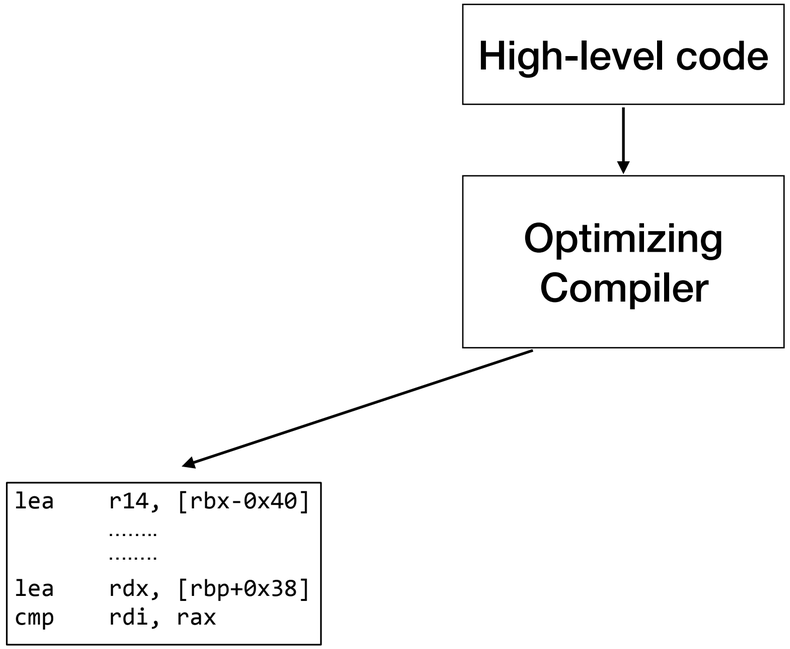
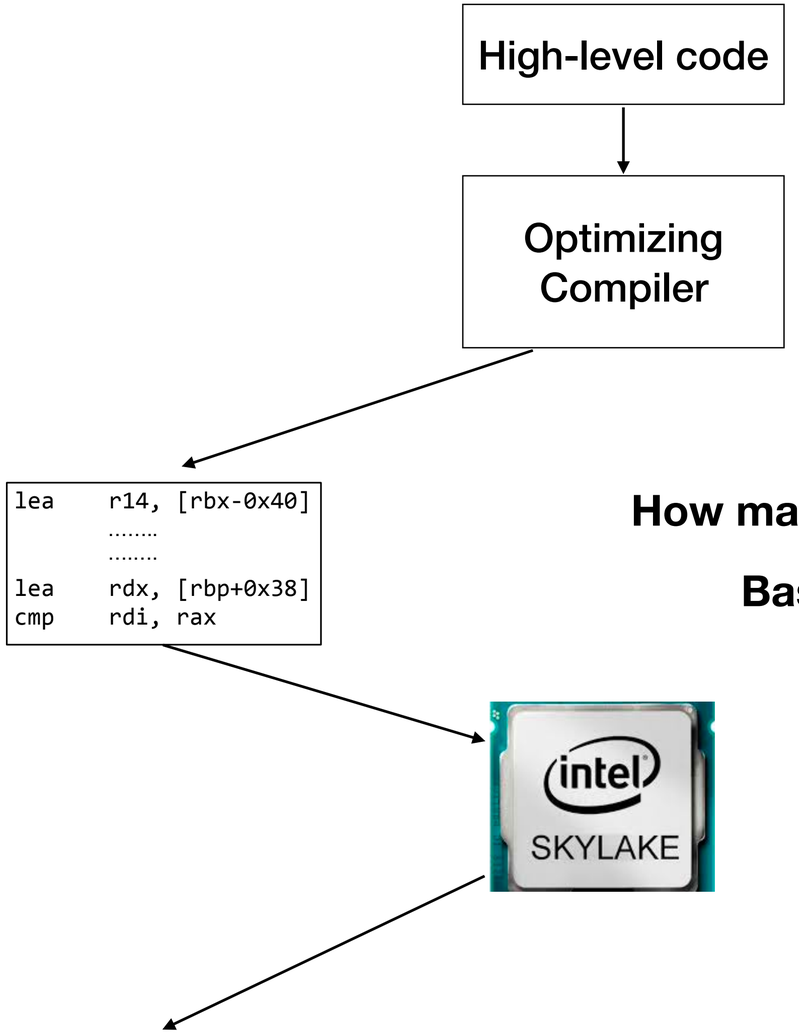
Ithemal: Accurate, Portable and Fast Basic Block Throughput Estimation using Deep Neural Networks

