Episode 24: Extending the Roofline Model
Host: Nicole Huesman (Intel)
Guests: Aleksander Ilic, Diogo Marques (INESC-ID & IST); Sujata Tibrewala (Intel)

Nicole Huesman: Welcome to *Code Together*, an interview series exploring the possibilities of cross-architecture development with those who live it. I'm your host, Nicole Huesman.

Understanding how hardware-imposed performance ceilings impact your code can be … well … challenging. If you're struggling to assess the optimization tradeoffs between memory bottlenecks and compute utilization—be it on CPUs or GPUs—the Intel Advisor's Roofline Analysis is a welcome friend. It helps you visualize application performance in relation to hardware limitations, including memory bandwidth and computational peaks.

Today, we'll talk to two researchers from INESC-ID in Lisbon, Portugal about how they've worked to ***expand*** the Roofline Analysis model.

As a Senior Researcher in the HPC Architectures & Systems Group at INESC-ID, **Aleksandar Ilic** is currently focused on research primarily in high-performance and energy-efficient computing, highly heterogenous systems, parallel applications and computer architectures. He is also an Assistant Professor with the Department of Electrical and Computer Engineering at the IST at University of Lisbon. So great to have you with us, Aleksander!

Aleksander Ilic: Thanks, Nicole. I'm very happy to be here.

Nicole: As a Researcher in the HPC Architectures & Systems Group at INESC-ID, **Diogo Marques** is currently focused on the modeling of multi-core processors and heterogeneous systems. His work contributed to improving the accuracy and insightfulness of roofline modeling based on Cache-aware Roofline Model by proposing the memory impact metrics and roof scaling methodology presented in the Intel Advisor framework. He is currently pursuing his Ph.D. in Electrical and Computer Engineering at the IST at University of Lisbon. Thanks so much for being here, Diogo!

Diogo Marques: Thanks, Nicole, and thanks for the opportunity to participate in the *Code Together* series.

Nicole: I'd also like to welcome Sujata Tibrewala, Intel's oneAPI Developer Community Manager. Welcome back to the program, Sujata!

Sujata Tibrewala: Thank you, Nicole! Thank you so much.

So Aleksander and Diogo, can you start us off by telling us how the Roofline Analysis model has helped you in your work?

Aleksander: Yeah, thanks Sujata. So, our contribution to Roofline Modeling represents a continuous research effort that started almost a decade ago. To be precise, it actually made part even of my PhD Thesis back then where we developed the Cache-aware Roofline Model together with Frederico Pratas and Leonel Sousa, who was my PhD supervisor. So, typically, this Cache-Aware roofline model nowadays is called CARM. As Nicole already said in the introduction, the Cache-aware Roofline Model actually represents a new take on Roofline Modeling, which represents the processing limits of modern computing hardware by specifically taking into account the impacts of different levels of memory hierarchy. So, in a nutshell, the CARM, or Cache-aware Roofline Model, is a very intuitive model and it's particularly useful for application optimization. So, in the roofline charts, you can visualize where the performance of your application currently stands, and from there, you can detect what are your current bottlenecks. Then you can also derive the next optimization strategies to apply to your application in order to squeeze the maximum performance out of your architecture. It also tells you how much performance you also missed.

However, besides the performance aspects, we also extended the CARM to the peak, the power and energy efficiency limits, and we also tackle different types of devices—not only CPUs, we also derived it for the GPUs. Basically, after publishing our CARM paper, like that was several years ago, in fact, we got contacted by Intel people and they say, 'Oh, we read your paper. And we think it may be nice to actually put it in our tools.' We obviously didn't believe that people from Intel read our papers, but it actually became true. And one year after that, we actually got the Cache-aware Roofline Model in the Intel® Advisor tool. And since then it became kind of popular both in academia and industry. So even before the product release of the Intel® Advisor roofline feature, we worked with the Intel Advisor team, especially with Zakhar Matveev and all people around him, in order to improve the insightfulness of this model and also to give some guidelines, okay, how it should be constructed and all the other perks of the Cache-aware Roofline Model in it that we had before.

