Optimal Register Allocation and Instruction Scheduling for LLVM

Roberto Castañeda Lozano – SICS, KTH

joint work with:

G. Hjort Blindell – KTH, SICS

M. Carlsson – SICS

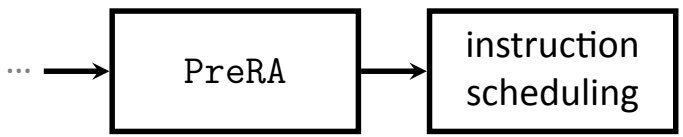
F. Drejhammar – SICS

C. Schulte – KTH, SICS

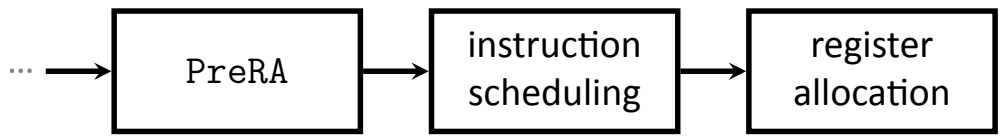
This research has been partially funded by LM Ericsson AB and the Swedish Research Council (VR 621-2011-6229)

Code Generation in LLVM

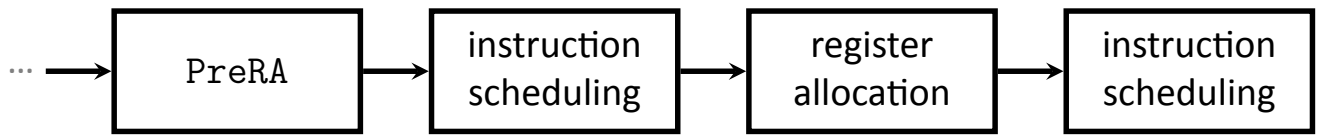
Code Generation in LLVM



Code Generation in LLVM

