Workshop title and acronym

20th edition of the Workshop on Compiler-Driven Performance (CDP)

Organizers:

The workshops is co-chaired by:

- Christophe Dubach <christophe.dubach@mcgill.ca>, McGill University, Canada
- Kai Ting Wang (Amy) <kai.ting.wang@huawei.com>, Huawei, Canada

Brief Abstract and format: (250 words max)

This is the 20th edition of the Workshop on Compiler-Driven Performance (CDP) at CASCON. This workshop provides a unique opportunity for academic researchers, industry researchers, and developers from across Canada and the United States to examine the state-of-the-art compiler technology and discuss future directions for research and development.

The CDP workshop consists of presentations of reports on research progress at various academic and industrial sites across Canada and in the United States.

Developments in computing technology motivate a number of key challenges for compilers to address. Topics to be discussed in the workshop will include, but are not limited to:

- innovative compiler analysis, transformation, and optimization techniques
- languages, compilers, and optimization techniques for multicore processors and other parallel architectures
- analysis and compilation techniques for dynamic programming languages
- compiling for streaming or heterogeneous hardware
- dynamic compilation for high-performance and real-time environments
- compilation and optimization for scripting languages
- compilation techniques for reducing power
- program safety
- whole system optimization and analysis
- tools and infrastructure for compiler research
- intermediate representations for code
- compiler techniques for hardware accelerators
- leveraging AI and LLMs for code generation

Workshop theme, goals, relevance, and success criteria (500 words max):

Theme and goals:

Continued language and hardware development requires continued effort to identify new sources of potential optimization, and develop novel techniques for new contexts. The workshop on Compiler-Driven Performance allows researchers to share progress in developing new approaches to existing problems and in identifying new optimization opportunities and performance vectors for current and future languages.

The workshop has the additional benefit of both providing and demonstrating strong cooperation between academia and industry; compiler optimization research is a highly practical domain, but also one in which cutting-edge research techniques can have direct application, and work in both academic and industrial contexts is improved through this level of cooperation.

Relevance: The proposed workshop addresses many issues of key relevance to CASCON. Since the end of processor frequency scaling, compiler optimizations have been even more important. Varying programming models and dynamic runtimes have made it hard to understand system performance. Multicore processors and other parallel accelerators make it difficult to achieve both high programmer productivity and performance. The proposed workshop is well suited to address these challenges.

Success criteria:

- The workshop will be deemed a success if we can attract 25 participants, with a mix of industry and academic participants.
- We will also count as a success any collaboration that is started between participants, as a result of the workshop. We will conduct a survey after the workshop to obtain information about the potential future collaborations that have been triggered because of the event.

Structure (500 words max):

The proposed workshop is an all-day multiple-speaker workshop that will tentatively run from 9am until 4pm. The workshop will consist of eight to ten talks of 20–25 minutes each, with additional time for questions after each talk. Talks will be grouped, by topic, into sessions of two or three talks, with either a half-hour break or lunch between sessions to provide further time for discussion and interaction.

Each talk will be followed by at least a 5-minute question-and-answer period, allowing the audience to discuss the presentation with the speaker and each other. The two scheduled breaks and the lunch period will also give time for further interaction and more open discussion between attendees.

As in previous years, we will solicit presentations by opening up a call for title/abstracts, which will be reviewed by the co-chairs. The target of the call will be recent work which may or may not have been published before. We will advertise this call to all our colleagues working on compilation in North America.

The titles/abstracts will be made available on the workshop website.

Workshop duration: 1 day

Extended abstract

This will be the 20th edition of the Workshop on Compiler-Driven Performance (CDP) at CASCON. This workshop provides a unique opportunity for academic researchers, industry researchers, and developers from across Canada and the United States to examine the state-of-the-art compiler technology and discuss future directions for research and development.

Developments in computing technology motivate a number of key challenges for compilers to address. The workshop has a particular focus on the following:

1. Innovative Compiler Analysis, Transformation, and Optimization Techniques

Today's software systems are often composed of several different languages and programming models. At the same time, the underlying processors and memory systems are typically complex, out-of-order, superscalar processors. Managing this complexity, trying to produce efficient systems in a way that does not increase the burden on programmers, requires constant innovation in compilation methods and optimizations.

