ICD-C Demo

Daniel Cordes
TU Dortmund University,
Department of Computer Sience XII

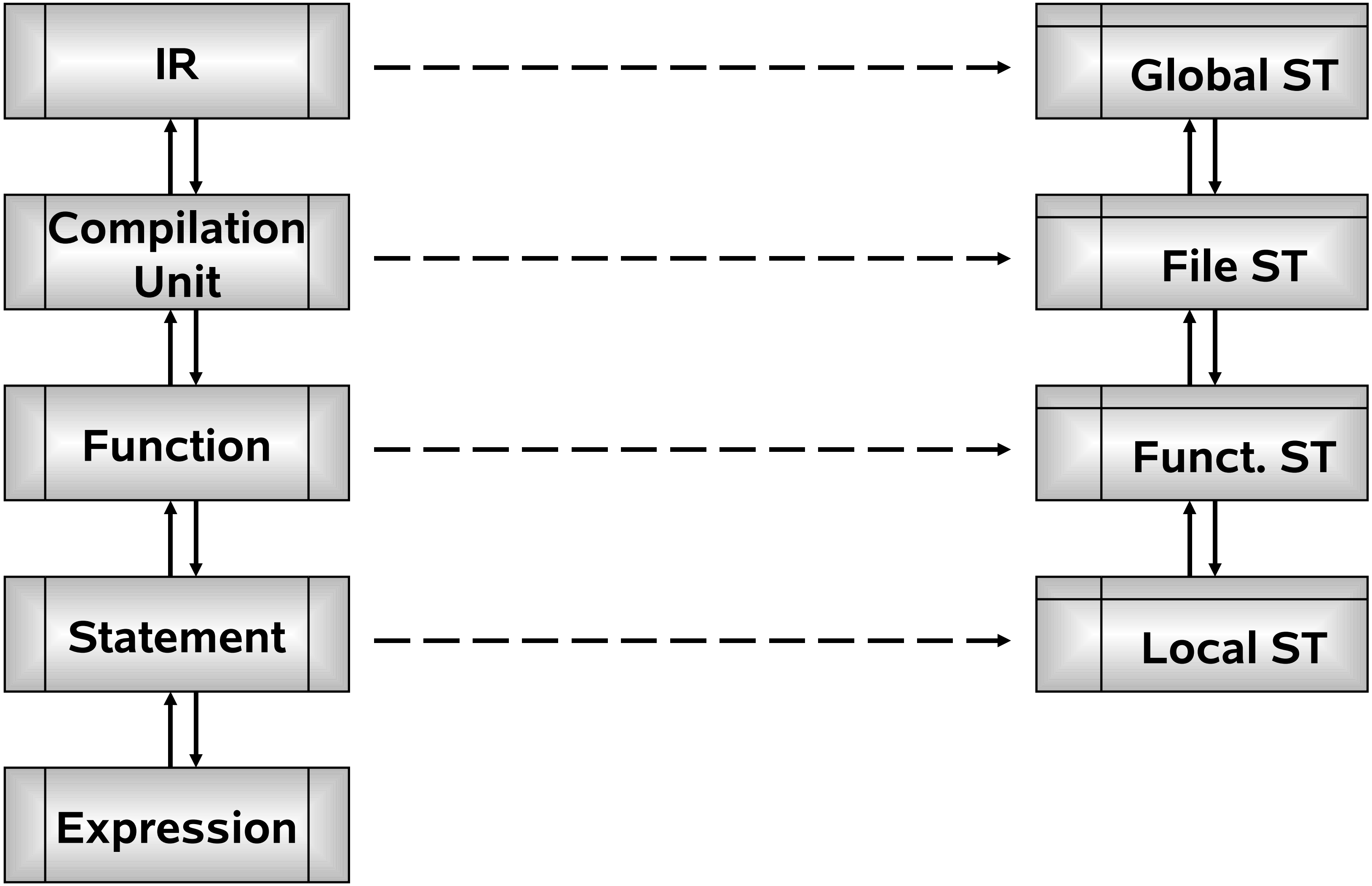
2008/11/27

ICD-C Overview

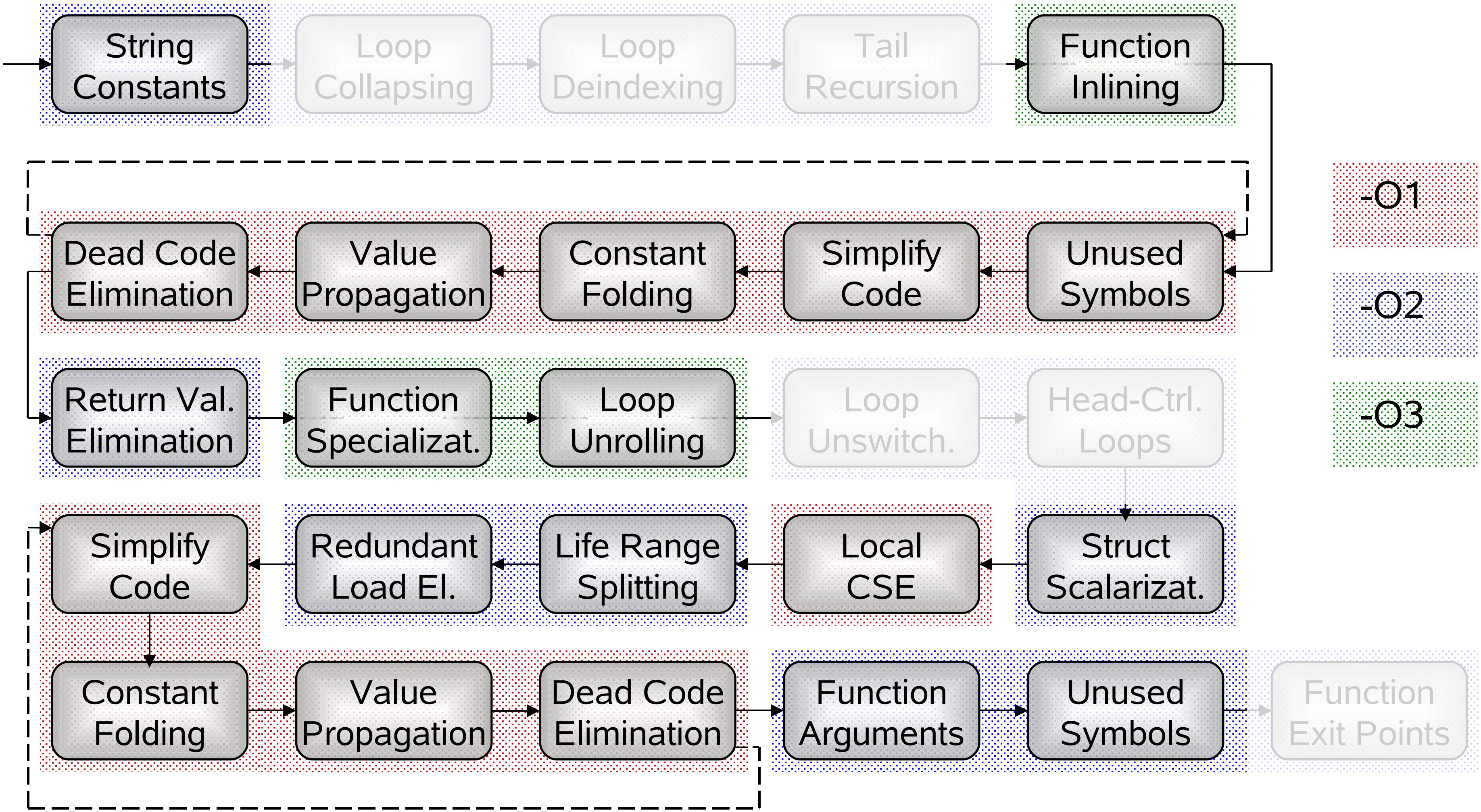
- ANSI-C Compiler Frontend
  - C89 + C99 Standards
- Configurable for multiple platforms
  - i.e. Bitwidth for Char or Integer types
- Object Orientated Implementation
  - Convenient to use
- Powerful Analyses & Optimizations
  - Control Flow, Data Flow & Other Analyses
  - > 20 Built-In Standard Optimizations
- Flexible Usage
  - High-Level IR* can be combined with every Compiler
  - Interesting i.e. for Source-to-Source Optimizations

(*) *IR = Intermediate Representation*

ICD-C Class Hierarchy



ICD-C Optimizations



String Constants

Loop Collapsing

Loop Deindexing

Tail Recursion

Function Inlining

Unused Symbols

Simplify Code

Constant Folding

Value Propagation

Dead Code Elimination

Return Val. Elimination

Function Specializat.