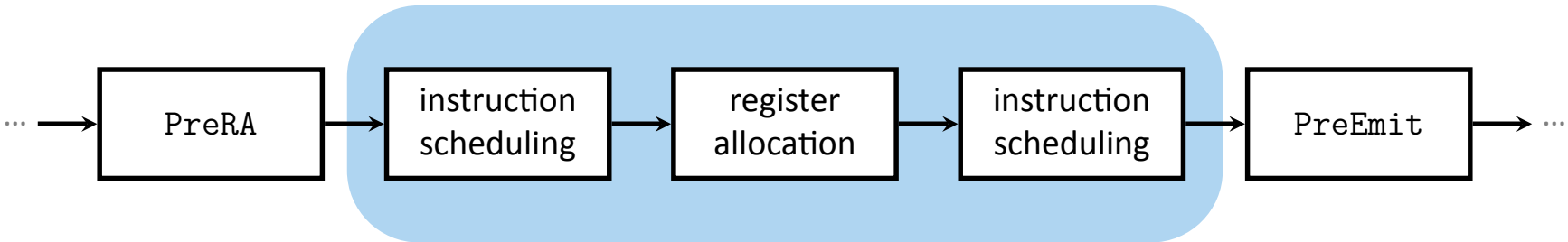


Code Generation in LLVM

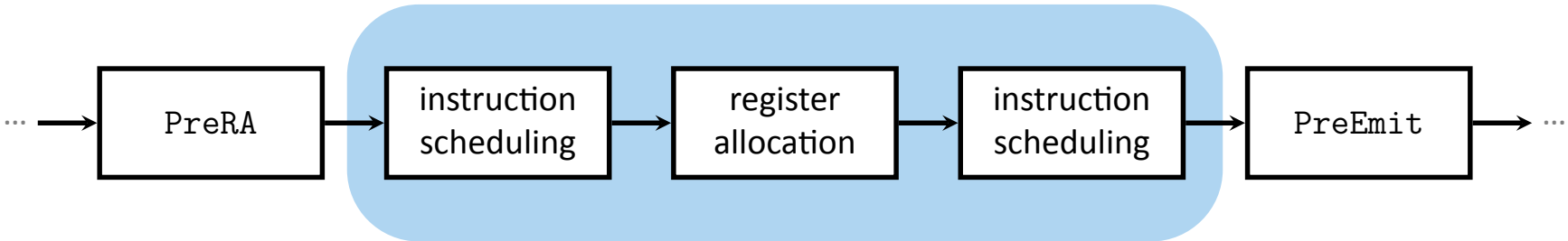


Code Generation in LLVM

Code Generation in LLVM

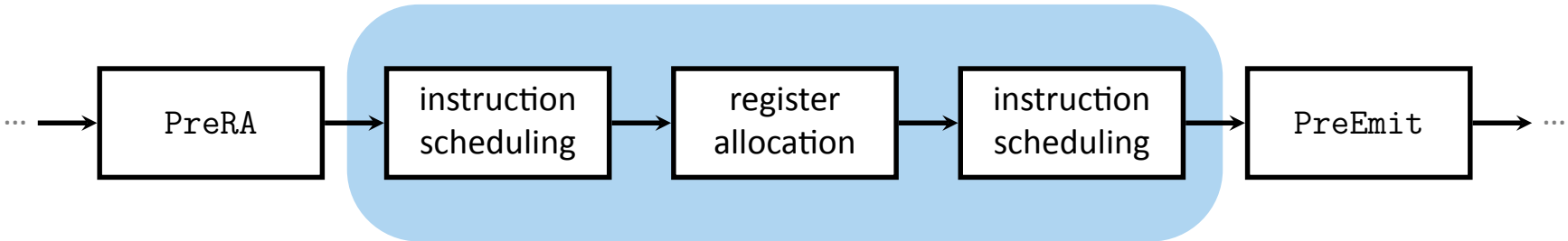


Code Generation in LLVM



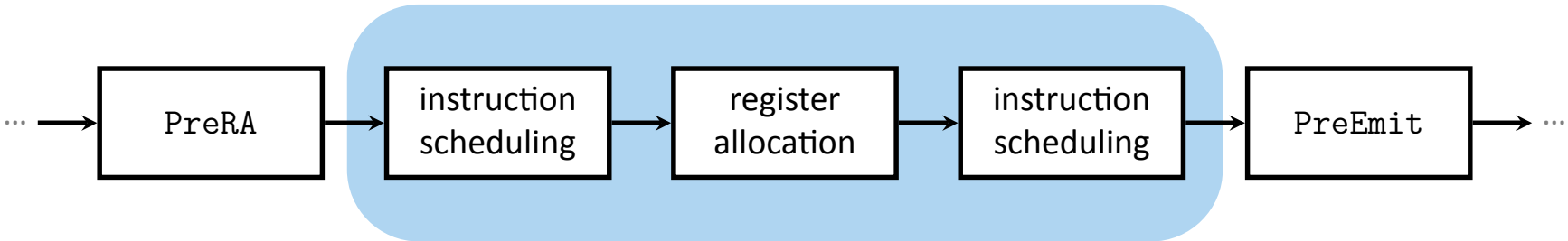
- Stages, heuristics

Code Generation in LLVM



- Stages, heuristics
- Pros: compilation speed

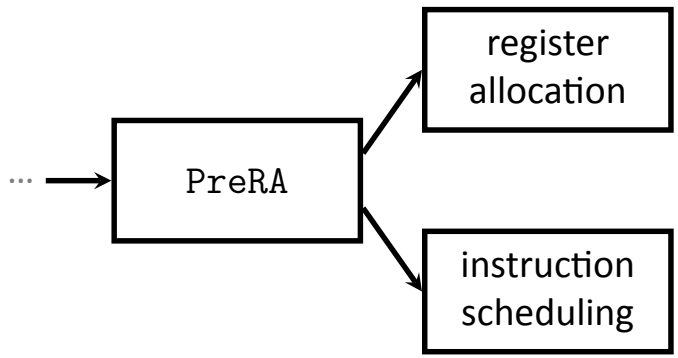
Code Generation in LLVM



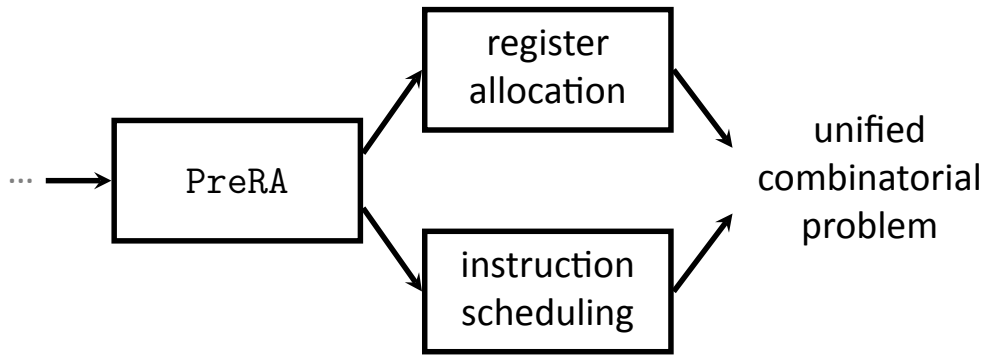
- Stages, heuristics
- Pros: compilation speed
- Cons: suboptimal, complex

