CAL@DI
University of Athens

Towards Accurate Performance Modeling of RISC-V Designs

Odysseas Chatzopoulos
George Papadimitriou
George-Marios Fragkoulis
Dimitris Gizopoulos

University of Athens, Greece

Fifth Workshop on Computer Architecture Research with RISC-V (CARRV 2021)

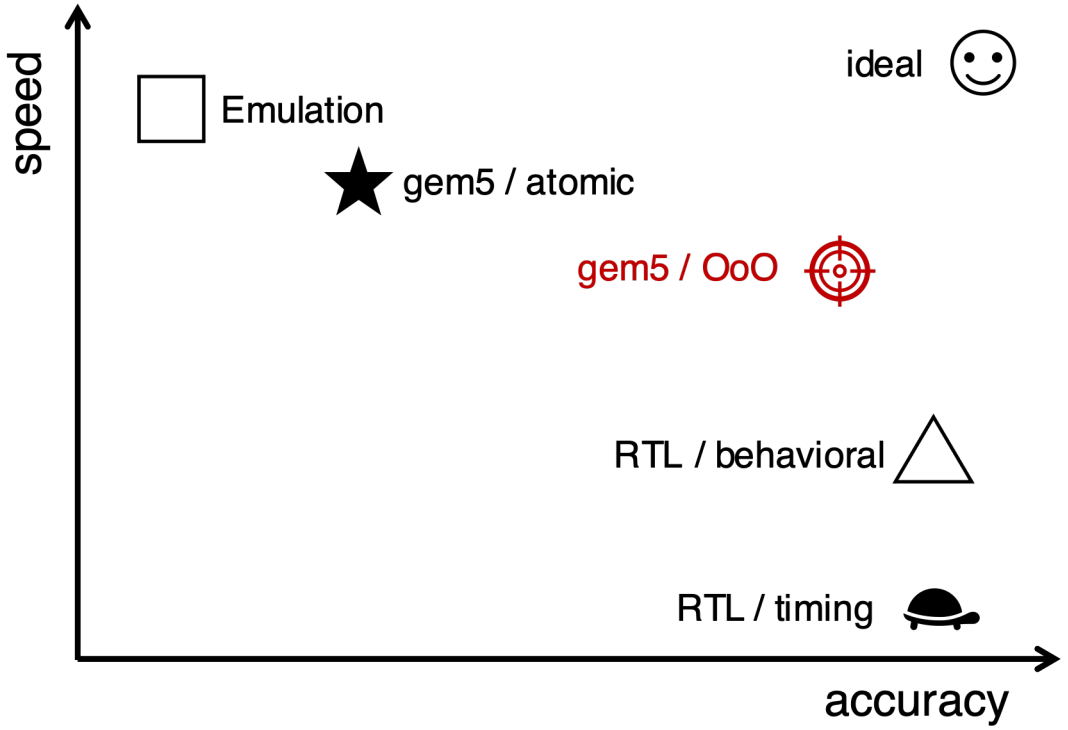
Microarchitectural Simulation

- Computer architects strongly rely on **microarchitectural simulators**
  - Performance-based studies
  - Evaluation and validation of new microarchitectures
  - Reliability evaluation
  - Power consumption
- **Accuracy**
  - Often fail to accurately model the real hardware providing design inconsistei

- **Speed**
  - High simulation throughput

RTL vs. Microarchitecture-Level Simulation

- **RTL simulation**
  - High accuracy
  - Low throughput when running in SW
  - Hard to modify and extend
- **Microarchitecture-level simulation**
  - Acceptable accuracy
  - High throughput
  - Extensible and easily configurable
  - Model several different architectures and microarchitectures



