franz Franchetti: Functions and a couple of tables. It's was arbitrarily extensible. And whenever you apply a substitution rule.

151 00:28:21.510 --> 00:28:31.290
Franz Franchetti: You don't change anything in a formula because it's semantics preserving rule, and we all learned in high school and early years of college. But then eventually

152 00:28:32.190 --> 00:28:39.690
Franz Franchetti: We don't want to do it by hand anymore, and therefore tools like mathematics and maple computer algebra systems automate that now.

153 00:28:40.050 --> 00:28:57.150
Franz Franchetti: The real estate of the art here is way beyond what I'm explaining here that's besides the point. My point is really just that retreat program generation and performance optimization. The way a computer algebra system treats something like integration. It's a pure rule based approach.

154 00:28:59.190 --> 00:29:09.060
Franz Franchetti: Now, once that is done what we have to do is he has to go through a couple of layers of different domain specific languages that are all

155 00:29:10.260 --> 00:29:20.190
Franz Franchetti: Add a new a couple of new concepts as then every you lower from one obstructions. The next abstraction. And whenever you lower from one instruction. The next one you gain

156 00:29:20.910 --> 00:29:26.100
Franz Franchetti: This new concept but you lose some ease of analysis and so

157 00:29:26.730 --> 00:29:37.650
Franz Franchetti: One of the next levels to be half is called sigma ol and the signal stands for the interactive some where it all came from originally. And the idea is that we stay as a mathematical

158 00:29:38.400 --> 00:29:50.160
Franz Franchetti: Domain Specific Language that stays point free and then states declarative. But while the highest level language has maps and reduces but basically his integrators

159 00:29:50.850 --> 00:29:55.890
Franz Franchetti: The duration order and the iteration itself is not explicit in the duration variables not explicit

160 00:29:56.580 --> 00:30:08.700
Franz Franchetti: So we lower now to a thing where the iterations are made explicit and they can be parallel. They can be reduction, all kinds of things. And that signified here by the iteration operation that we have in the middle.

161 00:30:09.420 --> 00:30:22.140
Franz Franchetti: Now once you start doing that you have to add arrays and we have to bring in a race in the abstraction that's compatible with our math. And so we basically see a basic

162 00:30:22.590 --> 00:30:40.680
Franz Franchetti: basis vector signifies a rewrite of an array and then we have atomic functions which are the operations that basically map values to values and they stay at this point over the real numbers, the arbitrary functions from reel to reel, but they are only mapping scales.

163 00:30:41.700 --> 00:30:48.000
Franz Franchetti: And so by doing that, we can get a level down and we can describe loop structure and recursion.

164 00:30:49.530 --> 00:30:54.090
Franz Franchetti: And no longer describe the data flow, but we still don't quite have it program.

165 00:30:55.680 --> 00:31:07.440
Franz Franchetti: I don't think that level of abstraction, we can do a number of performance optimizations. The in a standard compiler are very hard because of the analysis that you have to do, but

166 00:31:08.220 --> 00:31:12.960
Franz Franchetti: In our presentation. It turns out that these are very simple pattern match rewrites

167 00:31:13.530 --> 00:31:25.560
Franz Franchetti: That are usually context free and you just locally substitute and you just search until convergence and then you're done until. These are ideas like if you map and then you loop you might

168 00:31:26.160 --> 00:31:36.180
Franz Franchetti: Pull the map operation into the loop or if you map one faction and you map the second function, you can compose the map functions in the map them once and things like that.

169 00:31:38.550 --> 00:31:45.240
Franz Franchetti: Can easily be expressed and allow us to have two very, very powerful optimizations.

170 00:31:47.010 --> 00:31:57.750
Franz Franchetti: Now, once that is done. We have to go the next level of abstraction down, which is you have to leave the land or formal us and Andrew, the land of actual code.

171 00:31:58.140 --> 00:32:07.290
Franz Franchetti: And so what we did is we are defining our own abstract code that we called ICO internal code that goes back to David popular in the original spiral project.

172 00:32:07.950 --> 00:32:17.460
Franz Franchetti: And as a code abstraction. This language really is nothing special this language is really just a imperative little language.

173 00:32:17.880 --> 00:32:29.940
Franz Franchetti: But the way all programs that we use are constructed make it very special. So, at the same time it program that was constructed by our system is a abstract syntax tree.

174 00:32:30.390 --> 00:32:39.330
Franz Franchetti: It can be represented or is representing a abstract see program, but it's also still the equivalent math operator that we've started out

175 00:32:39.780 --> 00:32:47.550
Franz Franchetti: Modular floating point numbers did eventually come in and therefore it has a pure functional interpretation. And it really is a lambda function.

176 00:32:48.030 --> 00:33:01.200
Franz Franchetti: And so what do you have here is you have a imperative program that is basically a C program that has all these nice properties that allow us to now as the floor to see code level.

177 00:33:02.100 --> 00:33:09.660
Franz Franchetti: To do more optimizations that we couldn't nicely do at CMS level. So you see the pattern here and the theme is

178 00:33:10.200 --> 00:33:23.010
Franz Franchetti: Do the optimizations at the highest level that you can do that. But once you've exhausted the possibilities of that level lower to the next abstraction layer keep whatever you can to do the next optimization that become feasible at that level.

179 00:33:24.300 --> 00:33:33.660

Franz Franchetti: And so, turns out that the translation from the math formula that has scales and basis vectors and all of that into a this programming language.