2. Languages, Compilers, and Optimization Techniques for Multicore Processors and Other Parallel Architectures

Highly-parallel hardware has created new opportunities for the utilization of such resources by advanced compilers. Progress has been reported both in the development of new analysis techniques, the exploitation of new technological solutions to facilitate the expression of concurrency and to deal with the needs for computation synchronization, e.g., hardware-supported transactional memory, and for data communication between computing domains. Approaches to improve the utilization of heterogeneous hardware platforms could also be discussed.

3. Compiling for Streaming or Heterogeneous Hardware

In addition to highly-parallel multiple-core processors for general-purpose and scientific computing, the computer hardware industry is also aggressively pursuing custom computing cores to accelerate key applications. Such heterogeneous computing systems were once limited to the embedded domain, but are becoming increasingly common for general-purpose computing. Examples include cores for encryption, compression, pattern-matching, and systems that have FPGA co-processors, graphics processing units (GPUs) and custom accelerators, which will likely soon be incorporated on-chip with regular processors. The resulting heterogeneous hardware presents another key challenge that the workshop target audience is working to address.

4. Dynamic Compilation for High-Performance and Real-Time Environments

Of ever-increasing importance are compilers that dynamically translate or optimize programs, not only to support interpreted languages such as Java, but also to exploit the run-time behaviour of programs written in C and C++ to improve efficiency and performance. Run-time adaptation can usually take advantage of more precise state information, allowing systems to present abstracted interfaces while ensuring an efficient implementation. Such layers of abstraction are important to allow programmers to efficiently target the emerging highly-parallel and heterogeneous hardware.

5. Compilation, Optimization, and Analysis for Dynamic Languages

The performance of scripting languages such as Python, Ruby, PHP, Javascript, and others, are of increasing importance to the overall performance of most online systems. These interpreted, and often dynamically-typed, languages pose many challenges in terms of optimization design and analysis, requiring highly dynamic and adaptive techniques to overcome the lack of static information, language idioms, and novel workloads found in different execution contexts.

6. Compilation Techniques for Reducing Power

Reducing power consumption is a key challenge for all computer systems, from mobile devices through high-end supercomputers. Compilers that can optimize for power or coordinate the power-reduction features of other parts of the system are of great interest. This extends to the compiler itself, incorporating power-friendly methods in the context of dynamic compilation.

7. Program Safety

The size and complexity of many modern software projects makes programming errors both difficult to find and easy to produce. Compiler approaches have shown potential to improve code safety by detecting common bugs ahead of time or by automatically trapping more subtle errors at runtime. Such techniques are likely to play an increasing role in software development, lead to many analysis and runtime optimization challenges, and represent an interesting further application domain for software analysis.

8. Whole System Optimization and Analysis

Many applications are in practice run in a non-trivial context, along with other activities or programs. Individual program behaviours and resource competition then affect overall system performance. Designs that assess complementary or competitive behaviours, or that dynamically adjust individual execution to improve global system usage extend program optimization and analysis techniques to higher-level execution goals.

9. Tools and Infrastructure for Compiler Research

The changing technology landscape highlights the need for ever-improving compiler-based tools and infrastructure for understanding programs and performing research. Development of new analysis techniques and optimizations is facilitated by basic program exploration, profiling, and visualization, looking for further sources of semantic meaning that can be applied to improve performance, language design, or toward other optimization goals.

10. Leveraging AI and LLMs for Code Generation

The application of LLMs has proven effective for code auto-completion and programmer assistance. However, when it comes to automatic code generation, the need for correctness makes LLMs uniquely challenging. Despite these hurdles, research in this area is advancing rapidly, for instance Meta’s LLM compiler FTD trained with billions of LLVM IR tokens. While early results are promising, further research is needed to improve the accuracy of results and to overcome the limited context length of LLMs in order to make the work commercially viable.

Continued language and hardware development requires continued effort to identify new sources of potential optimization, and develop novel techniques for new contexts. The workshop on Compiler-Driven Performance allows researchers to share progress in developing new approaches to existing problems and in identifying new optimization opportunities and performance vectors for current and future languages. The workshop has the additional benefit of both providing and demonstrating strong cooperation between academia and industry; compiler optimization research is a highly practical domain, but also one in which cutting-edge research techniques can have direct application, and work in both academic and industrial contexts is improved through this level of cooperation.