

ACTS

Project Summary

Ever since the 1950s, following the appearance of work by Backus, Chomsky and others, automatic program compilation (and its related task of interpreted program execution) has been a vital ingredient in the creation of ever more complex software tools. Indeed, modern compilers and interpreters have been central to exploiting exponential advances in digital hardware capability. Without compilers and interpreters to automate the task of correctly transforming a high-level language description of an algorithm, modern computer systems could barely exist. Certainly, the wealth of scientific and engineering computational applications available today owe their existence to automation of major parts of the development process by compilers and interpreters. Enterprises as diverse as econometric forecasting, medical ultrasound diagnosis, chemical plant design and operation and climate and weather analysis and monitoring have been transformed by the advent of compilation technology.

With modern computational capabilities, workers in numerous fields use large-scale numerical simulation of complex interacting elements to understand physical, chemical, biological and economical phenomena. An increasing number of highly relevant scientific and engineering problems are modeled using sophisticated state-of-the-art mathematical techniques. Computer programs are being developed using high-level programming languages to simulate the behavior of these models numerically. Often these programs consist of several hundred thousand lines of code. Information on first and higher order derivatives of the model outputs with respect to certain input parameters is required for a better understanding of the problem itself and to perform certain optimizations for improving the numerical model.

The Adjoint Compiler Technology & Standards (ACTS) project aims to provide scientists, engineers, academics, and students with techniques, tools, reference projects, and easily accessible expertise allowing for new approaches to investigating numerical models of real-world problems based on mathematically precise derivative information. Considering the large-scale applications motivating the ACTS project there is not sensible alternative to Automatic Differentiation (AD) as the methods of choice. AD transforms the program implementing the original model into derivative code by augmenting it with instructions for computing partial derivatives. For example, the investigation of numerical sensitivities of certain values with respect to certain parameters becomes possible. Where observations are to be combined with model output so as to account for the information content of both, one has a very large-scale optimization problem. The solution to this problem lies with the ability to generate derivative code as well.

Because of the sheer size of the application programs that AD has to deal with efficiency of the generated derivative code is crucial. Inefficiency of the codes generated using AD will often mean inapplicability caused by infeasible hardware requirements. This project has potential to impact fields ranging from engineering design optimization projects to basic research to complex physical system monitoring endeavors. The proposal brings together computer scientists, physical scientists and engineers in a group effort to create a standard, reference implementation of an extensible toolkit for AD of numerical codes. Such a toolkit is a key asset in improving the fidelity and productivity of increasingly complex simulation code development efforts in many fields. In summary the goals of this project are three fold

1. development of an extensible, standards based, infrastructure that will allow research in the field of AD to flourish and that will allow academic developments and emerging commercial efforts to collaborate in a mutually advantageous manner;

2. development of an open-source, freely available reference compiler and support material that is as easy and reliable to use as a traditional compiler and that will nurture widespread adoption of AD as an key element of all manner of numerical simulation;

3. establishment of clear and firm connections between the mainstream compiler communities in computer science and the mathematics, physical science and engineering communities that have historically been at the forefront of AD tool development.

For mathematical, scientific and engineering codes, the work proposed can have far reaching consequences. Just as conventional compiler developments have provided the foundation for new generations of simulation tools the project proposed here will also provide a foundation for new generations of mathematically rigorous and robustly manufactured analysis, diagnostic and forecast software.