Motivation and Contributions

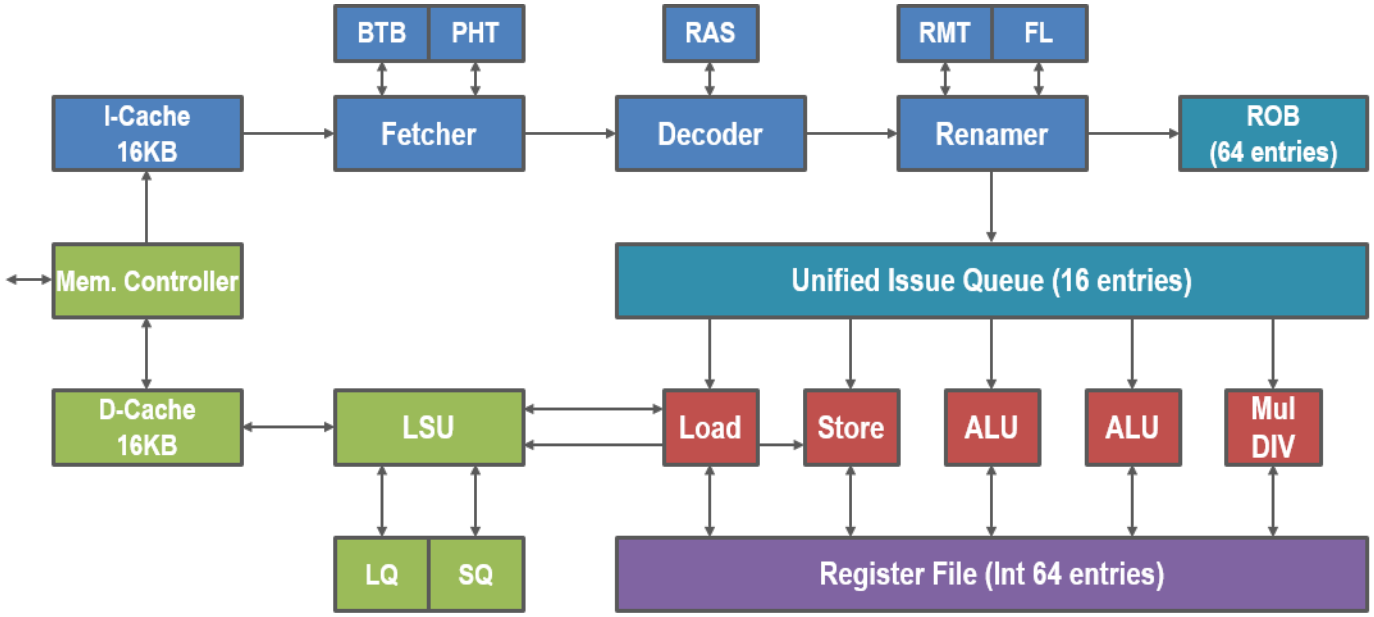
- RISC-V accurate performance modeling is still an ongoing task
- What is **the highest accuracy can be achieved** through microarchitecture-level simulation?
  - Comparison to an RTL simulation baseline
- **Our contributions**
  - Discuss about the **challenges and sources of error** between microarchitecture-level and RTL simulations
  - Modeling validation of two ISAs with **subtle differences**
    - RV32IM and RV64IM

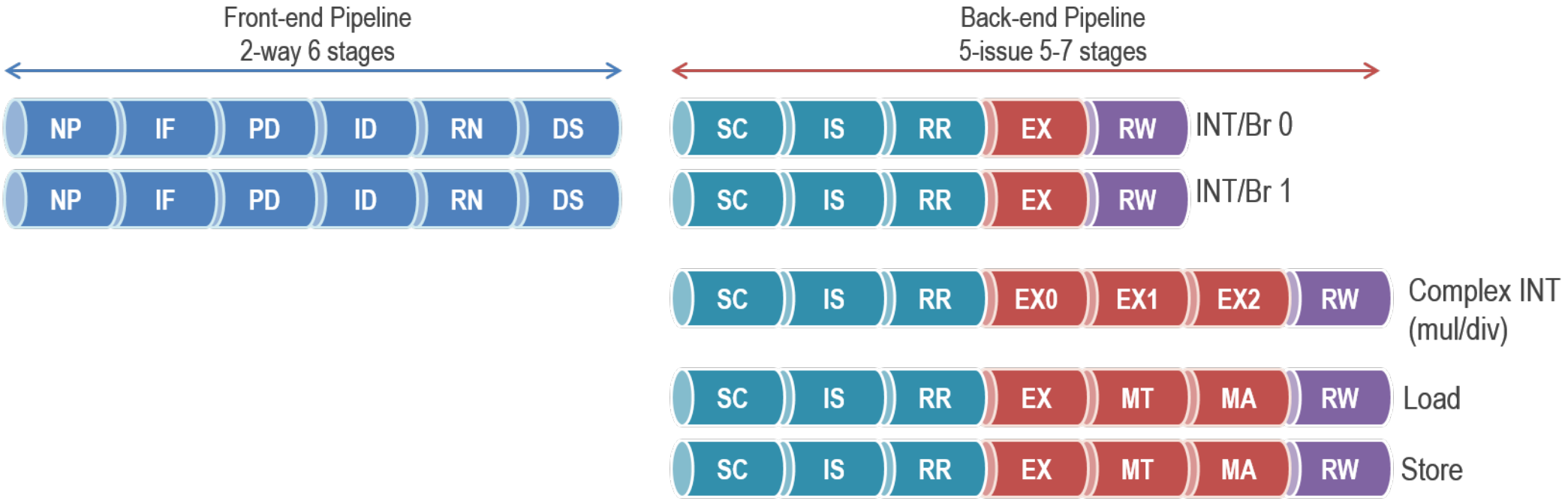
Sources of Simulation Error

- **Modeling errors**
  - When the simulator developers incorrectly model the desired functionality
  - Usually can be corrected when the simulator is gradually improved through its continuous maintenance
- **Specification errors**
  - When the developers are unaware of the functionality being modeled or must speculate about it
- **Abstraction errors**
  - When the developers abstract or fail to incorporate some details of the modeled design