180 00:33:34.110 --> 00:33:39.300
Franz Franchetti: Is a very simple rule based system again that where every rule is again semantics preserving

181 00:33:39.780 --> 00:33:48.510
Franz Franchetti: And so we can basically write something like a certified compiler with a couple of rules that translate it from the higher level math to the

182 00:33:49.440 --> 00:34:00.750
Franz Franchetti: Factual programming math to the imperative program and it's still the same thing. And since every single rule is semantics per serving. You basically get this unbroken chain of semantics preserving

183 00:34:02.220 --> 00:34:06.060
Franz Franchetti: Substitutions that at the very end will give us a proof that these are correct.

184 00:34:07.080 --> 00:34:09.000
Franz Franchetti: And so here you see the full

185 00:34:10.860 --> 00:34:19.230
Franz Franchetti: The full stack. We start out at the highest level where we start off with a mathematical specification that contains abstract concepts like polynomials

186 00:34:20.160 --> 00:34:31.410
Franz Franchetti: Distances in n dimensional space and things like that. And the highest level. We have a special substitution rule system that can do expansion and backtracking.

187 00:34:31.830 --> 00:34:46.890
Franz Franchetti: That brings us eventually the search and the optimization capabilities. Once you have settled on a particular expansion of the problem. We have a sequence of rules systems that in that switch back and forth between the blue and the green

188 00:34:48.660 --> 00:34:59.190
Franz Franchetti: The blue is always a lowering step that goes from a higher level representation to the next lower level representation. And then the green one comes in and cleans things up in a conflict term rewriting system.

189 00:34:59.940 --> 00:35:13.260

Franz Franchetti: That knows it's going to reach normal form because you have constructed it like that and wanted has reached normal form be go back and step down to the next lower level representation be clean up again the

190 00:35:13.800 --> 00:35:25.710
Franz Franchetti: Lower be clean up the lower. And so one way of one pass through all of that can easily be 10,000 or hundred thousand rule applications and the longest

191 00:35:26.100 --> 00:35:36.240
Franz Franchetti: Rule application sequences that we have seen can run 24 hours or longer. So this is substantial computation substantial symbolic computation in the computer algebra system. This is not

192 00:35:36.690 --> 00:35:45.150
Franz Franchetti: The usual thing in a kumbaya very in 30 seconds you would like to get an answer, but it's all symbolic and every rule application is semantics preserving

193 00:35:46.140 --> 00:35:53.850
Franz Franchetti: And the final results them is code that looks approximately like that. So this is not very long code. What you see here.

194 00:35:54.420 --> 00:36:00.840
Franz Franchetti: So this is like 50 lines of code hundred lines of code can be a couple of thousand lines of code largest is done is in the millions.

195 00:36:01.620 --> 00:36:11.100
Franz Franchetti: But for every single line in here. We know exactly why it's there. We've approved formal proven some cases informal proven out of cases.

196 00:36:11.670 --> 00:36:21.510
Franz Franchetti: White standby. It's correct. And it uses the latest, greatest instruction sets that are really, really hard to us and only the most expert programmers would usually use

197 00:36:22.500 --> 00:36:29.940
Franz Franchetti: This year, in particular has is using interval arithmetic and has a wrinkle from numerical analysis and air propagation on top of it.

198 00:36:30.480 --> 00:36:49.470
Franz Franchetti: And so this is really for extremely high value cold like a controller in a airplane or the colonel in the math library or things like that that we usually would use that. And so we have compiled this high level abstract program into this low level almost assembly program.

199 00:36:51.300 --> 00:37:04.620

Franz Franchetti: So now that we have seen how we can start off with the math formalized math and build a rule system that allows us to step by step lower it all the way down to high performance code.

200 00:37:05.190 --> 00:37:14.790
Franz Franchetti: The question is how are we going to build a system that we can easily retargeting across many different platforms, we would like to have the same math.

201 00:37:15.450 --> 00:37:27.540
Franz Franchetti: Sometimes targeting my cell phone and sometimes targeting a supercomputer and sometimes targeting a GPU and sometimes targeting an FPGA and sometimes targeting a quantum computer for that matter.

202 00:37:28.080 --> 00:37:41.580
Franz Franchetti: So how do you design the system such that the same problem specification can be targeting all these different machines that have lifetimes of 10s of years.

203 00:37:42.990 --> 00:37:46.440
Franz Franchetti: release dates spaced out over multiple decades.

204 00:37:47.610 --> 00:38:00.750
Franz Franchetti: So that's the landscape today that spiral is targeting and has generated programs that are at least as good as what the best human programmers can do across all those you see

205 00:38:01.410 --> 00:38:13.680
Franz Franchetti: The power envelope goes from very small, like in the single digit walked all the way up to the 10s of megawatt. It could be a supercomputer it

206 00:38:15.300 --> 00:38:26.130
Franz Franchetti: Rounds across many, many different CPUs, but also GPUs. And as I said, the latest addition to be a very proud of is targeting quantum processors.

207 00:38:27.060 --> 00:38:40.380
Franz Franchetti: And so how do we do that. Well, the idea is to go as abstract as you can to really capture the properties of the data flow graphs of the programs that you're trying to map to the hardware.

208 00:38:40.920 --> 00:38:48.270
Franz Franchetti: And so this is a old slides that still captures the idea of spiral. We have you have a space of architectures.

209 00:38:48.720 --> 00:39:01.260

Franz Franchetti: And as an abstract space that is pyramid Christ by machine parameters not cycles or Layton sees Buck structure parameters like how large is your mic to register, how many cores to you have

210 00:39:01.920 --> 00:39:16.080
Franz Franchetti: Things like that and give an abstract algorithm space. That's the land that's captures the things that we would like to compute can be FTS can be matrix multiplication can be signal processing can be simulations, whatnot.