Introducing Unison

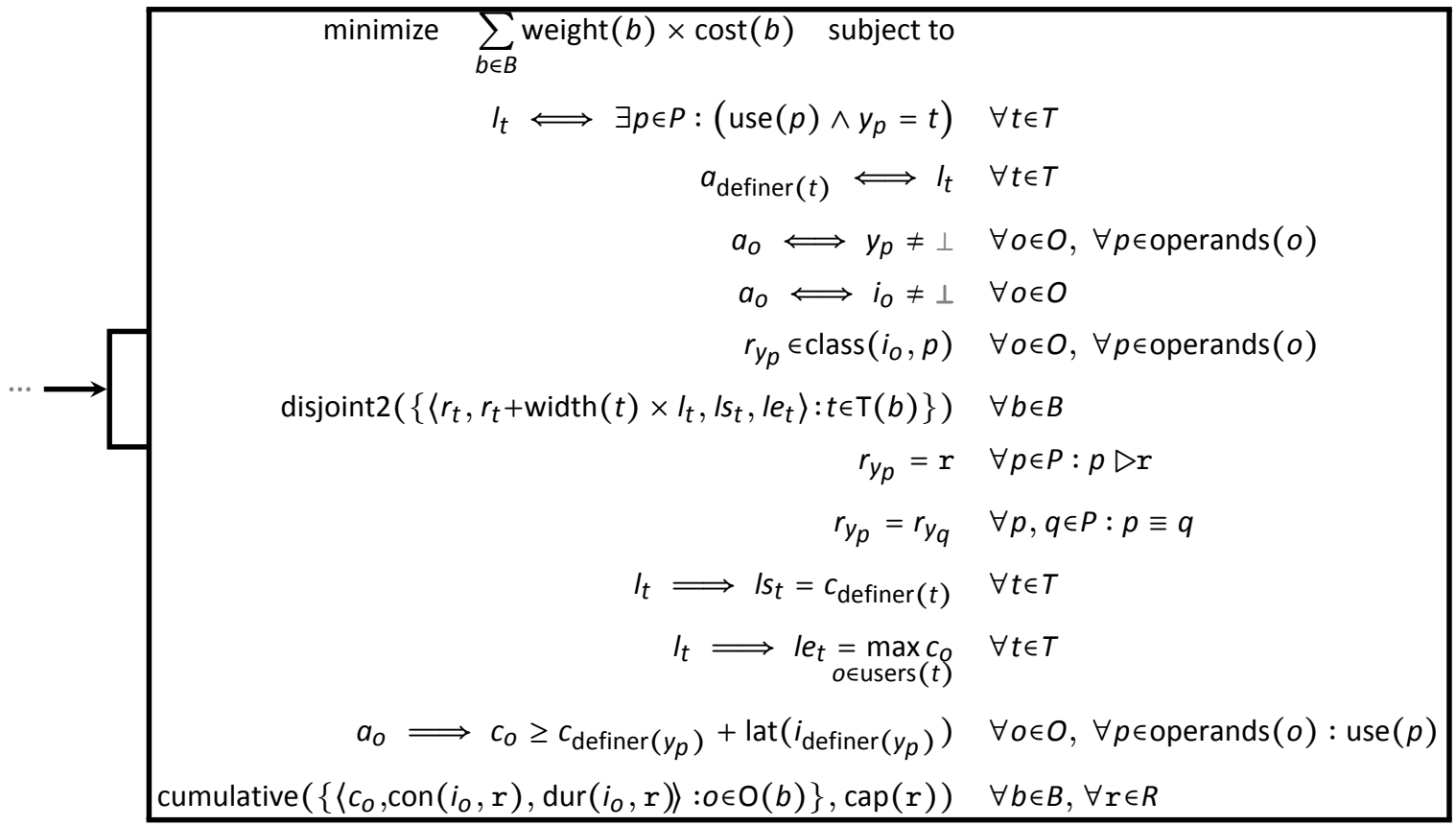
Introducing Unison



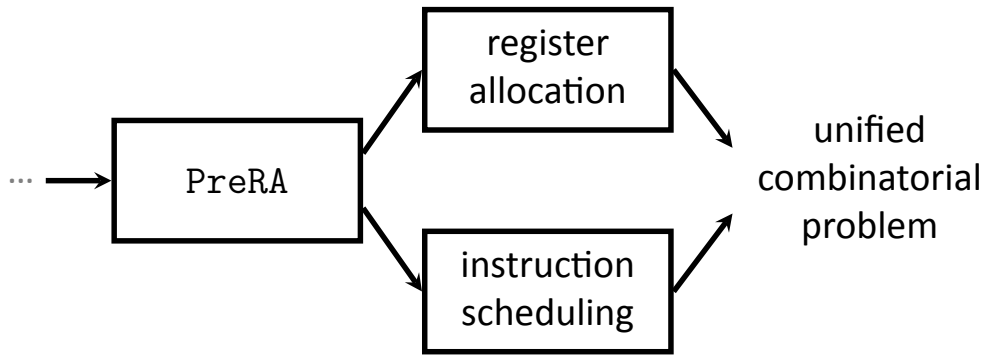
Introducing Unison



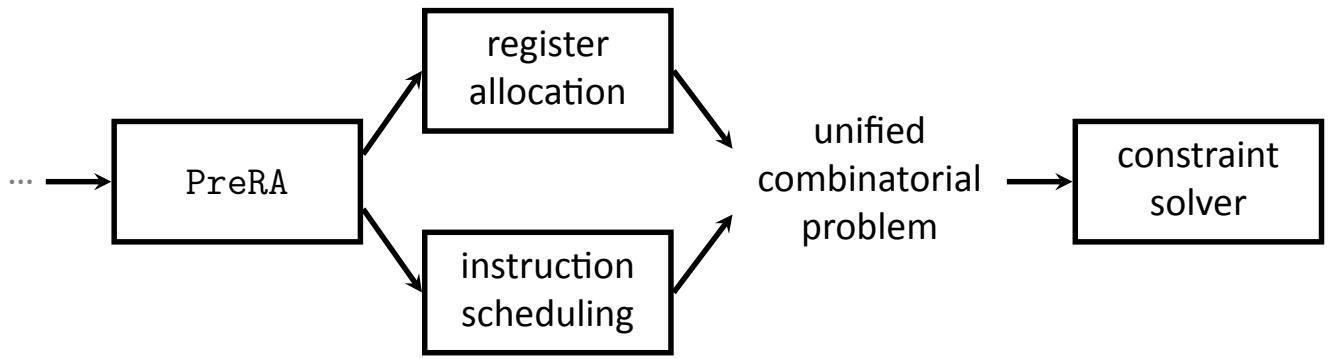
Introducing Unison