RSD: A Reference RTL Model

- Open-source RISC-V OoO microprocessor core optimized for FPGAs
  - https://github.com/rsd-devel/rsd
- Provides high performance by supporting advanced microarchitectural features
- The RSD pipeline is structured using three basic blocks
  - **Front-end block**
    - It supports the **gshare branch predictor**
  - **Scheduling block**
  - **Execution block**


BTB
PHT
RAS
RMT
FL
I-Cache 16KB
Fetcher
Decoder
Renamer
ROB (64 entries)
Mem. Controller
Unified Issue Queue (16 entries)
D-Cache 16KB
LSU
Load
Store
ALU
ALU
Mul DIV
LQ
SQ
Register File (Int 64 entries)

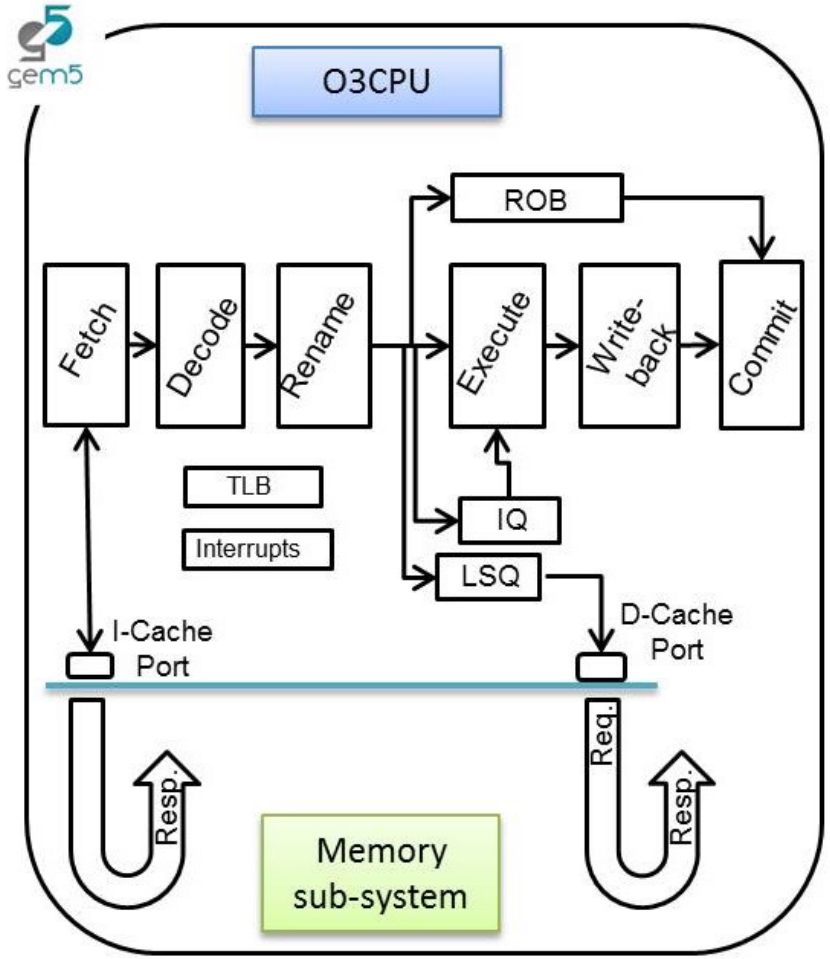

Front-end Pipeline
2-way 6 stages
Back-end Pipeline
5-issue 5-7 stages
NP IF PD ID RN DS
NP IF PD ID RN DS
SC IS RR EX RW
INT/Br 0
SC IS RR EX RW
INT/Br 1
SC IS RR EX0 EX1 EX2 RW
Complex INT (mul/div)
SC IS RR EX MT MA RW
Load
SC IS RR EX MT MA RW
Store

gem5 Microarchitectural Simulator

- **Widely used** open-source microarchitecture-level simulator
- Employed in both academia and industry
- Supports variety of ISAs, **including RISC-V**
- Two main simulation modes
  - **System-call emulation (SE):** does not load OS
    - System calls are emulated by the host system
  - **Full-system (FS):** executes both user-level and kernel-level instructions by loading an OS into the simulator

gem5 Utilized Components

- Variety of built-in microprocessor and memory system models
- Microprocessor Model → O3CPU
- The gem5 O3CPU pipeline consists of 5 stages
  - Fetch
  - Decode
  - Rename
  - Issue/Execute/Writeback
  - Commit
- Memory Model → SimpleMemory
- SE simulation mode is used throughout our work
  - RSD does not support a full OS
  - RISC-V FS mode was in early stages of development



