Fixed point Precision Tuning via Compiler Analyses & Transformations

IWES 2019

Stefano Cherubin, D. Cattaneo, M. Chiari, G. Agosta

30 September 2019

Fixed Point Representations

Usually exploited when floating point unit is not available.

Programmable hardware can implement fixed point computation using custom width. We consider the general case of **x86-like** architecture.

- Keep fixed representation width (8, 16, 32, 64 bits)

Coding in Fixed Point

- The *Embedded C* language allows programmers to use native fixed point data types.
- What about other programming languages?
  - Manual conversion of floating point code
  - Fixed point ADT
  - Source to Source compilers

NO EASY WAY(?)

Coding in Fixed Point

- The *Embedded C* language allows programmers to use native fixed point data types.
- What about other programming languages?
  - Manual conversion of floating point code
  - Fixed point ADT
  - Source to Source compilers

NO EASY WAY(?)

Compiler Transformation Approach

Perform analysis and code conversion within the compiler

- Provides the same features of the S2S compiler
- Static code analysis instead of dynamic profiling

Let the Compiler Do the Work

- Programmer specifies ranges of input values
- Compiler propagates ranges to intermediate values
- Compiler statically analyze the code
- Compiler automatically selects the best data type
- Compiler performs code conversion
- Compiler provides optimized code

The Rise of TAFFO

Tuning
Assistant for
Floating point to
Fixed point
Optimization

open source on github
`https://github.com/HEAPLab/TAFFO`

TAFFO Infrastructure

Based on the LLVM compiler toolchain

- Annotations are natively supported by *clang*
  - Applies to a large variety of programming languages
- Packaged as self-contained clang plug-in
- *clang* can be used instead of the system compiler
  - or can be easily integrated with the target toolchain

Annotation Example

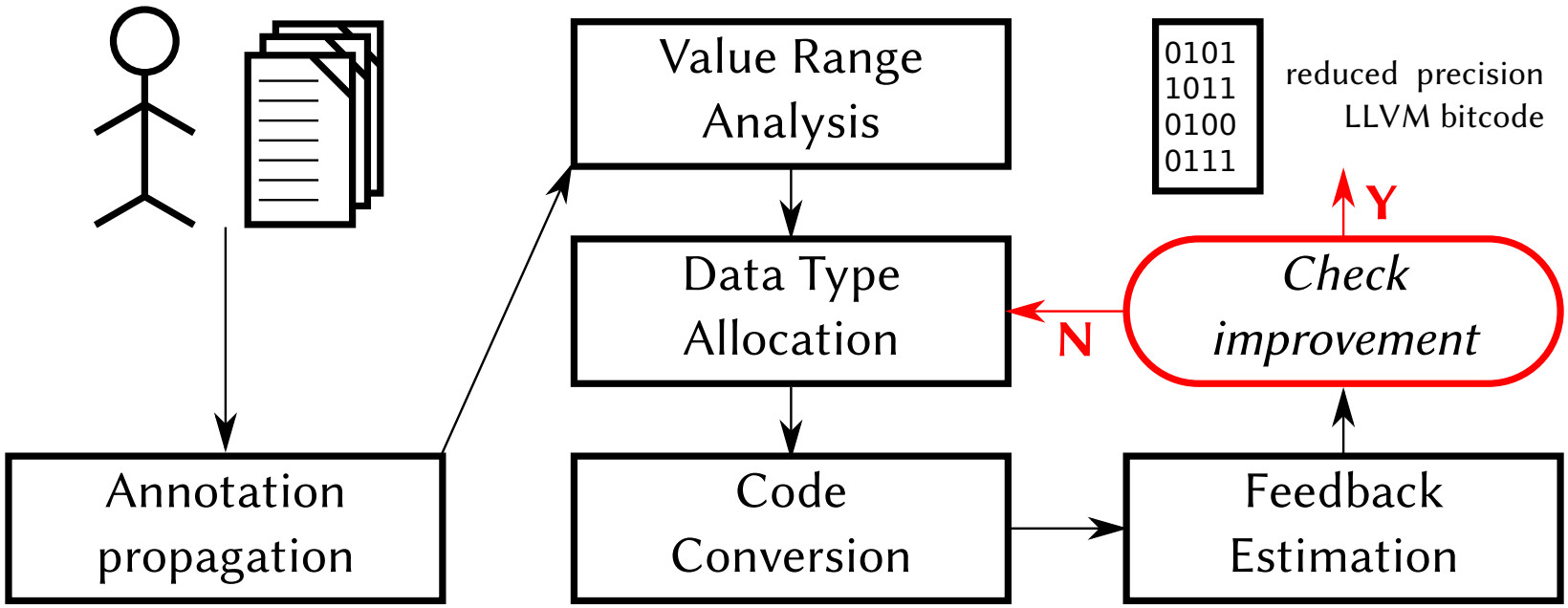
Variables and computations which are converted to fixed point.
Connections between them highlight the operations affected by the conversion.

It is possible to specify the size of the integer and fractional part of the representation.

```
float a __attribute((annotate("force_no_float")));
int b = 98;
a = b * 2.0;
a += 10.0;
float c __attribute((annotate("no_float 12 20")));
c = function1(b) + 3.5;
a += c * 2.0;
```

The `no_float` and `force_no_float` annotations indicate which variables have to be converted to fixed point. The conversion propagates to related computations in different ways: `force_no_float` entails the conversion of the dependencies, while `no_float` propagates only to intermediate values.