211 00:39:16.890 --> 00:39:25.380
Franz Franchetti: And then both of them are formalized in the same abstract language that's mathematical formulas to be crafted for that purpose.

212 00:39:27.000 --> 00:39:39.510
Franz Franchetti: And these in the common space are the program transformations. So we have a common space of program transformations algorithms and computer architectures that allow us to

213 00:39:40.080 --> 00:39:58.740
Franz Franchetti: Basically map them to Jada and treat as a constraint solving problem. So it's not a feed forward problem, a mapping problem, like in a compiler. It really souls a complicated constraint problem, similar to a prologue problem offer look interpretive with do it maybe

214 00:40:00.510 --> 00:40:16.380
Franz Franchetti: And so what we had to do is to formalize our formal language that we've spent the first third of the talk to describe things like Lena transformations which were the first 10 years of spiral and then either computations like communications algorithms.

215 00:40:17.880 --> 00:40:26.130
Franz Franchetti: Image processing algorithms, but also differential equations HBC simulations and machine learning and AI algorithms that should be on that slide.

216 00:40:27.660 --> 00:40:37.380
Franz Franchetti: So they build up this library and a spiral that page. We're basically open source effort to try to capture all of that and make it available step by step.

217 00:40:37.950 --> 00:40:51.390
Franz Franchetti: And so you see the theme infrastructure, the theme language that captures all of that, but it's not as general as a programming language. It can be as general, it's still just capturing the mathematics of the algorithms and not the actual programs.

218 00:40:52.440 --> 00:41:08.610

Franz Franchetti: And there's a second layer that's really orthogonal to describing the algorithms and that layer is describing the hardware. And what you see here is you see these funny little hats or bars or annotations on the formulas

219 00:41:09.180 --> 00:41:13.200
Franz Franchetti: And that's a tagging mechanism that's designed to be auditor look forward

220 00:41:14.460 --> 00:41:21.030
Franz Franchetti: generalizable whenever new hardware comes in. We can formalize he's hardware in this language to make it

221 00:41:21.600 --> 00:41:32.940
Franz Franchetti: Interlock with the algorithm descriptions and so you have one layer. This describes the hardware and the second layer that describes the algorithms and these two layers interact through the rule system.

222 00:41:34.410 --> 00:41:41.310
Franz Franchetti: And the idea is that you can describe things like, what does it mean to be parallel. What does it mean to have

223 00:41:42.090 --> 00:41:54.630
Franz Franchetti: memory blocks. What does it mean to have cache lines in the mathematics. And so basically the idea is human expert goes and formalizes the idea that well embarrassingly parallel

224 00:41:55.290 --> 00:41:59.070
Franz Franchetti: Is a concept that I can capture. If a formula has a certain shape.

225 00:42:00.030 --> 00:42:10.770
Franz Franchetti: And so by doing that work to identifying the shape or form of us basically describing a BF of all operations that have a certain property.

226 00:42:11.280 --> 00:42:23.850
Franz Franchetti: With AHAVA property gifts as a extensible the way that's interoperable, and as independent and orthogonal across hardware platforms as possible to describe how hardware and computation interacts

227 00:42:25.530 --> 00:42:32.340
Franz Franchetti: And not just described, we tried to make all these known facts about hardware and about the interaction of hardware and algorithms.

228 00:42:32.790 --> 00:42:39.870
Franz Franchetti: As small as possible and as compostable as possible. So the idea is not the right one special case, big

229 00:42:40.380 --> 00:42:49.860
Franz Franchetti: Triangle formula. But as many little fragments as possible that, then the constraints over can pluck together in the like polar program.

230 00:42:50.310 --> 00:43:04.500
Franz Franchetti: And the problem then becomes how do we design that rule system in a way that this really happens because rule systems are notoriously hard to design in a way that you don't get like loops and things like that. And so that's really where we have to spend our effort in making that work.

231 00:43:05.940 --> 00:43:21.870
Franz Franchetti: And to give you an idea, this slide here is human almost in unreadable. But what you see here is that if you want apparently lies to operations you can paralyze them independently as long as you put a barrier in between.

232 00:43:23.760 --> 00:43:32.700
Franz Franchetti: Or if you want to transpose the matrix, you can block the matrix transpose and transpose two blocks and then transpose inside the blocks.

233 00:43:33.360 --> 00:43:41.040
Franz Franchetti: Or you do it the other way and many, many other things. So spiral has just hundreds or thousands, probably have rules.

234 00:43:41.820 --> 00:43:54.420
Franz Franchetti: That maybe even 10,000 that capture these ideas every one of these rules is either a integer identity or a identity that you would walk to a whiteboard and throw out and explain to someone

235 00:43:56.130 --> 00:44:09.810
Franz Franchetti: And we will we have to do is find the proper guards the proper conditions and the proper substitutions on it. And that's the work that you're doing to formalize the algorithms formalized the program transformations and formalize the hardware.

236 00:44:10.980 --> 00:44:13.530
Franz Franchetti: Once all of that is said and done, we

237 00:44:15.000 --> 00:44:20.340
Franz Franchetti: Can get to the to actually solve the problem. And so what you're doing there is

238 00:44:21.210 --> 00:44:30.240
Franz Franchetti: You start off with the problem definition. Save it would like to do at eight point and 50 on the engine X instruction set treated as to director of complex numbers.

239 00:44:30.810 --> 00:44:42.990
Franz Franchetti: That gives rise to to known fact, first we need to basically do to way complex double which tells us there are a bunch of phase operations.