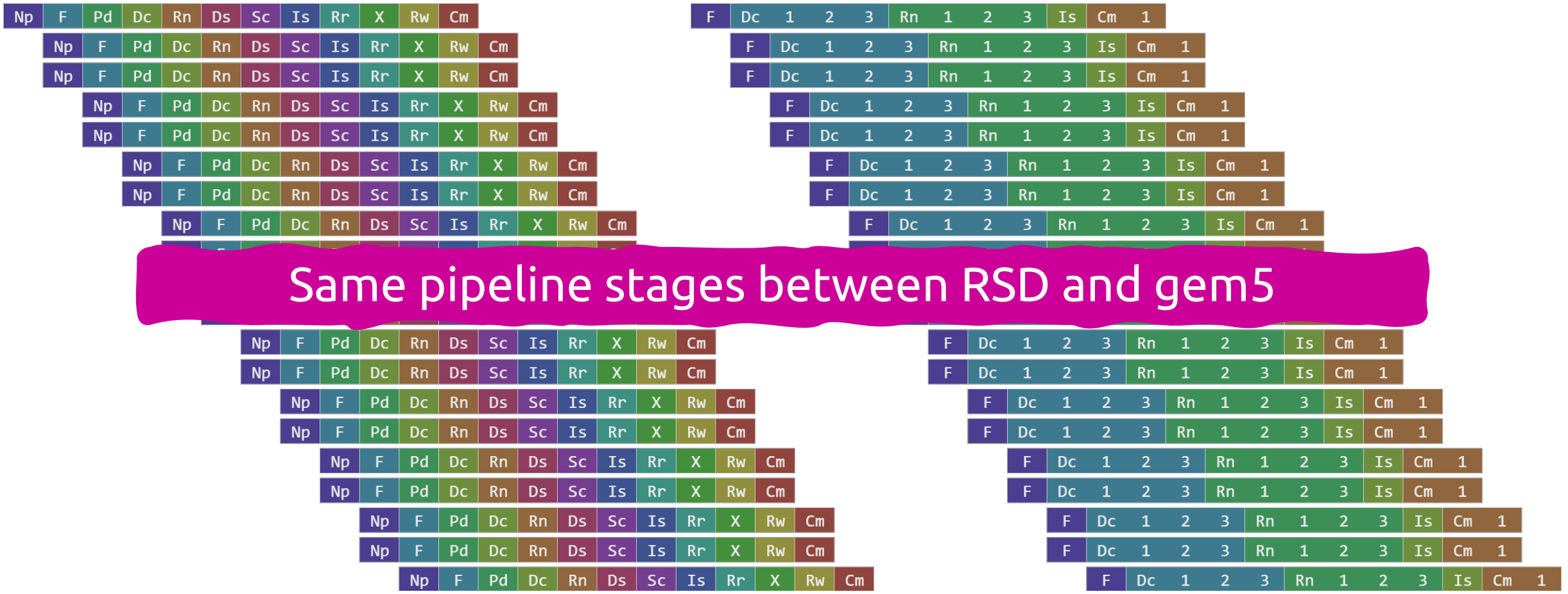
Experimental Methodology

- **Extract microarchitectural parameters** from RSD source code and configuration files and **translate** them to gem5 parameters
- To test parameter matching we developed **stressmarks targeted at specific system components** and run them on both simulated models
- Analyze simulation results
  - Compare IPC and Microarchitectural Event Counters
  - Use **Konata pipeline viewer** to further investigate execution flow
- Adjust gem5 parameters according to above results
- Run general purpose benchmarks

Main Modeling Parameters

Parameter	Value
ISA	RV32IM (RSD) – RV64IM (gem5)
L1 Data / Instruction Cache	4KB / 4KB (2-way)
Cache Line Size	8 Bytes
Replacement Policy	Tree-PLRU
L1 Hit / Miss Latency	1 / 100 clock cycles
Fetch / Decode / Rename / Commit Width	2
Issue / Writeback Width	5
Reorder Buffer	64 entries
MSHR	2 entries
Branch Predictor	gshare (2048 History Table)
Branch Target Buffer	1024 entries
Load / Store Queue	16 entries
Physical Register File	64 registers