Component Schema



Benefits

- Relies only on the well-known LLVM framework
- Does not require any customization of the compiler
- Source language agnostic
- Easy to maintain

Key Features

- Support all C-like languages
- Target all LLVM back-end architectures
- Inter-procedural optimization
- Can be combined with other compiler optimization
- Active support and maintenance

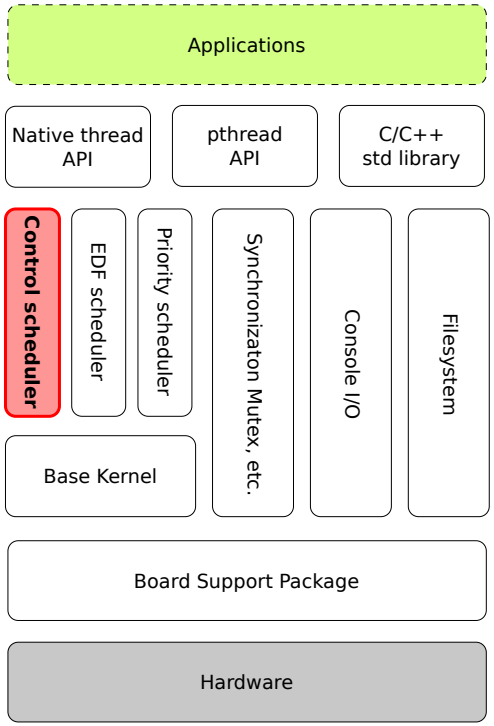
Success Stories

- PolyBench/C benchmark suite
- Miosix scheduler
- HPC approximate computing applications
- WIP on dynamic precision tuning for HPC
- ...

We are open to new challenges!

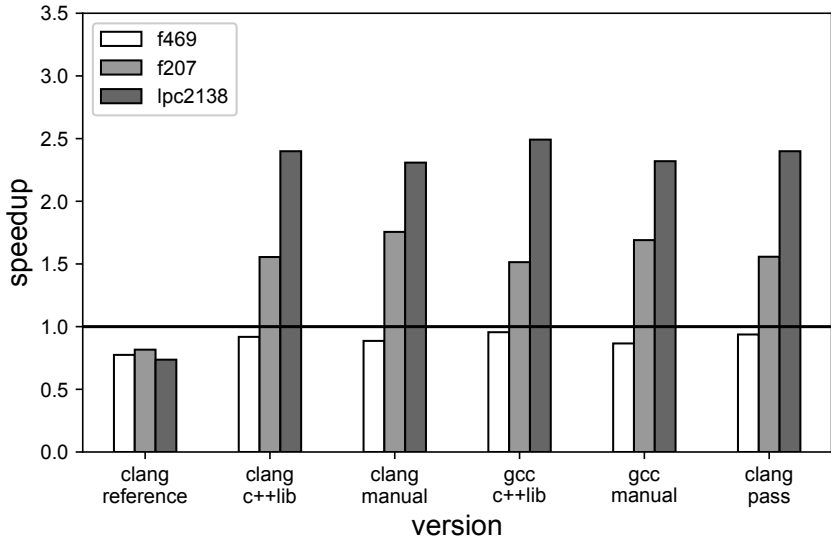
Real Time OS

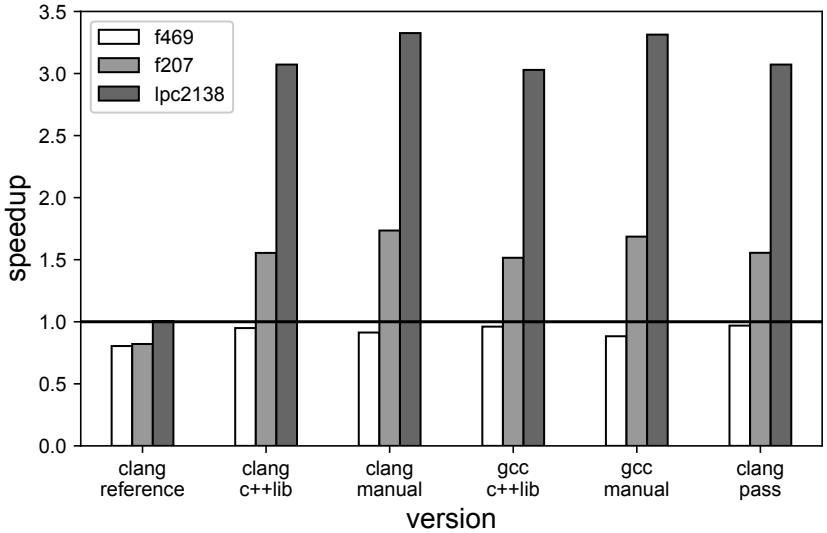
We applied our tool to the Miosix embedded operating system



- Convert the Control-Theoretical scheduler from floating point to fixed point
- Test on scheduling time of real-time benchmarks
- Short paper at Euromicro DSD 2018

Miosix results





f469 STM32F469I-DISCO – ARM Cortex M4 (YES HW FPU)

f207 STM3220G-EVAL – ARM Cortex M3 (NO HW FPU)

lpc2138 custom development board – ARM7TDMI (NO HW FPU)

Lesson learnt

- Fixed point may improve performance in several architectures – from HPC to microcontrollers
- TAFFO proved to be helpful to deal with large code bases
- Help us to feed our pet!
  - We are looking for new interesting use cases

Lesson learnt

- Fixed point may improve performance in several architectures – from HPC to microcontrollers
- TAFFO proved to be helpful to deal with large code bases
- Help us to feed our pet!
  - We are looking for new interesting use cases

Questions?

?

Thanks for your attention!

stefano.cherubin@polimi.it
daniele.cattaneo@polimi.it
michele.chiari@polimi.it
giovanni.agosta@polimi.it
heaplab.deib.polimi.it