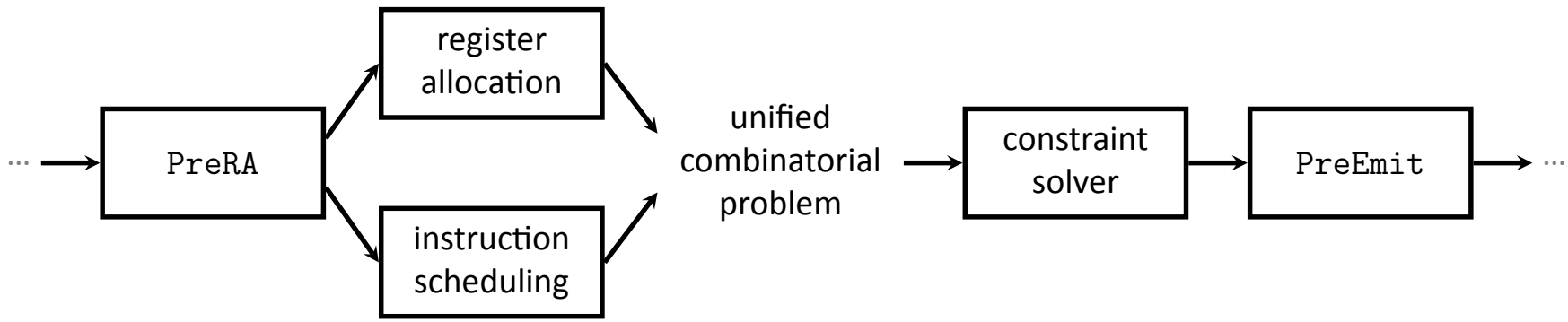
Introducing Unison



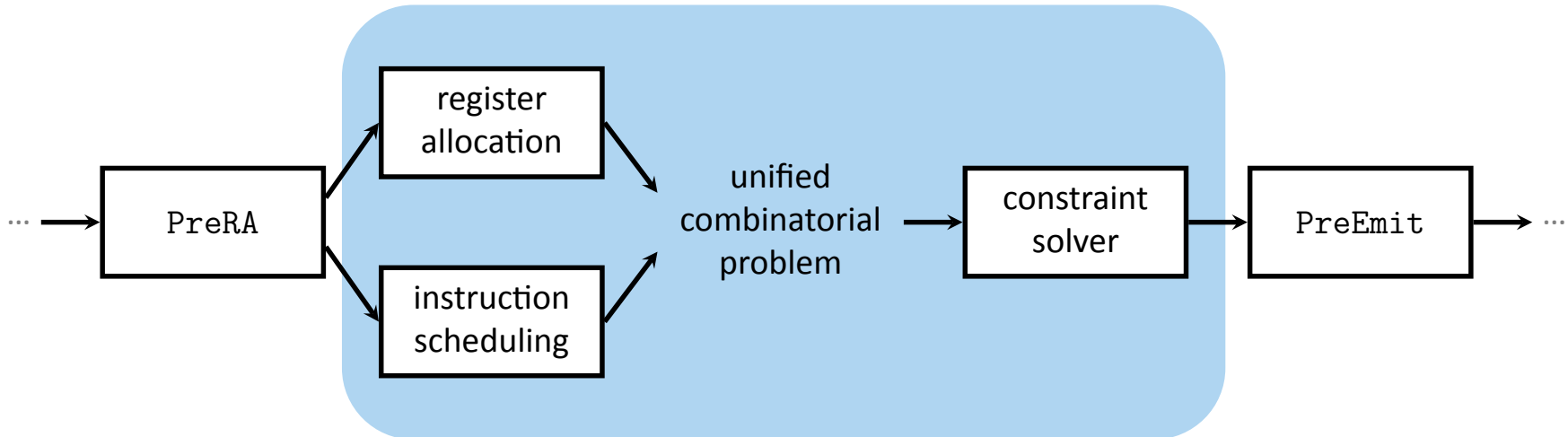
Introducing Unison



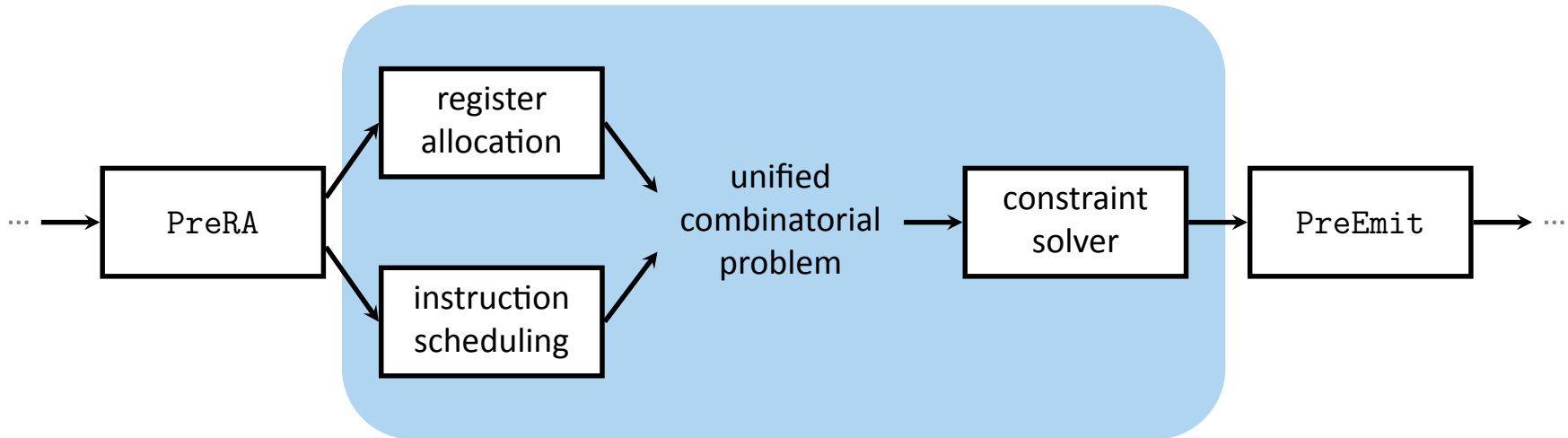
Introducing Unison