Visual Pipeline Comparison using Konata



RSD

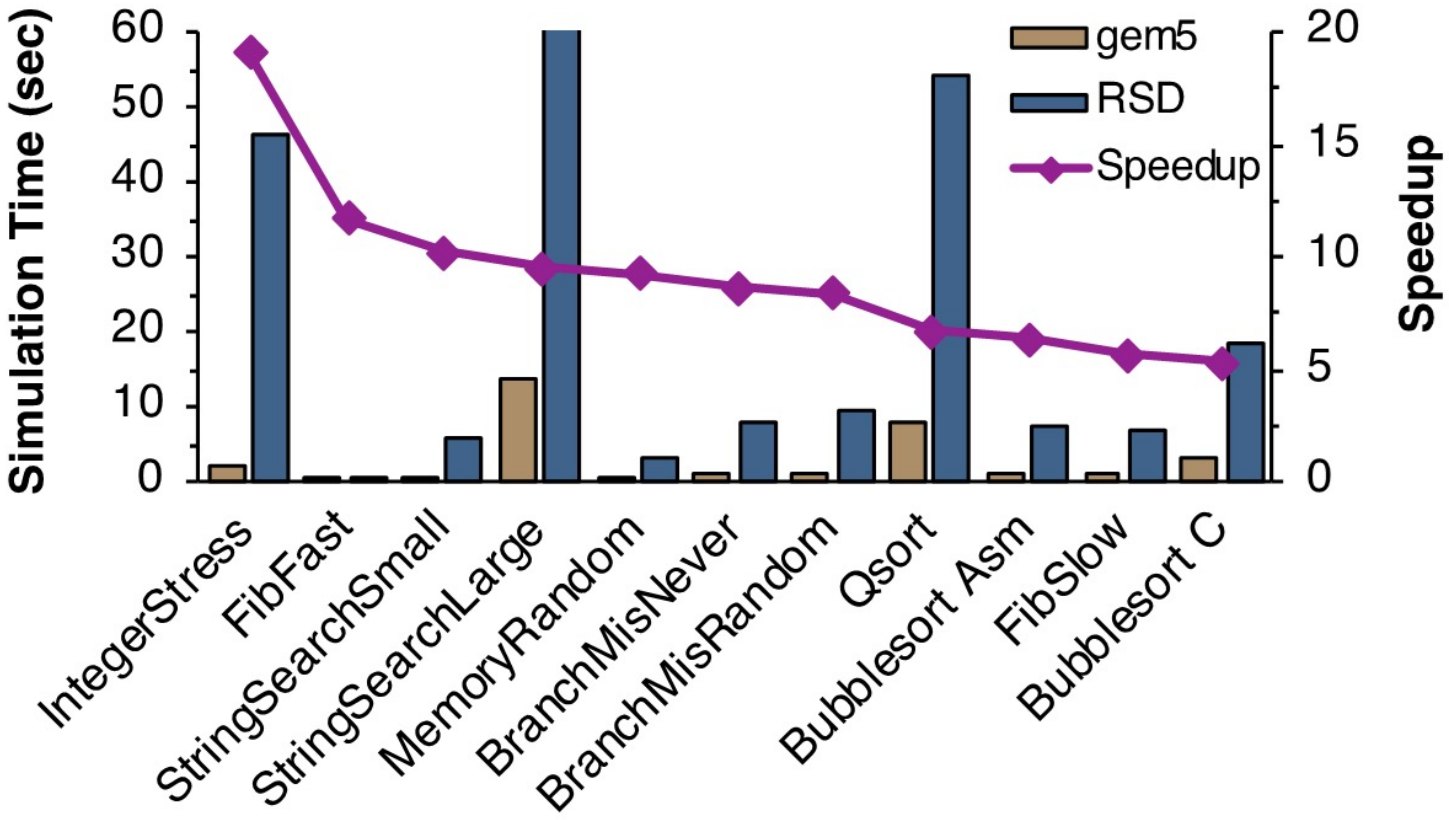
gem5

Utilized Benchmarks

- BubblesortC & BubblesortAsm: Sort a 250-entry array with integer values using the Bubblesort algorithm. Assembly version of Bubblesort was written because C version shows considerable difference in committed instructions between the two ISAs
- MemoryRandom: Study the behavior of memory upon random accesses
- FibSlow & Fibfast: FibSlow computes the 20th term of the Fibonacci sequence. FibFast computes all terms in the range 1 through 45 and reduces exponential complexity to linear
- IntegerStress: Stresses the integer functional units
- BranchMisNever: Always taken branch in a loop
- BranchMisRandom: Specify branch misprediction latency studying gshare's behavior on branches using random conditions
- StringSearchLarge & StringSearchSmall: Search for given words in phrases using a case insensitive comparison algorithm
- Qsort: Sorts a large array of strings into ascending order using the quick sort algorithm

