A Clustering-based Approach for Exploring Sequences of Compiler Optimizations

Luiz Martins¹,², Ricardo Nobre³, Tiago Carvalho³, Eduardo Marques², João Cardoso³

¹ Universidade Federal de Uberlândia (UFU), Faculdade de Computação (FACOM), Brazil
² Universidade de São Paulo (USP), Inst. de Ciências Matemáticas e de Computação (ICMC), Brazil
³ Universidade do Porto (UP), Faculdade de Engenharia (FEUP)/INESC-TEC, Portugal

gustavo@facom.ufu.br, ricardo.nobre@fe.up.pt, tdrc@fe.up.pt, emarques@icmc.usp.br, jmpc@acm.org

Abstract:

In this poster we present a clustering-based approach for reducing the number of compilation engines used in design space during the optimization sequences exploration. The basic idea is to identify similarities between functions and to use the engines previously explored for each function. The identification is performed from a symbolic encoding, named DNA of the program, and using three different techniques: Normalized Compression Distance (NCD), Neighbor Joining (NJ) and Strong Ambiguity-based Clustering (SAC). Main structures of the source code are translated to sequences of symbols (DNA of the program) based on transformation rules. NCD calculates a matrix of distances similarities. NJ constructs a tree representing possible relationships among program DNAs. SAC extracts potential clusters hidden in the tree topology. Two design space exploration schemes were used aiming at investigate sequences of optimizations targeting a Xilinx MicroBlaze processor for 41 functions from Texas Instruments benchmarks. The first one chooses the best of 300 sequences sampled randomly from the design space reduced by our clustering-based engines selection. The other scheme is based on Genetic Algorithms (GA) and does not adopt the previous engines selection. The results show that the usage of our approach allows achieves a significant gain on execution time (around 14× over GA) still achieving significant performance improvements in DSE schemes (36% against 44%).
A Clustering-based Approach for Exploring Sequences of Compiler Optimizations

Luiz Martins1, Ricardo Nobre2, Tiago Carvalho3, Eduardo Marques2, João Cardoso1

1Univ. Federal de Uberlandia, FACOM, Brazil 2Univ. de Sao Paulo, ICMC, Brazil 3Univ. do Porto, Fac. de Engenharia (FEUP)/INESC-TEC, Portugal

gustavo@facom.ufu.br, ricardo.nobre@fe.up.pt, tdrc@fe.up.pt, e.marques@icmc.usp.br, jmpc@acm.org

Motivation and Approach

1. Developers often target heterogeneous (e.g. CPU + FPGA and/or GPU) or multiple Hardware/Software platforms. Each component or platform is dependent on a set of specific mapping tools.
2. Developers typically have to engage in source code modification and test multiple compilation sequences to achieve satisfactory design solutions for a specific target platform.
3. The quantity of engines provided by the compiler tool and the length allowed for the optimization sequence affect considerably the design space explored.
4. Our approach uses LARA, an aspect-oriented language, in order to control a design space exploration (DSE) process relying on a toolchain responsible for code transformation, target-specific compilation, and simulation.
5. A clustering-based engines selection is applied reducing the number of engines used and reducing the execution time of the DSE.

Clustering Process

The LARA-based design-flow consists of a set of integrated compilation and synthesis tools which are controlled through LARA aspects. An integrated and unified view of the design-flow provided by the LARA approach allows to program and apply easily different DSE strategies (SA – Genetic Algorithms, Simulated Annealing, etc.).

The CoSy compiler engine is integrated in the Reflect compiler, which includes CoSy based VHDL (OWHDL), Microblaze and other code generators.

New Function

Function 1

Function 2

Function 3

Case 1

Case 2

Case 3

Clustering-Based Engines Selection

Experimental Results

Exploring sequences of compiler optimizations targeting a Xilinx MicroBlaze processor for 41 functions from Texas Instruments benchmarks.

Conclusions

✓ LARA effectively enables design space exploration by providing control over the components of a toolchain and allows the evaluation of design alternatives based on feedback information reported by different stages of the design-flow.
✓ LARA aspects are translated to an IR that is toolchain- and target-independent.
✓ Clustering-based compiler engines selection achieves a significant gain on DSE execution time still achieving significant performance improvements.
✓ Ongoing work is focusing on researching heuristics that explore each clustering-based set of engines and on exploring sequences of compiler optimizations in the context of hardware compilers (targeting FPGAs).

Weaving Process in CoSy

A CoSy** based toolchain built around LARA abstracts the developer from the interaction with all used mapping tools while effectively guiding the design flow process in the exploration of alternative designs targeting the chosen architecture. A CoSy weaver engine executes other CoSy engines according to the optimization actions described in LARA aspects, allowing DSE to control and explore sequences of compiler engines from more than a hundred engines provided in the CoSy distribution.

References

DNA: Clustering-based compiler engines selection

DNA: Design Exploration (using CoSy toolchain)

DNA: Clustering Evaluation (using CoSy toolchain)