240 00:44:43.410 --> 00:44:58.050
Franz Franchetti: That the Harvard can do and that's a library inside the stereo system that understands the harbor and that's understands. Well, if you have too complex numbers packed in the vector register, then you can do to way vector addition efficiently by just a standard victimization.

241 00:44:59.160 --> 00:45:08.370
Franz Franchetti: And the proper instruction for complex that the multiplication also exist. And so that's been formalized then there's another library that knows

242 00:45:09.120 --> 00:45:18.330
Franz Franchetti: Things like the cool to get 50 the radar FST. The good Thomas at 50 and our algorithms and they're all every one of those. Usually there's a chapter in the textbook.

243 00:45:18.630 --> 00:45:25.920
Franz Franchetti: Here, it's a single one liner rewrite rule that explains. Well, if you have an FFT of a certain size is constraints, it becomes that one.

244 00:45:26.790 --> 00:45:34.470
Franz Franchetti: That we have the program transformations. They just all kind of tiling rules and victimization and parallelization rules and blocking rules.

245 00:45:34.980 --> 00:45:42.840
Franz Franchetti: That we have in order to break down bigger loops into smaller loops and loop fission fusion distribution and whatnot.

246 00:45:43.380 --> 00:45:49.500
Franz Franchetti: And now you'll see that the read the blue integrate real system just get thrown together into a big, big rule system.

247 00:45:49.980 --> 00:46:01.080
Franz Franchetti: And they, the SEC that what you get is a usual space and then you drill space thing the section would be this dark gray line here all points on the intersection our solutions to the problem of

248 00:46:02.280 --> 00:46:05.730
Franz Franchetti: Do an FFT of size eight on a VX

249 00:46:06.930 --> 00:46:17.790

Franz Franchetti: Given all the degrees of freedom that the rule system has until now, our constraints over constructs that black line one after the other, and every point on the black line is a solution.

250 00:46:18.150 --> 00:46:25.830
Franz Franchetti: Given by a tree and gets translated into a data flow graph in our internal language and then compiled to program.

251 00:46:26.190 --> 00:46:38.220
Franz Franchetti: And once it's compiled it gets measured and then automatic performance tuning is done and then feedback comes back to tell the system. What's the next solution to try out to find. Finally, the best one.

252 00:46:38.880 --> 00:46:45.900
Franz Franchetti: And so here we are in expansion backtracking part of the system. And the next slide. We're going to see the gray stack going down.

253 00:46:47.460 --> 00:46:54.600
Franz Franchetti: So once the system has committed for the time being on a particular data flow graph.

254 00:46:55.920 --> 00:47:06.120
Franz Franchetti: Then the first step that we just looked at is done and now for that particular first step, we have to do the recursive descent conflict term rewriting

255 00:47:06.810 --> 00:47:23.190
Franz Franchetti: Step by step and you see that we go through and we have a data flow graph, then we make loops explicit and then we begin again. They've actually a C program. And it's just a very, very abridged version of what happens, but

256 00:47:24.330 --> 00:47:25.920
Franz Franchetti: So this is kinda like the step by step.

257 00:47:27.150 --> 00:47:36.810
Franz Franchetti: Procedure that gets us from the problem specification, all the way down to extremely high performance assembly program and that shows us how we can search

258 00:47:38.190 --> 00:47:44.250
Franz Franchetti: Now, since all the operations are basically mathematical operations.

259 00:47:44.820 --> 00:47:53.790
Franz Franchetti: If all operations, happened to be linear, which is true for for a lot of the algorithms and us through all this in the past, but nowadays it's only true for a subset

260 00:47:54.270 --> 00:48:01.620
Franz Franchetti: In that case, we can have very strong verification properties because Linda operators can always be turned into matrices.

261 00:48:02.070 --> 00:48:18.300
Franz Franchetti: And so we're in the computer algebra system where we can verify matrices, up to a pretty large size we can do symbolic correctness checks and also just empirical correctness checks. Because we know what the answer should be

262 00:48:19.650 --> 00:48:30.750
Franz Franchetti: Now for a subset of what you're doing here, what you get is also fear improver correctness and that one. We are doing

263 00:48:31.980 --> 00:48:37.140
Franz Franchetti: And my student body as Oliver over at CMU is about to graduate on that baby really

264 00:48:38.190 --> 00:48:47.370
Franz Franchetti: Take the cop computer algebra system to prove for not nonlinear operations that is still correct as a side thread here that is also can be done.

265 00:48:48.420 --> 00:48:52.440
Franz Franchetti: So now that they have basically given the first idea of

266 00:48:53.490 --> 00:49:05.670
Franz Franchetti: How can we represent the algorithms and how can we get performance portability. The remaining question is how do we get users to use the system because what you've seen so far.

267 00:49:06.300 --> 00:49:13.560
Franz Franchetti: I personally would consider pretty scary. It's a lot of math. A lot of computer science. A lot of non standard programming.

268 00:49:14.220 --> 00:49:19.470
Franz Franchetti: So if you were to just say, Okay, guys, here it is. Go and program your problem in it.

269 00:49:20.130 --> 00:49:30.840
Franz Franchetti: Then it's not going to work very well. And I say that in hindsight, because we've tried that and we, the adoption of spiral is very hard because you have to know all of that stuff and so

270 00:49:31.290 --> 00:49:38.160

Franz Franchetti: Two years ago, approximately, we started working with our Franz from the excess kill computing project to develop a 50 x

271 00:49:39.030 --> 00:49:52.410
Franz Franchetti: A library from them for spiral and that particular pattern that you'll see in the next couple of slides has turned out to be a very good way to go. And we have started doing that for not just the 50s, but for a variety of domains.

