GSoC23-LLVM

This is a summary of the work done in GSoC'2023 under LLVM.

- Name: Krishna Narayanan
- Mentors: Vassil Vassilev, David Lange
- Organization: LLVM
- Project Title: Tutorial development with Clang-Repl
- Proposal Page: https://bit.ly/3ONWYNX
- Project Page: https://summerofcode.withgoogle.com/programs/2023/projects/qxuEqL8W

Contributions:

This is the list of contributions according to the pull requests I have sent:

1. Update Clang-Repl Documentation

- https://reviews.llvm.org/D152109
- https://reviews.llvm.org/D156858

This patch adds support for Clang-Repl documentation, basically, it provides all details regarding the usage of Clang-Repl. It emphasises the features of Clang-Repl that it offers to the user. The REPL nature enables users to prototype, experiment and gives a user-friendly experience. Similarly, the second patch adds Clang-Repl Execution Handling Results by Saqib in which we enabled the graphviz extension for the llvm/clang documents to support the graphviz convention for pictorial representation (under review).

2. Add C++ InterOp Documentation Setup

- https://github.com/compiler-research/CppInterOp/pull/87
- https://github.com/compiler-research/CppInterOp/pull/99
- https://github.com/compiler-research/CppInterOp/pull/105
- https://github.com/compiler-research/CppInterOp/pull/121
- https://github.com/compiler-research/CppInterOp/pull/134

CppInterOp is a clang-based C++ Interoperability library, which enables interoperability with C++ code to more interactive languages like Python. The above patch adds documentation setup for CppInterOp consisting of both sphinx and doxygen documentations. The above patches have covered all topics, points catering to the development and usage of CppInterOp which include building from source(installation) , usage, FAQs, developer’s documentation, tutorials and references. The tutorials in the

docs give a detailed understanding of C++ InterOp usage, which includes C-C++ interoperability and C++-python interoperability.

3. Add xeus-clang-repl Documentation Setup

- https://github.com/compiler-research/xeus-clang-repl/pull/33
- https://github.com/compiler-research/xeus-clang-repl/pull/34
- https://github.com/compiler-research/xeus-clang-repl/pull/35

Xeus-clang-repl integrates clang-repl with the xeus protocol and is a platform for C++ usage in Jupyter Notebooks. The above patch adds a documentation setup for xeus-clang-repl consisting of both sphinx and doxygen documentations. The documentation covers all information regarding installation, usage, importance and references for xeus-clang-repl. It includes tutorials portraying the different features that can be used in xeus-clang-repl, especially the C++-python integration executing simultaneously in the Jupyter cell with the help of magic commands(%%python).

4. Add xeus-cpp Documentation Setup

- https://github.com/compiler-research/xeus-cpp/pull/13

Xeus-cpp is an interactive programming environment that allows you to execute C++ code in a Jupyter Notebook. The above patch adds a documentation setup for xeus-cpp consisting of both sphinx and doxygen documentations.The documentation covers all information regarding installation, usage, importance and references for xeus-cpp. It includes tutorials portraying the different features that can be used in xeus-cpp, especially the C++-python integration executing simultaneously in the Jupyter cell with the help of magic commands(%%python).

5. Others

- https://github.com/compiler-research/xeus-cpp/pull/12
- https://github.com/vgvassilev/clad/pull/610

These are miscellaneous patches contributing to the development of current documentation setup and content. It includes migration from the v1 to the v2 configuration for readthedocs setup.

Conclusion

All the goals that were originally proposed have been completed to the best of my abilities (the xeus-cpp setup has not been merged yet). I will be working on improving these things after suggestions in the upcoming weeks to make tutorials and documentation more understandable for users. I will keep working with the compiler research group even after the GSoC period, contributing to tutorials and landing other development patches.

I am extremely grateful to my mentors, Vassil and David, for their constant help and support in the last two months. Many thanks to Vassil for providing help and reviewing the code promptly and guiding me towards the final goal, special thanks to Parth and Baidyanath for guiding me during the initial phase of GSoC. The journey has been quite full of learning, experiencing new techstack and realising the importance of documentation and tutorials for a better user experience. I thank Vassil and all LLVM community members for giving me this great opportunity to work with such an evergreen and interesting community.

Finally, thanking everyone at the compiler research group for assisting throughout the GSoC period with many new concepts and help I needed. I am also thankful to Google for providing me the opportunity to work on this project during the summer, which helped me learn a lot and will also surely help in my career in the future.

Project Proposal:

https://github.com/compiler-research/compiler-research.github.io/blob/master/assets/docs/Krishna_Proposal_2023.pdf