

Combined Source-Level Transformations and Instruction Set Extension



Institute for Computing Systems Architecture
University of Edinburgh

Alastair Murray

Richard Bennett

Björn Franke

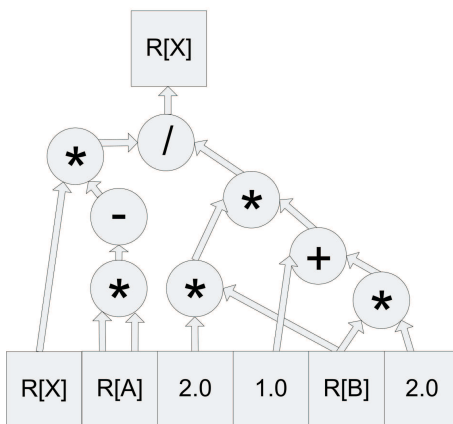
Nigel Topham

Abstract

The standard design flow when using automated instruction set extension (ISE) tools is to use the plain C source files as input into the ISE tool. The output is then a description of new instructions and modified source code that explicitly uses these new instructions where appropriate. The compiler can then optimise this modified code.

Our approach is to transform the code prior to using an ISE tool to allow better custom instructions to be found and to allow compiler transformations and ISE to be treated as a single design space. This combined space is sampled by selecting sequences of transformations to apply from a set of 70 and it is shown that this novel, integrated approach outperforms existing techniques of ISE generation.

Motivating Example

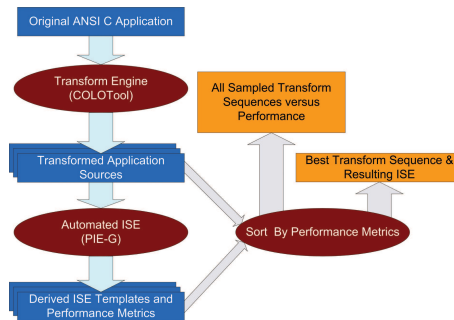


$$\text{diff} = (\text{diff} * \text{-(rad * rad)}) / ((2.0 * \text{inc}) * (2.0 * \text{inc} + 1.0))$$

The above instruction implements the shown code excerpt from the SNU-RT FFT benchmark. Each use of the instruction saves 13 cycles of execution time and reduces code size by 7 and it may be implemented on a 3-input, 1 output custom functional unit.

Methodology

Our methodology uses a probabilistic search algorithm and a source-level transformation tool to generate many different – but semantically equivalent – versions of the input program. We present each generated version to our ISE tool, which creates a set of new instructions and profile-based estimates of the speed-ups gained by using these instructions.

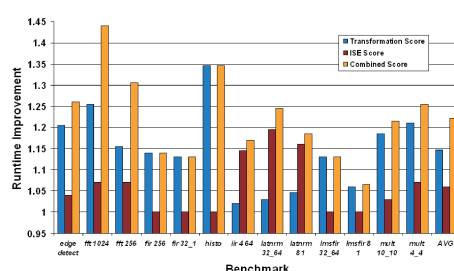


The search space of potential transformations is sampled with uniform probability, at random points across the entire space. A sample in this sense is an ordered set of transformations and 10000 samples are taken per benchmark.

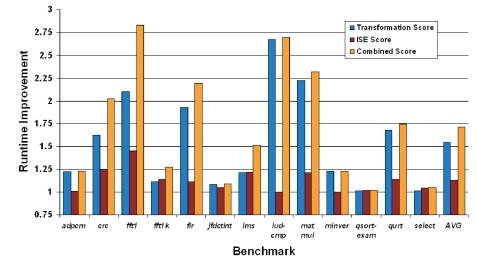
Each sample is then processed by an automated profiling ISE tool based on the Atasu et al. Integer Linear Programming method. During the final extension phase the top four most beneficial instructions are taken and estimated speed-ups are calculated based on a profile.

Results

UTDSP Speed-ups:



SNU-RT Speed-ups:



Average speed-up achieved by the combined technique is 1.47x across both benchmark suites, with a peak of 2.85x seen for SNU-RT FFT.

Conclusions

We have described a methodology for improved ISE generation that combines the exploration of high-level source transformations and low-level ISE identification. We have demonstrated that source-to-source transformations are not only very effective on their own, but provide much larger scope for performance improvement through ISE generation than any other isolated low-level technique.

We have integrated both source-level transformations and ISE generation in a unified framework that can efficiently optimise both hardware and software design spaces for extensible processors.

Further Information

For further information please refer to our LCTES '07 paper, a reference to which may be found in the poster abstract.

Our institute web site can be found at: <http://www.icsa.inf.ed.ac.uk/>

Email: a.c.murray@sms.ed.ac.uk

r.v.bennett@sms.ed.ac.uk

bfranke@inf.ed.ac.uk

npt@inf.ed.ac.uk