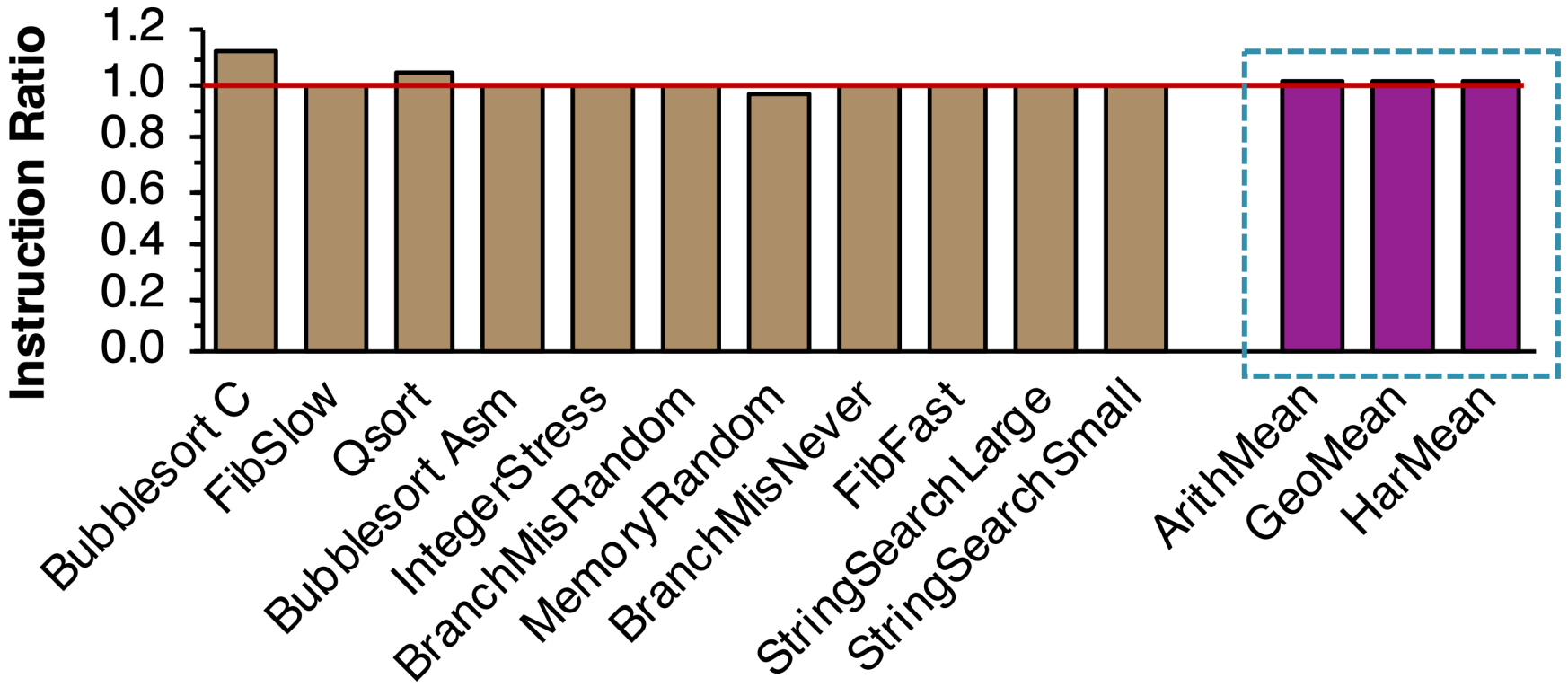
Simulation Speedup

- Speedup of microarchitecture-level ranges between 5x and 20x compared to RTL simulation