272 00:49:52.830 --> 00:49:58.590
Franz Franchetti: In the latest project, which is the snow white project, though, at the very end scene. So the idea is

273 00:50:00.090 --> 00:50:07.410
Franz Franchetti: That if you think about what the high performance computing community and really well was for numerical in algebra.

274 00:50:08.550 --> 00:50:20.010
Franz Franchetti: They defined la pack and Bloss and basically wrote a specification and let vendors optimize the Bloss the low level of thing and then

275 00:50:21.450 --> 00:50:30.090
Franz Franchetti: Computer computing researchers can still belong, but really la Pack has been relatively stable interface.

276 00:50:30.420 --> 00:50:39.570
Franz Franchetti: Over the years that everybody rides against and nobody has to worry too much about performance because at least a very good performance as a chief by just linking against the vendor boss.

277 00:50:40.170 --> 00:50:48.240
Franz Franchetti: So the same does not exist for FTS although FST w is the standards that's been around for quite a while.

278 00:50:49.620 --> 00:50:57.960
Franz Franchetti: But the problem is that with the advent of GPUs and FPGA and new special purpose hardware.

279 00:50:59.550 --> 00:51:10.530
Franz Franchetti: The model is no longer really works. And so what we set out to do is to generalize the 50 layer a little bit and move it from C to c++ in the long run.

280 00:51:11.220 --> 00:51:28.980

Franz Franchetti: To basically build the 2020 vintage have a 50% of the 50 library. But that by itself only gets us that far, the real thing is to build the equivalent of La pack and we call it spectral pack and that is much harder because

281 00:51:30.000 --> 00:51:34.110
Franz Franchetti: The space of spectral algorithms hasn't been quite

282 00:51:35.340 --> 00:51:43.050
Franz Franchetti: Structured yet like linear algebra and the fact that these algorithms are all order of and login and not in square and cube.

283 00:51:43.440 --> 00:51:52.800
Franz Franchetti: Means that you cannot really do a nice library based approach there because a you lose everything in the interfaces and never gets to the actual computation.

284 00:51:53.280 --> 00:51:57.870
Franz Franchetti: And so what we decided is the build a library from then that looks and feels

285 00:51:58.200 --> 00:52:05.970
Franz Franchetti: Like la pack into Bloss. But really, instead of having a library there. We have a code generator there that purpose built your particular computation.

286 00:52:06.330 --> 00:52:12.570
Franz Franchetti: Sometimes it just in time compiler sometimes like it offline compiler code generator auto tuner depending on

287 00:52:12.900 --> 00:52:19.440
Franz Franchetti: Which a scenario you're in. If you are in a high performance embedded computing scenario that you're flying in

288 00:52:19.740 --> 00:52:26.910
Franz Franchetti: Some kind of equal, you would not want the just in time compiler to kick in, you would like to have everything done before you go on a mission.

289 00:52:27.360 --> 00:52:31.320
Franz Franchetti: And a supercomputer, you are in a queuing system and you can run on a smaller node.

290 00:52:31.950 --> 00:52:40.680
Franz Franchetti: To profile everything before you go into big note and in a more consumer setting. You might just have a reference implementation that runs

291 00:52:41.070 --> 00:52:50.340

Franz Franchetti: And then the just in time compiler kicks in and optimize as well. He used to default and eventually switches over to the high performance version. So there are many ways of how that can be done.

292 00:52:51.060 --> 00:52:59.550
Franz Franchetti: But the key thing is, you first have to get people to write their problem specification. And so what do you usually get is a slide like that.

293 00:52:59.910 --> 00:53:08.280
Franz Franchetti: Yet someone math in that case from our collaborators from LDL that tell us how a particular partial differential equation could be solved.

294 00:53:09.060 --> 00:53:14.970
Franz Franchetti: And you see that this is greens function integral transforms and whatnot.

295 00:53:15.570 --> 00:53:31.830
Franz Franchetti: So what you have to do is you have to convert that into something that as a computational scientists is more our language which will do something like that you have an input box that's blue and has a certain dimension that gets zero pattern to a larger dimension, Dan.

296 00:53:33.090 --> 00:53:39.420
Franz Franchetti: Dan pointers multiplied with a different data cube that has certain symmetries has certain size.

297 00:53:39.960 --> 00:53:45.540
Franz Franchetti: Once you're done you inverse at 50 and then you take only the yellow one, and throw away everything else.

298 00:53:46.140 --> 00:53:57.780
Franz Franchetti: So that is how we would like to draw it on a whiteboard to explain to somebody. This is the algorithm that works for whatever reasons the applied mathematician has. But that's the computation. We have to do

299 00:53:58.290 --> 00:54:07.860
Franz Franchetti: And so that will be asked to write. And so if you would like to do now is to give the domain scientist interface that does that. And so this year is the old see version.

300 00:54:08.730 --> 00:54:17.700
Franz Franchetti: That basically looks almost like in FST W call. It's a bit more complicated. And we have also c++ versions of it and

301 00:54:18.180 --> 00:54:32.790

Franz Franchetti: More modern versions. But what you really do is you basically ride a constructor in object oriented system here but you say instead of doing one library call after the order you hands back a sequence of library calls that then can be lazy.

302 00:54:33.330 --> 00:54:40.980
Franz Franchetti: Delay that really true, or can be Justin time compiled and whatnot until what really happens is you have an executable.