In fact, in 2017, these joint efforts were awarded by the HiPEAC community with the tech transfer award, which actually recognized this collaboration as one of the most successful technology transfers in Europe. So naturally our collaboration with Intel Advisor and now the oneAPI team has continued even to these dates through many scientific publications, tutorials that we jointly performed. For example, every year, since 2017, we have a roofline tutorial on SC, which is jointly done between our Institute, Intel and also the guys from Berkeley. Some done recently from even some other companies like Nvidia and also Argonne people. So, a lot of us who are working in Roofline, we join together and do a tutorial every year at SC (Supercomputing Conference) and also various other places such as IST. Recently, with the oneAPI team and Sujata in particular, we've also started adopting this content and also preparing tutorials that are in the oneAPI Dev Summits—the one we already had a year ago, and I think in two weeks from now, we'll have another one, right Sujata?

Sujata: Yeah, we are looking forward to that tutorial!

Aleksander: Yeah, so in that one, we will be talking about how we can use actually a Roofline Model—and Cache-aware Roofline Model, in particular—on GPUs, and a little bit on CPUs, in order to improve the performance of one bioinformatics application, which is actually one of the topics that they always work. In fact, the August topic for the PhD thesis, because he's my PhD student is actually aimed on providing the new ways and expanding the insightfulness of the Cache-aware Roofline Model. So within a couple of Intel research grants that we also had a couple of years ago, Diogo who was heavily involved in the expansion of the Roofline Modeling and its integration in the Intel® Advisor tool, and has also given additional features in the Intel Advisor [tool] to support the analysis. Since this is your topic, Diogo, maybe you can give us more insights on those developments.

Diogo: Sure, Professor. So, our collaboration with Intel focused mainly on the different requirements of the applications. For example, given the high diversity of instructions that the current processes use, applications can contain more scalar instructions or even using vector instructions like AVX, or even the most recent ones, AVX-512. And this is not only between applications, since it can also occur in the different phases of the same application. For example, some loop can completely different of some loop later executed. And the main issue we were having with the Cache-aware Roofline Model was that we were not considering the different performance limits of the microarchitecture that resemble these specifics of the applications. For example, an application that was scalar was characterized in a AVX-512 roofline. And this was leading to some weird scenarios, like one application that was memory-bound that was appearing compute-bound, and vice versa. And you can imagine someone like some software developer is trying to optimize his application and like these, the model doesn't provide the best guidelines.

So, instead of facilitating the process of optimization, we are in fact, making it more difficult. So, to solve these issues, we focus on scale the roofs according to these application specifics, and to do this, we perform really an extensive experimental evaluation of the micro-architecture capabilities under different execution scenarios. And also, to better identify which bottlenecks—mostly from the memory side—that is affecting and degrading our performance, we also derived some performance memory metrics that represent how much time each memory level contributed to the overall time or how much bytes (amount of information) are served by each memory. You can, in fact, see these proposals in Intel® Advisor, with the introduction of the scalar rooflines and memory share metrics that are also integrated in the Integrated Roofline Model. Also from all this research, we were able to do a publication, the Application-Driven Cache-aware Roofline Model, and also to present several tutorials in different venues, such as ISC High Performance, oneAPI Intel Dev Summit, etc. And in fact, these tutorials are probably the most inspiring part of these works since it allows us to share our knowledge in the area of application optimization, which can have impact in several areas. This is the most inspiring parts from this work because we present our knowledge about application optimization, and since we gather several software developers from different research areas, this can have an impact from diverse areas that have an impact on our society.

Sujata: Well, that's really true, yeah. The work that you do really does impact developers of all different applications. So yeah. Thank you for your work.

Now one follow-up question I have is, since we're talking about oneAPI and heterogeneous computing, to Nicole's point earlier, how do you think Intel® Advisor has addressed the needs of a heterogeneous application and how could it be improved to address the needs of such applications in future?