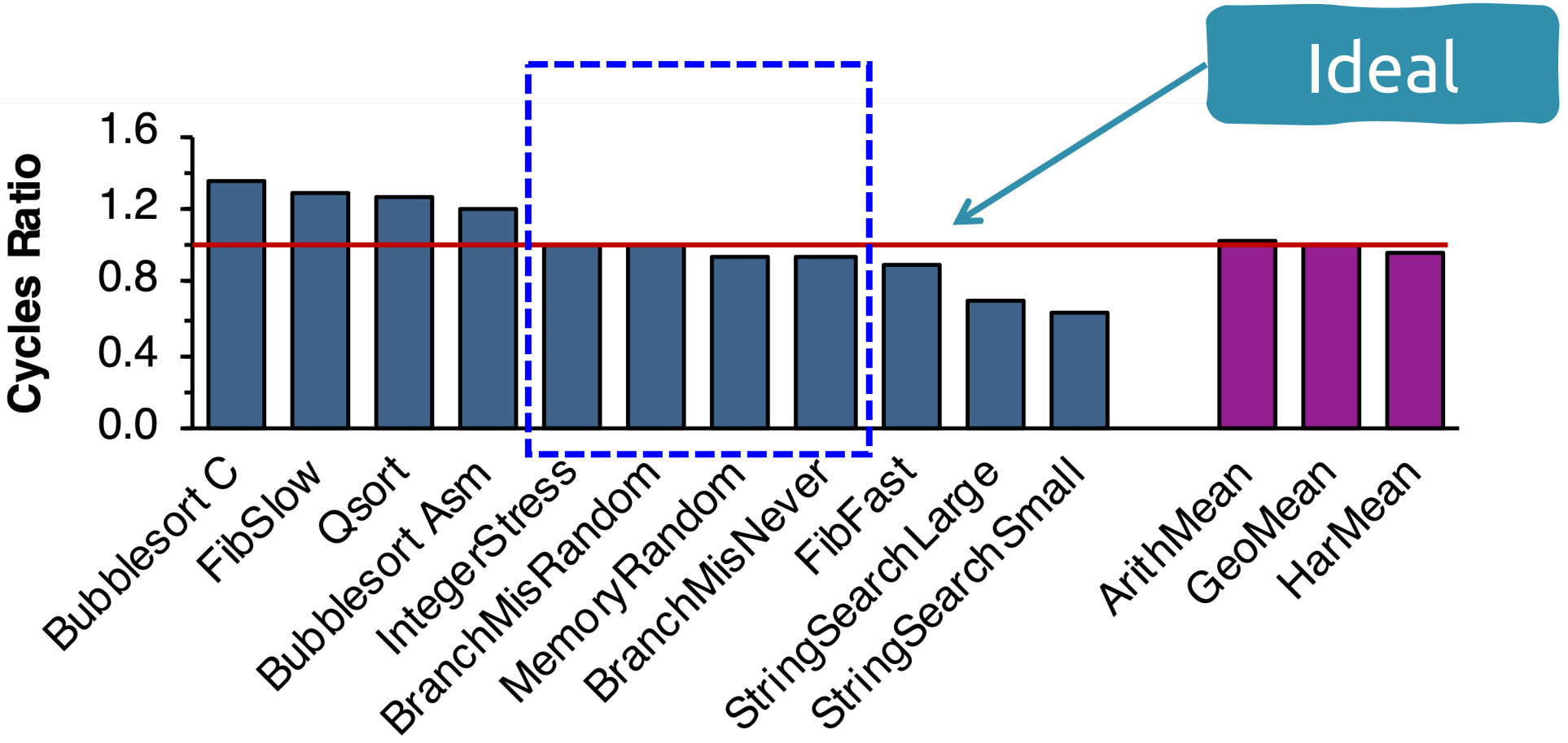


Simulation Accuracy: Committed Instructions

Most benchmarks provide similar instruction count between gem5 and RSD



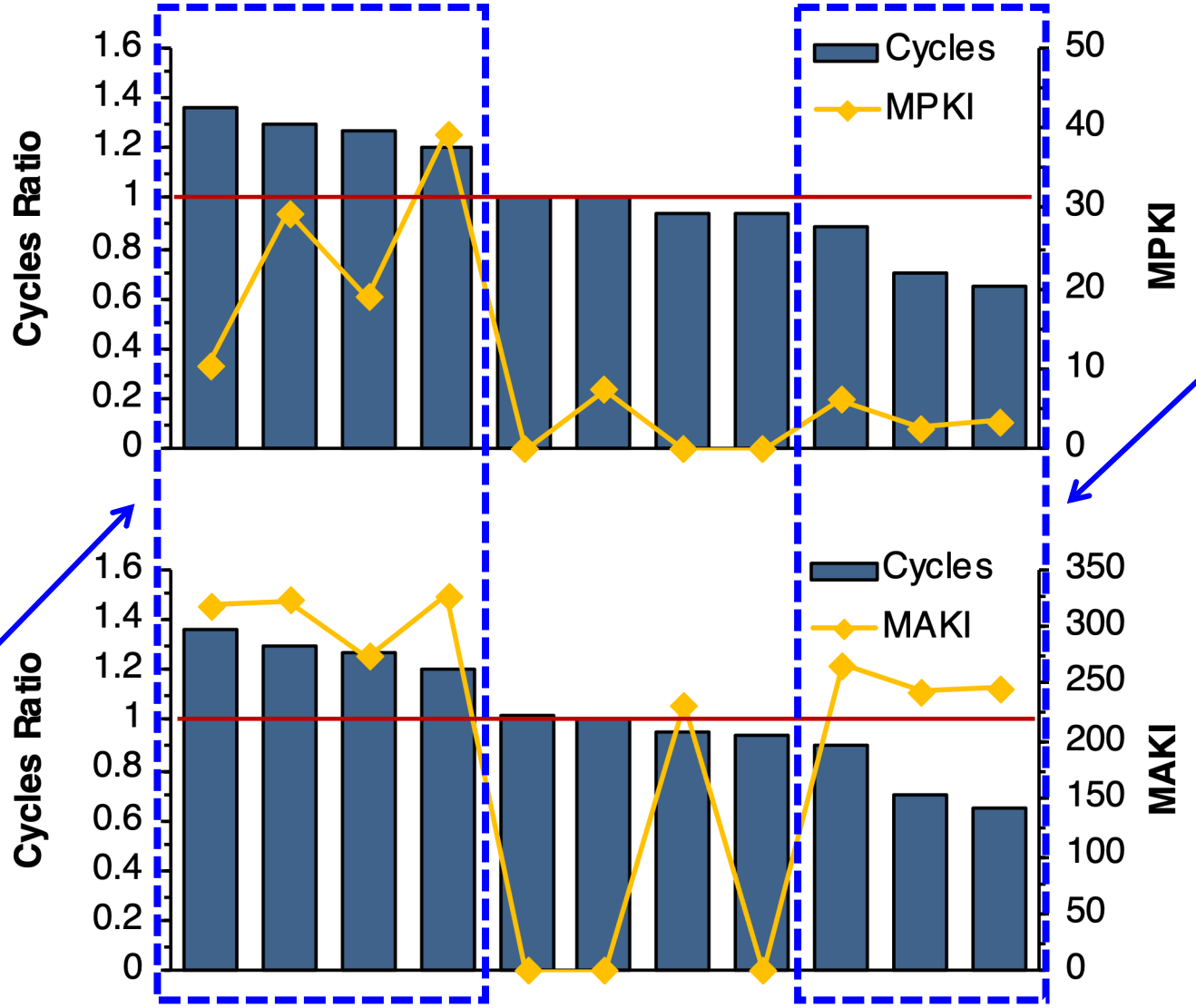
Simulation Accuracy: Clock Cycles



Main Sources of Error

- We employ microarchitectural counters for
  - Memory Accesses
  - Cache Misses
  - Branch Prediction Misses
  - MSHR Hits
  - Memory Latency
- **Memory system** and **branch predictor** seem to be **the main sources of error**
  - Branch mispredictions per kilo-instructions and memory accesses per kilo-instructions show **the highest correlation with simulation error**

Main Sources of Error



High misprediction rate and the highest memory accesses

Low misprediction rate but still very high memory accesses

Memory Accesses per Kilo Instructions

Conclusion & Future Work

- **Accuracy validation** of performance modeling of microarchitecture-level simulation
  - Present the challenges of performance modeling
- Clock cycles difference can be up to **36%** due to the **abstraction errors**
- Make targeted changes to the **memory system and branch predictor**
- Use an OS capable RISC-V processor along with HW acceleration to validate the FS simulation mode of gem5