303 00:54:42.030 --> 00:54:51.180
Franz Franchetti: That runs and you have a couple of calls sites and instead of actually running the code you trap it in the spiral system.

304 00:54:51.870 --> 00:54:57.390
Franz Franchetti: That has all of the infrastructure that we've talked in the first two parts of the talks which is the upper half here.

305 00:54:57.840 --> 00:55:07.260
Franz Franchetti: And then use a steady infrastructure to generate purpose built modules for the particular call side and the particular runtime parameters.

306 00:55:07.950 --> 00:55:14.100
Franz Franchetti: And so there's of course a lot of configuration necessary to do that. Right. But basically it's like a just in time compiler.

307 00:55:14.490 --> 00:55:23.520
Franz Franchetti: That purpose fields for the particular hardware into particular execution what you want and can take the time, right, because if you own a supercomputer. It doesn't really matter.

308 00:55:24.030 --> 00:55:32.280
Franz Franchetti: And then you're on really really efficiently and that's can be wrapped up in C and c++ in Python.

309 00:55:32.700 --> 00:55:42.450
Franz Franchetti: In Julia, maybe in other languages and we're just looking at that as our new paradigm of how to actually get the complications off the spiral system.

310 00:55:42.900 --> 00:56:03.630
Franz Franchetti: And out better adopted by people who do not really want to have to deal with all the complexity. And so that has turned out to actually work and via of, you know, pursuing that in a number of directors and so let me just give you the summary. Little bit of where we are right now.

311 00:56:05.040 --> 00:56:15.600

Franz Franchetti: So first, generally speaking, spiral can generate software implementations and hardware implementations on FPGA as

312 00:56:16.230 --> 00:56:24.960
Franz Franchetti: For FFT like and all the other algorithms that we've talked about that are on par with the best a human can do

313 00:56:25.800 --> 00:56:37.860
Franz Franchetti: If it is a codes that human programmers have spend quality time with if it's code that humans have not optimized effect of five or 10 gain over the best out there that's already decent

314 00:56:38.400 --> 00:56:55.980
Franz Franchetti: Can be easily achieved and example is for to power FTS spiral beats something like the intellect kale and FW by a couple of percent for something like a cosigner assigned transform the same graph shows to view like five times as fast

315 00:56:57.420 --> 00:57:01.020
Franz Franchetti: And not observation. So these are relatively old plots, because

316 00:57:01.830 --> 00:57:15.300
Franz Franchetti: You know, it's been going on for a while and it's becoming problematic to just track the performance results. So it's kind of like one of the things on my list to rerun all the old experiments and get later performance numbers. But here's the important thing

317 00:57:16.470 --> 00:57:26.760
Franz Franchetti: If you look at this one. This is an old GPU code, what really matters here is that the blue line that video library is faster for a couple of sizes. So what we do in this case is it try to understand

318 00:57:27.240 --> 00:57:36.660
Franz Franchetti: Are we missing an algorithm. And if they're missing an algorithm, then the act is algorithm or for that matter I optimization method or instructions that they're using

319 00:57:37.920 --> 00:57:42.870
Franz Franchetti: The assets to spiral and rerun and then usually that line jumps up

320 00:57:44.190 --> 00:57:50.520
Franz Franchetti: Now you see that here for ozone everything something with the one TIP program from

321 00:57:51.030 --> 00:58:01.500
Franz Franchetti: UK where we did special FTS of special problem sizes and there be kept like three x two x five x over the state of the art.

322 00:58:01.800 --> 00:58:13.620
Franz Franchetti: Although there are a couple of sizes, where we haven't quite spent the effort. And so if it turns out that the domain scientist says, Look, I really need size 47 build spent the effort.

323 00:58:14.130 --> 00:58:20.940
Franz Franchetti: To get that performance up or if you kind of like do it for just the bragging rights. Those are my sometimes spent the effort.

324 00:58:21.510 --> 00:58:33.330
Franz Franchetti: Right, so that that that's the idea. The idea is not study or developing new algorithms, necessarily, although sometimes we do. The idea is to try to soak up all the knowledge out there of how people optimize

325 00:58:33.660 --> 00:58:39.930
Franz Franchetti: Particular things because once the computer algebra system and the system knows it. It can just hire list to try it out.

326 00:58:41.040 --> 00:58:41.580
Franz Franchetti: So,

327 00:58:42.690 --> 00:58:52.440
Franz Franchetti: To give you an idea over the years on a single node, we have touched things from less than a lot to more than 10 kilowatt

328 00:58:52.920 --> 00:58:59.490
Franz Franchetti: From single core to 500 course of the CPU or 40,000 cores in the GPU from

329 00:59:00.240 --> 00:59:13.710
Franz Franchetti: Couple of kilo, two megabytes to multiple terabytes from the giga flop range or below to the multiple hundred teraflops range 20 years of really states. It's a one to

330 00:59:14.220 --> 00:59:20.940
Franz Franchetti: One source code that I'm modified can always get to the highest performance across this dynamic range and a single node.

331 00:59:21.480 --> 00:59:30.270
Franz Franchetti: Now, on top of that we've touched anything from single node CELL PHONE TO THE BIGGEST supercomputers at that time that was

332 00:59:30.660 --> 00:59:42.090

Franz Franchetti: Blue Jean Pierre are gone. And right now, we're working on the summit machine. The current number one. So basically, it's always a game of getting access to the latest, greatest and trying to move it.