Introducing Unison

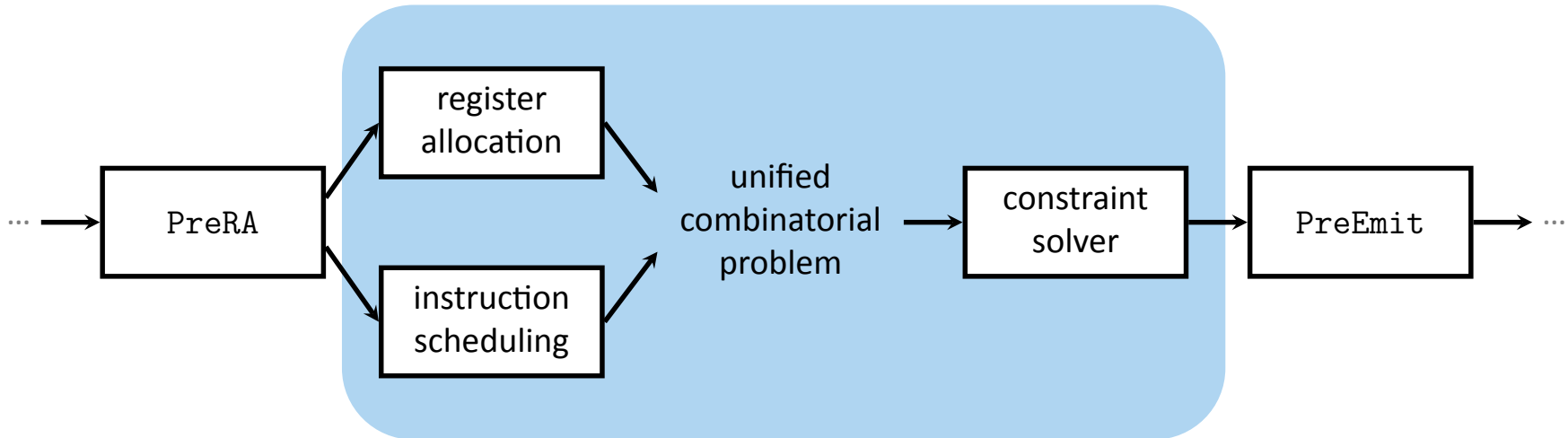


Introducing Unison



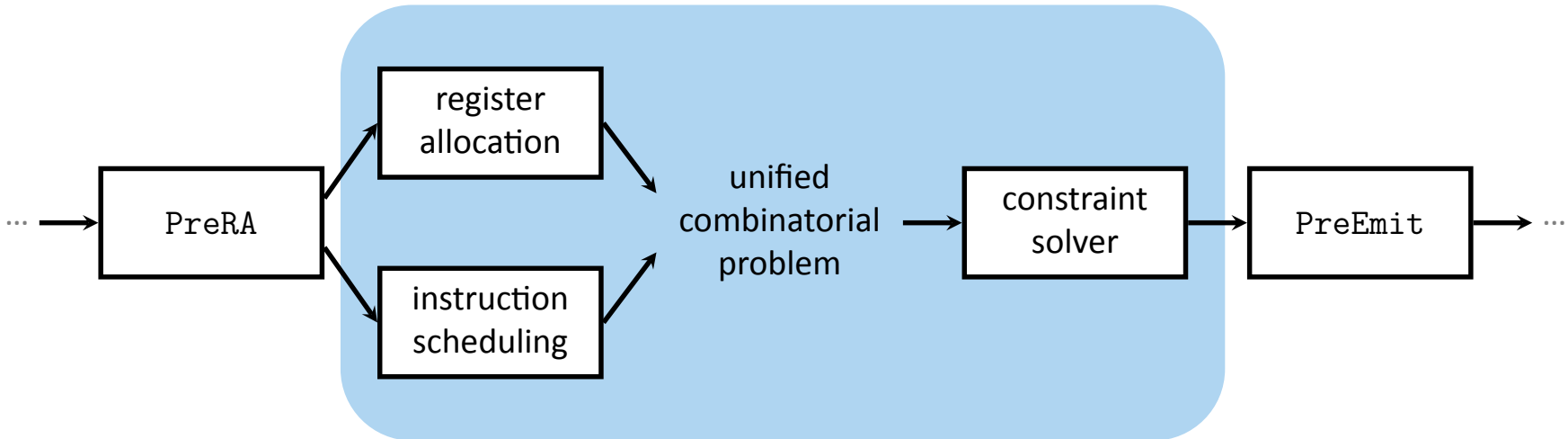
- Integration, combinatorial optimization

Introducing Unison



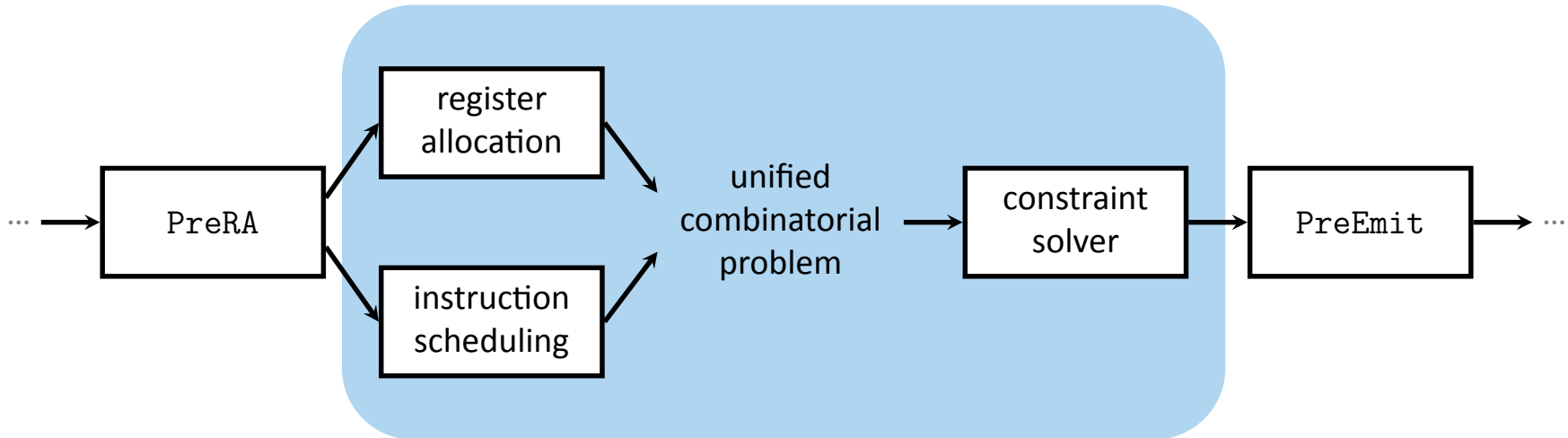
- Integration, combinatorial optimization
- Pros: simple, optimal

Introducing Unison



- Integration, combinatorial optimization
- Pros: simple, optimal
- Cons: compilation slowdown