- Employing **automated scripts** that provide the most important microarchitectural parameters of black-box processors

рахмат
danke
謝謝
ngiyabonga
teşekkür ederim
Баярлалаа
mersi
kia ora
barka
welalin
tack
spas
mahalo
tapadh leat
спасибо
faafetai lava
vinaka
спасибі
blagodaram
dank je
misaotra
paldies
grazzi
matondo
kiitos
dankie
nanni
nandri
dhanyavad
thank you
gracias
хвала
asante
manana
bedankt
bayarlalaa
gràçie
hvala
maururu
köszönöm
dankon
aciû
akun
obrigada
tenki
murakoze
enkosi
djiere dieuf
tau
mochchakkeram
дякую
mamnun
chokrane
dziękuję
chnorakaloutioun
gratias ago
gràcies
sulpáy
go raibh maith agat
sobodi
děkuji
sagolun
sukriya
kop khun krap
taiku
obrigado
mèsi
didi madloba
najis tuke
ありがとう
tanemirt
grazie
arigatô
takk
dakujem
trugarez
terima kasih
rahmet
diolch
dhanyavadagalu
shukriya
mercé
мерси
kam sah hamnida
rahmat
তোমাকে ধন্যবাদ
감사합니다
xièxie
ευχαριστώ
merci

Talk on Thursday, June 17, 2021

Towards Accurate Performance Modeling of RISC-V Designs

Odysseas Chatzopoulos
George-Marios Fragkoulis
George Papadimitriou
Dimitris Gizopoulos

University of Athens, Greece
http://cal.di.uoa.gr

Fifth Workshop on Computer Architecture Research with RISC-V (CARRV 2021)

CAL@DI
University of Athens