333 00:59:42.450 --> 00:59:55.800
Franz Franchetti: But to really be able to keep a system alive and around that really keeps the knowledge of all of that rat race event and it's databases. So that's why spiral has become

334 00:59:56.340 --> 01:00:10.410
Franz Franchetti: AI system, it was, it started off as a performance tuning system. It started out as a program generation system as a special purpose compiler, but it really over the years has morphed into something more than that.

335 01:00:12.270 --> 01:00:21.990
Franz Franchetti: Now currently what we are working on our spiral photographs that's being seated by the ideas of German Kepner over Lincoln Labs and daughters.

336 01:00:22.470 --> 01:00:33.420
Franz Franchetti: graph algorithms in the language of the knowledge of Brian, you have a best paper up a student paper on this year's H picnics. I think this week actually being presented very plaque spiral behind

337 01:00:34.350 --> 01:00:42.930
Franz Franchetti: The graph Bloss interface to do the same thing that I talked about 15 very proud. And so we started doing special for quantum computing and

338 01:00:43.620 --> 01:01:02.670
Franz Franchetti: One of my students just got a supercomputing Best Paper, a Best Poster candidate there so we work on that a harvest of the CO design and generating automatically accelerators on FPGA. And now with zoom us new capabilities to fab chips again.

339 01:01:03.870 --> 01:01:09.360
Franz Franchetti: This is also one of the things that we are always doing. And then we do the entire software stack from doing

340 01:01:09.900 --> 01:01:17.520
Franz Franchetti: Accelerators for risk five on the low end all the way up to the software library that drives it and another project where

341 01:01:18.180 --> 01:01:31.950
Franz Franchetti: MIT is also involved is the type of pop up program where our take is to move spiral as AI system or high-level reasoning into compilers and that aspect of spiral is called Snow White, because

342 01:01:32.580 --> 01:01:41.130
Franz Franchetti: It's the Seven Dwarfs from physical our original originally to describe the different computational patterns and Snow White is laying the law on the fourth

343 01:01:41.970 --> 01:02:04.770
Franz Franchetti: So in a some in summary spiral. Is it emerging AI system for high performance coding for performance engineering and you can find it on spiral.net it combines algorithm formalization correctness guarantees and hardware mapping and he really forward compatibility to future hardware.

344 01:02:06.570 --> 01:02:12.420
Franz Franchetti: And if you go to spiral.net I've lost the slide. Unfortunately, you will find

345 01:02:14.490 --> 01:02:22.170
Franz Franchetti: Spiral downloadable as the open source spiral system on wwe.spiral.net

346 01:02:22.650 --> 01:02:34.800
Franz Franchetti: And you find there the tutorial for spiral you find from last year's H back all the latest papers or the latest downloads and a ever increasing documentation and

347 01:02:35.370 --> 01:02:49.410
Franz Franchetti: What I'll be doing this year and next year is probably more more talks and little presentations to walk through different aspects of the spiral system in a training effort that comes together with the open source spiral.

348 01:02:52.440 --> 01:03:06.990
Julian Shun: Okay. Thank you. Great. Thanks a lot, Franz, um, does anyone have any questions? If you have any questions, feel free to unmute yourself and speak up or you could also type in the chat. I can read it. Oh.

349 01:03:09.900 --> 01:03:16.890
Neil Thompson: Yeah, I have a question. Could you talk to the end, very briefly about the snow white project and their connection to the

350 01:03:16.890 --> 01:03:17.400
Motif

351 01:03:18.690 --> 01:03:25.350
Neil Thompson: That was done by Coca Cola. Can you talk a little bit more about that and how are you, incorporating specular

352 01:03:25.920 --> 01:03:38.670
Franz Franchetti: Yes, absolutely. So the idea is you have seen that we have a way to describe algorithms in an extremely high level. Right. So the idea is that every different motif.

353 01:03:39.750 --> 01:03:49.230
Franz Franchetti: May require its own domain specific language, but the spiral system is basically the host language that allows us because it's all math.

354 01:03:49.680 --> 01:04:11.910
Franz Franchetti: To this design sub languages of spiral. For example, you can define arbitrary infinite dimensional lists to do arbitrary lists and things like that. And so by doing that you can arbitrarily extend a spiral and extend the formalization. Now the other thing is that the front end is also

355 01:04:13.050 --> 01:04:22.140
Franz Franchetti: very extensible by our work that we've been doing in Snow White, where we basically developed a object oriented system.

356 01:04:22.680 --> 01:04:34.080
Franz Franchetti: That allows us to describe similar to originally 5050 W three does problems and solvers just it's fully object oriented and independent of the actual host language.

357 01:04:34.590 --> 01:04:41.760
Franz Franchetti: And so therefore, when you can write this, you can write small service like here's a silver for an FFT. And he has a small sober for a graph algorithm.

358 01:04:42.090 --> 01:04:52.470
Franz Franchetti: And he has a sober for a matrix factorization much you have that you can write something like a spacetime adaptive processing but pulling together a bigger solver based on smaller solvers.

359 01:04:52.830 --> 01:05:02.760
Franz Franchetti: And then to feature extraction and then maybe do some graph algorithm and AI at the very end. And so there is a invariant here that as long as everything is written in that

360 01:05:03.900 --> 01:05:06.360
Franz Franchetti: Pattern. That's really a framework that has

361 01:05:07.410 --> 01:05:17.490
Franz Franchetti: Problems and solvers, and is stateless and the semantics of the operation is eventually written in the domain specific way spiral does it

362 01:05:17.970 --> 01:05:24.510
Franz Franchetti: Inspire language after parsing everything and after doing inspector execute or it's all the same thing.