Loop Unrolling

Loop Unswitch.

Head-Ctrl. Loops

Struct Scalarizat.

Local CSE

Life Range Splitting

Redundant Load El.

Simplify Code

Constant Folding

Value Propagation

Dead Code Elimination

Function Arguments

Unused Symbols

Function Exit Points

-O1

-O2

-O3

Use Case 1: Loop Analyzer

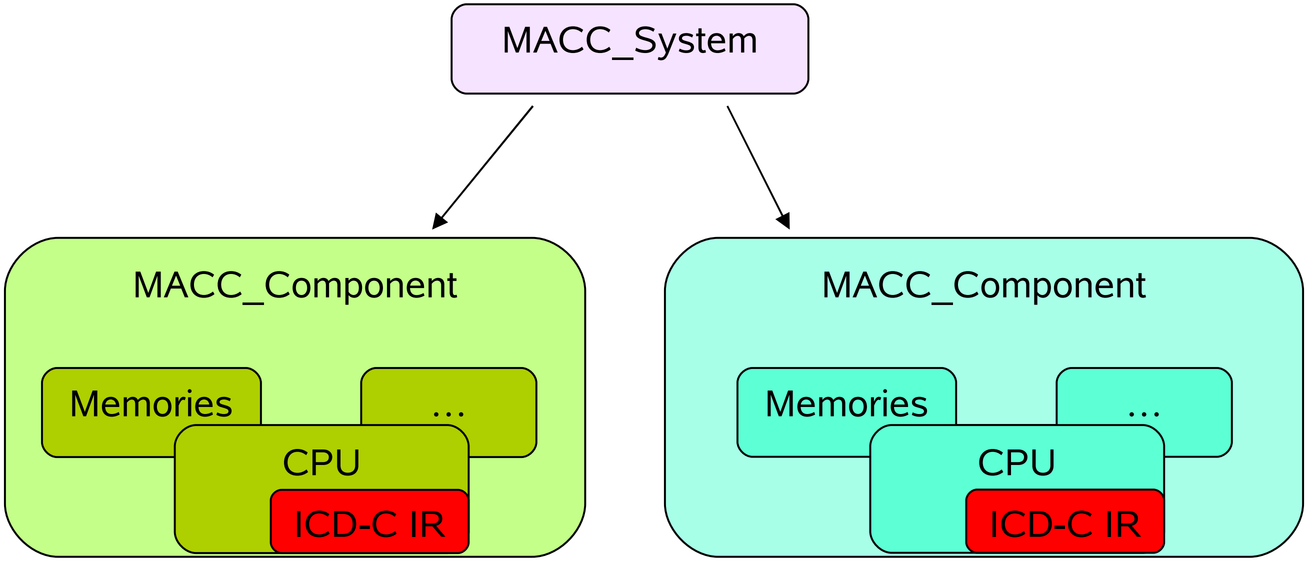
- Complex Loop Analyzer
  - Techniques of Abstract Interpretation
  - Combined with Polyhedra Models
  - Based on ICD-C
  - Integrated into WCC & Stand-Alone Tool
  - Paper accepted at CGO 2009

“A Fast and Precise Static Loop Analysis based on Abstract Interpretation, Program Slicing and Polytope Models“

- Loop Bounds necessary for loop parallelization
  - No need for profiling / user annotations

Use Case 2: MACC

- Library to describe system architectures
  - Also communication + memory hierarchy
  - Every MACC CPU has ICD-C IR
  - Full access to all optimizations / analysis
  - Interesting for parallelization → MNEMEE project



Why is ICD-C framework useful for parallelization?

- Control Flow Analyses
  - Basic Block & Statement Level
  - Domination / Post-Domination
  - Reachability Analysis
  - Function Call Graph Analysis

- Data Flow Analyses
  - Def / Use Analysis
  - Def & Use Chains
  - Static Alias Analysis

Why is ICD-C framework useful for parallelization?

- Other useful Analyses
  - Automatic Loop (bound) Analysis
  - …
- Many Optimizations available
  - Optimizations to clean up code
  - Optimizations to reduce program run time

End of presentation

Questions?

Demo available at:

http://www.icd.de/es/icd-c/

Start of demonstration

Thanks to Dr. H. Falk
for some slides