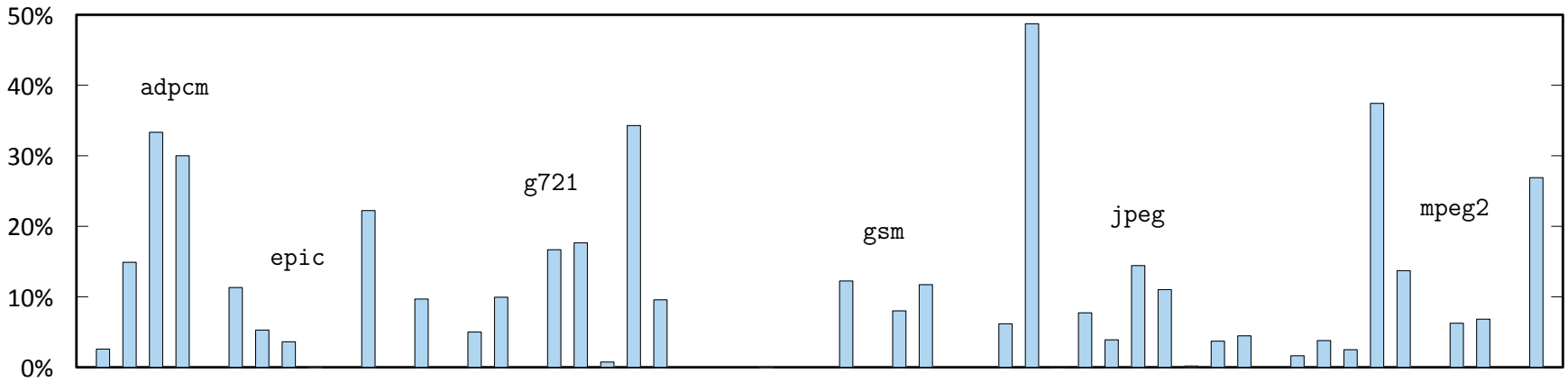
Introducing Unison



- Integration, combinatorial optimization
- Pros: simple, optimal
- Cons: compilation slowdown

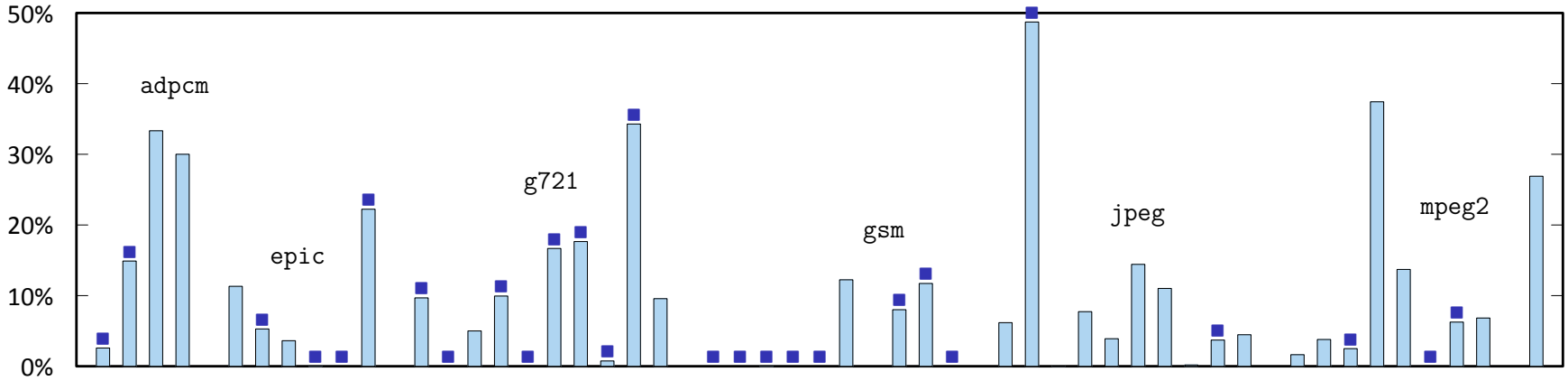
perfect complement to LLVM!

Speedup over LLVM 3.8



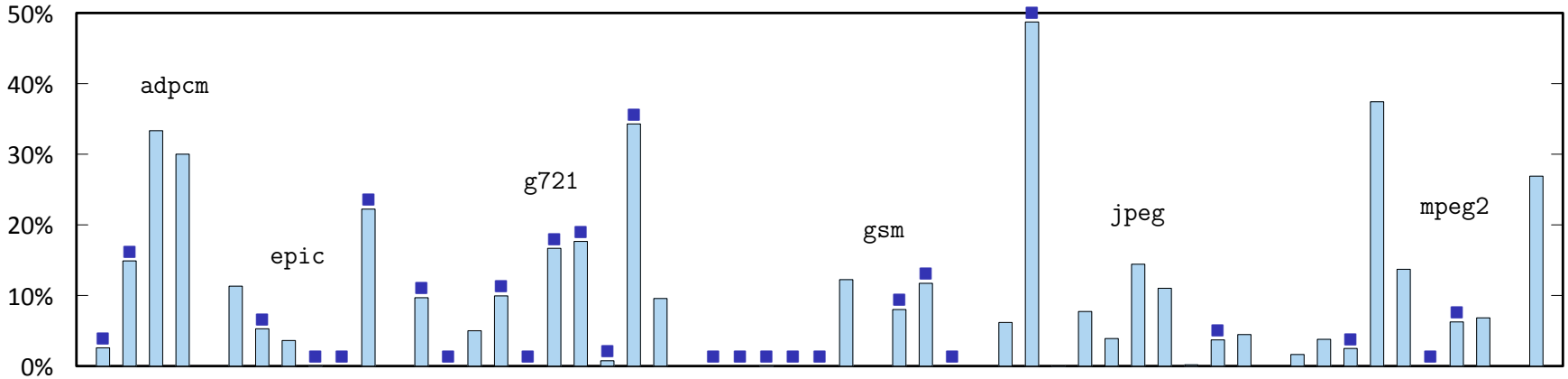
- 50 MediaBench functions
- Hexagon V4 processor

Speedup over LLVM 3.8



- 50 MediaBench functions
- Hexagon V4 processor
- Provably optimal (■) for 54% of the functions