363 01:05:25.470 --> 01:05:38.220

Franz Franchetti: And so you can do cross dwarf optimization. You can do cross library call optimization. You can do just in time compilation. You can do offline optimization. You can do partial evaluation or you can just run the

364 01:05:39.510 --> 01:05:43.140
Franz Franchetti: Standard implementation and let the user debug.

365 01:05:44.550 --> 01:05:57.870
Franz Franchetti: All of that is possible. And so that's why. Basically, the idea here is to do cross dwarf or cross motif optimization. Not even just this kind of like the next level in full program optimization, so to speak.

366 01:06:02.790 --> 01:06:06.990
Julian Shun: Great, thanks. I'm very other questions.

367 01:06:10.410 --> 01:06:22.890
Julian Shun: So I have one question. I'm curious for your, for your project on the graph algorithms. Did you have to add any additional graph specific optimizations to get the good performance there.

368 01:06:23.040 --> 01:06:27.810
Franz Franchetti: Yeah, so this is an interesting one. So basically you also see the time scale of things so

369 01:06:29.010 --> 01:06:39.450
Franz Franchetti: About 10 or 15 years ago I learned a few attending H back from listening to Jeremy Kepner of the effort of writing graph algorithms in the language of linear algebra.

370 01:06:39.690 --> 01:06:47.820
Franz Franchetti: The basically Gilbert Kepner and it in bulk and others were pursuing and it has become the food droplets community know turns out that

371 01:06:48.390 --> 01:06:53.730
Franz Franchetti: The moment good is cripes graphs. So sparse matrices over semi rings.

372 01:06:54.450 --> 01:07:03.420
Franz Franchetti: You're basically inspire land because everything sparse matrices. The only thing is, it becomes irregular sparse. So yes, what we had to add is

373 01:07:03.840 --> 01:07:13.560
Franz Franchetti: The idea that matrices are stars, but no longer structurally sparse to way attends a product does. So that was one extension extension is that well.

374 01:07:14.130 --> 01:07:21.570

Franz Franchetti: The base field is no longer a real numbers. So therefore days abstraction of semi rings mapping to machine numbers.

375 01:07:22.080 --> 01:07:32.490
Franz Franchetti: And also to mapping to actual abstract numbers. So you go from semi ring to integer to machine integer of rare something like infinity becomes different things.

376 01:07:33.150 --> 01:07:43.860
Franz Franchetti: Then you have to think about things like how do you describe set intersection algorithms, independent of the same earring and independent of the data structure and independent of the veteran structures.

377 01:07:44.490 --> 01:08:00.600
Franz Franchetti: So you're basically have to design extremely fragmented object oriented programming generator that stick to the old to gather and if you did all of that as a program and not a program generator, you would lose thousands. Next performance.

378 01:08:01.770 --> 01:08:09.330
Franz Franchetti: But if you make it a compiler that basically partially evaluates everything you can do it. And so it turns out it's the second student on it now.

379 01:08:10.320 --> 01:08:19.350
Franz Franchetti: It's been going on. I first started 2007 I think we started maybe five years ago in earnest.

380 01:08:19.890 --> 01:08:34.320
Franz Franchetti: The second PhD student on it. It's longer slow going right, but we're there. No. And what we can do is we can get the same codes that are my collaborate attending little last year submitted for each Beck challenge.

381 01:08:35.100 --> 01:08:42.600
Franz Franchetti: The same code that his students would write by hand. We can automatically generate. It's basically bite for fighting distinguishable with the proper printer printer.

382 01:08:43.080 --> 01:08:59.340
Franz Franchetti: So it's again shows. I have no idea what this algorithm actually does. I don't care. Honestly, I just want his students to tell us how to do it right. And once that is done my students formalize it and then we automatically do it and then it's cataloged

383 01:09:00.690 --> 01:09:05.760
Franz Franchetti: It takes a while, you know, every dwarf every mode. This can be a PhD student or more

384 01:09:06.390 --> 01:09:22.230

Franz Franchetti: But as long as an area has its domain specific language that's really a library. It could be TensorFlow. It could be by torch. It could be GB TL, it really doesn't matter as long as there's an agreed upon interface. We can embark on doing

385 01:09:24.480 --> 01:09:26.730
Julian Shun: Great, great, that's very helpful. Thanks a lot.

386 01:09:29.400 --> 01:09:30.540
Julian Shun: Any other questions.

387 01:09:36.750 --> 01:09:47.280
Julian Shun: Okay, great. It seems like there are no more questions right now. Well, let's thank Franz again for the excellent talk. Since we're on zoom in virtual. APPLAUSE

388 01:09:48.750 --> 01:09:49.650
Franz Franchetti: Well, thank you so much.

389 01:09:50.550 --> 01:09:59.880
Julian Shun: Yeah, thanks. So I guess if anyone has any questions on spiral. You can send them to me and all for dumpster France or you could

390 01:10:00.960 --> 01:10:04.860
Julian Shun: Probably email him directly as well. So thanks again.

391 01:10:05.250 --> 01:10:08.190
Julian Shun: And we'll definitely be sure to check out the spiral project.

392 01:10:08.610 --> 01:10:12.870
Franz Franchetti: Perfect, thank you so much. Do you want updated slides to post for me, actually.

393 01:10:13.260 --> 01:10:17.340
Julian Shun: Oh yeah, that would be great if you could send me the slides, I can post them on the seminar website.

394 01:10:17.370 --> 01:10:18.780
Franz Franchetti: Will do great.

395 01:10:18.870 --> 01:10:20.370
Julian Shun: Thanks a lot. Great. Thanks a lot.