FreeCompilerCamp: Online Training for Extending Compilers

Alok Mishra  
Lawrence Livermore National Laboratory  
Livermore, California, USA  
Stony Brook University  
Stony Brook, New York, USA  
alok.mishra@stonybrook.edu

Anjia Wang  
Lawrence Livermore National Laboratory  
Livermore, California, USA  
University of North Carolina at Charlotte  
Charlotte, North Carolina, USA  
awang15@uncc.edu

Chunhua Liao  
Lawrence Livermore National Laboratory  
Livermore, California, USA  
liao6@llnl.gov

Yonghong Yan  
University of North Carolina at Charlotte  
Charlotte, North Carolina, USA  
yyan7@uncc.edu

Barbara Chapman  
Stony Brook University  
Stony Brook, New York, USA  
Brookhaven National Laboratory  
Upton, New York, USA  
barbara.chapman@stonybrook.edu

ABSTRACT

In this presentation, we introduce an ongoing effort of an online training platform aimed to automate the training of developers to quickly extend compilers – FreeCompilerCamp.org. Our free and open platform allows anyone who is interested in developing compilers to learn the necessary skills. A live training website, built on top of Play-With-Docker, is set up so that anyone with internet access and a web browser will be able to take this training. The entire training system is open-source and developers with relevant skills can contribute new tutorials and deploy it on a private server, workstation or even laptop. We have created some initial tutorials on how to extend the Clang/LLVM or ROSE compilers to support new OpenMP features. Using a web interface consisting of two side-by-side panels, users can follow the tutorials on one side and immediately practice what they learned in a terminal sandbox embedded on the other.

1 EXTENDED ABSTRACT

In the development landscape, new languages and improvements on existing languages are escalating day by day. These languages require new tools for building – especially compilers. Compilers are the backbone of software development process. They are the programs that processes the statements written in human readable language, like C/C++, and transform them into machine code. LLVM [3] is one such collection of modular and reusable compiler and toolchain technologies. LLVM makes it easier to not only create new languages, but to enhance the development of existing ones. Its primary C compiler is Clang.

However, the entire process of this transformation is not simple. A compiler need to parse the code, check for syntax correctness, gather necessary semantic information (like type checking or variable declaration before use), then convert the source from high level language to intermediate representation and perform several optimization before transforming them into machine codes [1]. Compilers themselves have very complex design so that the work of an application developer becomes simpler. Owing to the complexity of design, extending a compiler to add new feature is a very time consuming job.

Let us take an example of OpenMP, the de-facto portable programming interface for exploiting node-level parallelism [2]. More and more researchers and developers are interested in designing various extensions to OpenMP in order to tame the increasing complexity of heterogeneous node designs in high performance computing. Such extensions could be used to enhance the expressiveness, performance or productivity of OpenMP. But a major challenge is the complexity of implementing them in OpenMP compilers. It is difficult for a beginner to self-teach how to modify OpenMP compilers with millions of lines of code. Relying on expert developers to train beginners consumes lots of human efforts, which is not scalable in a long run.

Although there are several cloud-based tools, like CloudLab [5] or MOOC [9], that have been leveraged for computer science education in many cases, there is a clear lack of such tools to teach compiler development. Many such tools and resources are available across several domains of computation, but compiler development is still devoid of such online tools.

There are several major pain points for compiler training. Users may need to go through some paperwork to access suitable machines with complex configuration. Depending on the specific subjects, different machines and experienced instructors are required. The traditional text-only tutorials are not very effective. To solve the problems, a free online tool based on Docker for the training could help. It’s flexible to fit user’s need with lower cost. It can be easily used in a browser without too much help from the instructors. Hands-on tutorials can be created to let users learn by practice, which is more efficient.

In this presentation, we introduce an ongoing effort, FreeCompilerCamp.org, a free and open online training platform aimed to automate the training of researchers and developers to quickly extend compilers and help them learn the skills of compiler development. FreeCompilerCamp.org is a training system with several distinct design principals:

- It provides pre-configured compiler development environments in an online sandbox, which eliminates the burden for beginners’ tedious and error-prone software installation processes.
- It allows anyone who is interested in understanding the internals of OpenMP compilers to learn the necessary skills for free.
- A live training website based on the system is set up, so a web browser and an Internet connection are the only requirements for anyone to get the training.
It enables anyone who has the relevant skills to contribute new tutorials.

The entire training system is open-source, so it can be also be deployed by anyone on a private server, workstation or even personal laptop.

Currently, FreeCompilerCamp.org is built on top of Play-With-Docker [6], a docker playground for users to conduct experiments in a sandbox. We have created some initial tutorials to train users to learn how to extend the Clang/LLVM or ROSE [8] compilers to support new OpenMP features.

FreeCompilerCamp.org provides a browser-based interactive interface with two panels, which is easy for user to learn and practice effectively. The left panel contains the tutorial in text, while the right panel contains a live terminal sandbox for real-time practice. This will help the user focus on learning the skills of compiler and OpenMP development, and teach themselves in their own convenient place, at their own comfortable pace.

Figure 1 shows one example of such a tutorial which helps a developer learn how to introduce a new directive in the compiler. This tutorial explains how to add a new language feature support in OpenMP, by extending the Clang/LLVM compiler. Here the goal is to add a new OpenMP directive metadirective (#pragma omp metadirective [clause[,clause]...]), defined in OpenMP Specification 5.0 [7], that can specify multiple directive variants of which one may be conditionally selected to replace the metadirective based on the enclosing OpenMP context. The figure shows a part of the tutorial where the compiler first identifies the new directive. For this the developer need to declare the new directive in the file OpenMPKinds.def and add parsing information for the directive in ParseOpenMP.cpp. The next part of the tutorial goes on to explain how to identify the different clauses of metadirective and how to implement it.

These tutorials gives step-by-step instructions to the developers on what changes to make in the source code to support the aforementioned extensions. The code snippets are shown in two distinct ways in the training environment:

1. Code that appears in a black colored window can be clicked on and it will automatically be typed in to the appropriate terminal window. These are usually command line instructions, like which file to open for editing, or which command to run for building, etc.

2. Code appearing in gray colored window is code that a developer should type in themselves. These are usually those snippets that the developer will update in the existing code. The result is available immediately. Developers can follow the simple steps as given in the tutorial, or can dig deeper into the code or try something else for their understanding by experimenting in the live terminal on the right panel. Similar tutorials are provided on how to extend the ROSE compiler, or how to add a new Clang plugin, or how to use the AutoPar [4] tool in ROSE, etc. More such tutorials will be added in future.

FreeCompilerCamp.org is portable and can be self-hosted on any machine. Instructors or students can make customization easily and deploy it on any local server or even their own laptop. For instance, after inserting proper tutorials and modifying appropriate docker image, FreeCompilerCamp.org can be extended and deployed to teach compiler fundamentals in a classroom environment as well.

We welcome anyone to try out our system, give us feedback, contribute new training courses, or enhance the training platform to make it an effective learning resource for the OpenMP community.

2 ACKNOWLEDGEMENT

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344, and partially supported by the U.S. Dept. of Energy, Office of Science, ASCR SC-21), under contract DE-AC02-06CH11357, IM Release Number - LLNL-ABS-784872. This material is also based upon work supported by the National Science Foundation under Grant No. 1833332, 1652732, and 1409946. This research was also supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.
REFERENCES