Speedup over LLVM 3.8



- 50 MediaBench functions
- Hexagon V4 processor
- Provably optimal (■) for 54% of the functions
- Compilation time: from seconds to minutes

Unison Is Practical and Effective

- Integrated
  - register allocation
  - instruction scheduling

Unison Is Practical and Effective

- Integrated
  - register allocation
  - instruction scheduling
- Simple, optimal, slower

Unison Is Practical and Effective

- Integrated
  - register allocation
  - instruction scheduling
- Simple, optimal, slower
- Complements LLVM:
  - traditional LLVM for compile/debug cycle

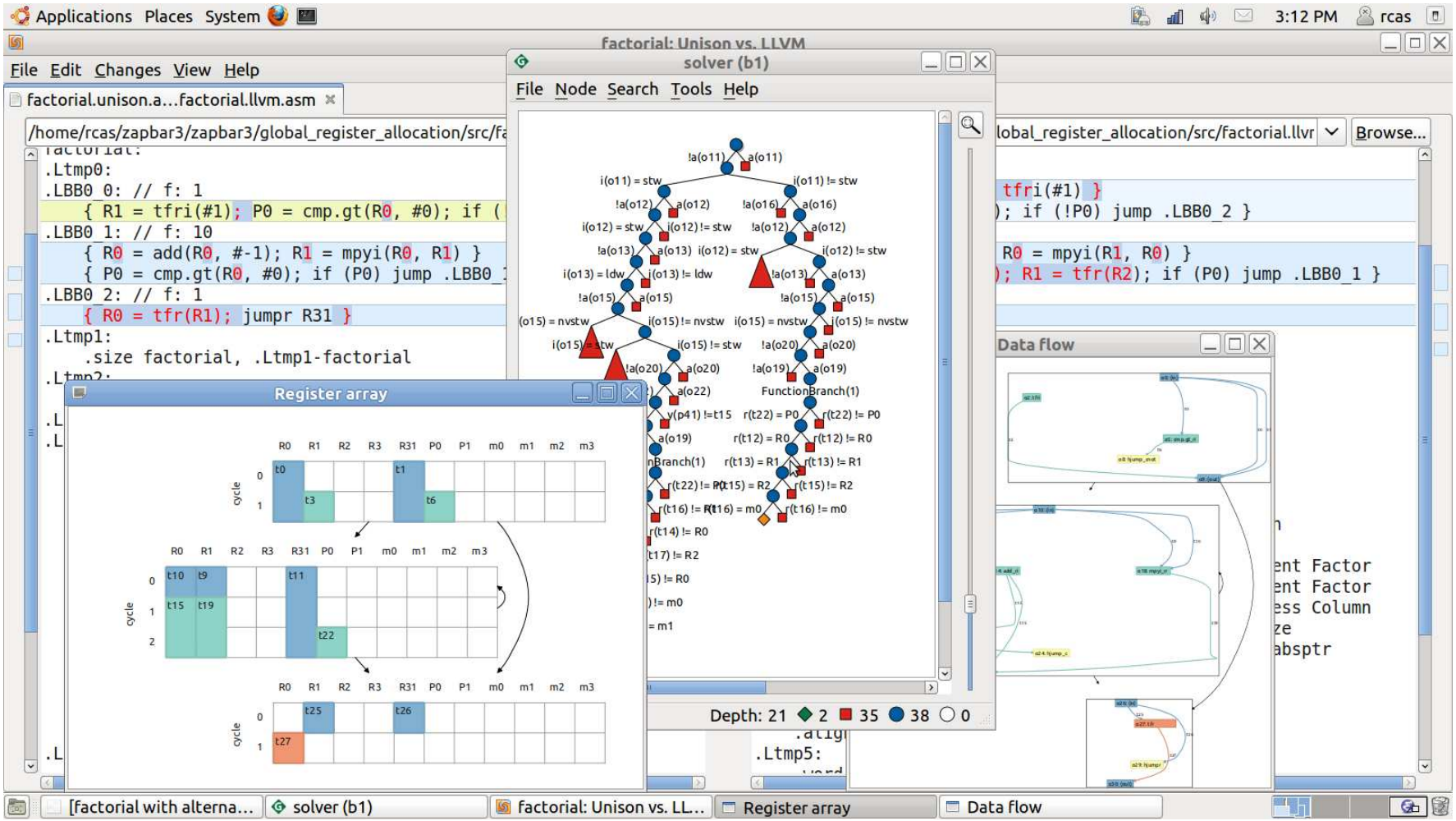
Unison Is Practical and Effective

- Integrated
  - register allocation
  - instruction scheduling
- Simple, optimal, slower
- Complements LLVM:
  - traditional LLVM for compile/debug cycle
  - LLVM + Unison for release builds

Unison Is Practical and Effective

- Integrated
  - register allocation
  - instruction scheduling
- Simple, optimal, slower
- Complements LLVM:
  - traditional LLVM for compile/debug cycle
  - LLVM + Unison for release builds
- Useful analysis tool for LLVM developers
  - how good is my heuristic?

Demo at CC2016 (12:20)



www.sics.se/~rcas/unison-demo