Diogo: So from the point of heterogeneous environments, I think since we have CPU and GPU in Intel® Advisor, we can individually evaluate the performance of those kernels, but maybe as a limitation, because we cannot see the performance as its global, let's say, because we need to divide the application to different parts. And here is the part of developing a model for heterogeneous environments. So first, for CPU, we needed to carefully design the micro benchmarks, what things, the limits of the software computation. And it's not easy to port to different device. We need to, again, perfectly design for different benchmarks that can do that exact job. And this is the challenge from the microarchitecture point of view. Then from the application point of view, there is also the need to know how the work is being distributed, how much work, because that is going to set our model. I mean, this is only what I think, because we are starting to move in that direction, but we don't still have a clear direction of how this will work.

One advantage—because it cannot meet *all* of our challenges, right?!—before, we needed two different languages, like C++ for CPUs and then OpenCL for GPUs, and now (with DPC++) we can in a single programming model, unify our benchmarks, which makes much easier our work and that it's making us going towards that heterogeneous model.

Sujata: Wow, that's a great perspective. Aleksander, did you want to add something to this?

Aleksander: No, I totally agree and support Diogo's idea. In fact, I would like to see it implemented until the end of his PhD thesis. That would be awesome. But it's quite challenging task to actually create a heterogenous Cache-aware Roofline Model for two different devices together. So, he was right in terms of our micro benchmarking strategy, we have to really think, and yes, he sees a really good potential in DPC++ to kind of alleviate that problem. But yeah, it requires significant research in order to push it, but it can be a great contribution for all of us and especially for his thesis!

Diogo: Yeah, especially for my thesis, that's true!

Sujata: Yeah, that's awesome. When you point it out, like, you know, in the past you needed different languages and now with DPC++ with the same language, you're able to target two different devices and do the profiling on the same tool, that really helps you and helps the researching.

Aleksander: Yeah, I think it really helps for example, application characterization. So that one will be easier with the DPC++ provided that we overcome designing the model in the right way, you know?

Nicole: It's been fantastic to hear really how you contributed early on to the Roofline Model and how you've really been able to contribute in its evolution. And it'll be great to see how we continue to collaborate. Thank you so much both of you for really sharing your insights. As we wrap up today, Alexander, where can we point folks to learn more about your work?

Aleksander: As always, you know, given that we're producing scientific papers, you know, typically this is the best place to start and I promise that those ones on the Roofline are not difficult to read. So we can probably provide a couple of links to some of our most fundamental papers regarding the Cache-aware Roofline Model and the extension of applications, even Cache-aware Roofline Model that Diogo was mentioning. Also we would really like to invite you to join us at tutorials, you know, when oneAPI Dev Summit, where we are going to share our experiences and explain also how the Roofline works, where we can use it and show some practical examples of how we can improve the applications and different devices.

Nicole: And it'll be so great to have you at Dev Summit. We're really looking forward to that. And Diogo, would you like to give our listeners any pointers to any additional information?

Diogo: I think they can consult the Intel® Advisor website, check out how Roofline analysis can be performed, and also they look at the code samples because they can help understand the model.

Episode 24: Extending the Roofline Model
Host: Nicole Huesman (Intel)
Guests: Aleksander Ilic, Diogo Marques (INESC-ID & IST); Sujata Tibrewala (Intel)

Nicole: Excellent, Diogo, it's been so great to have you here. Thank you so much for diving into those technical details for us.

Diogo: Thanks, Nicole. I really enjoyed the discussion.

Nicole: And Aleksandar, thanks so much for being back on the program. Always great to talk to you.

Aleksander: Thanks, Nicole. It’s been a great pleasure to join this discussion today. Thanks.

Nicole: And a big thank you to all of our listeners for joining us today. Let's continue the conversation at oneapi.com.

Reference: https://www.sciencedirect.com/science/article/pii/S0167739X19309586