Charith Mendis
Alex Renda
Saman Amarasinghe
Michael Carbin

Compilers need to search through code Sequences



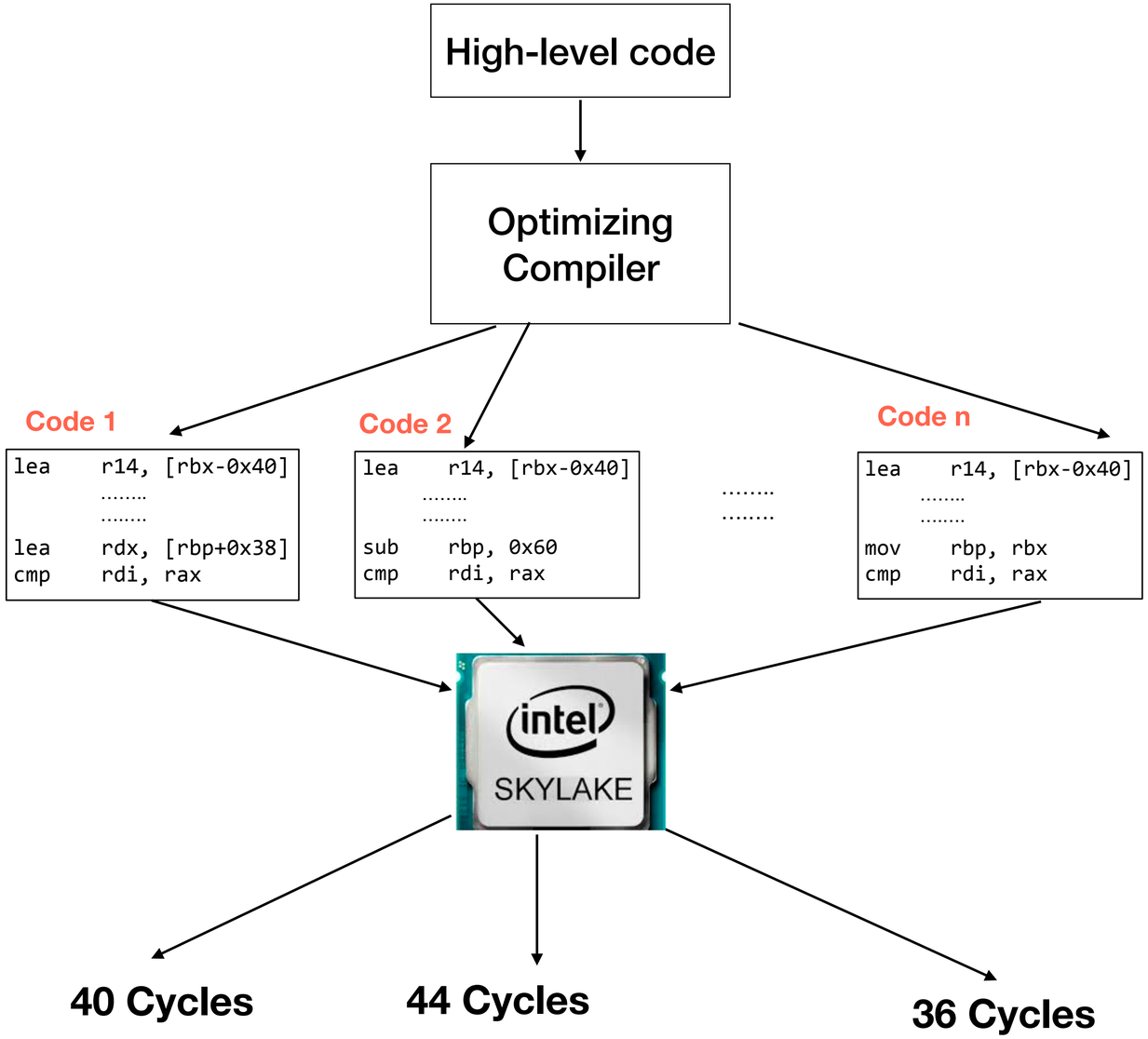
Compilers need to search through code Sequences



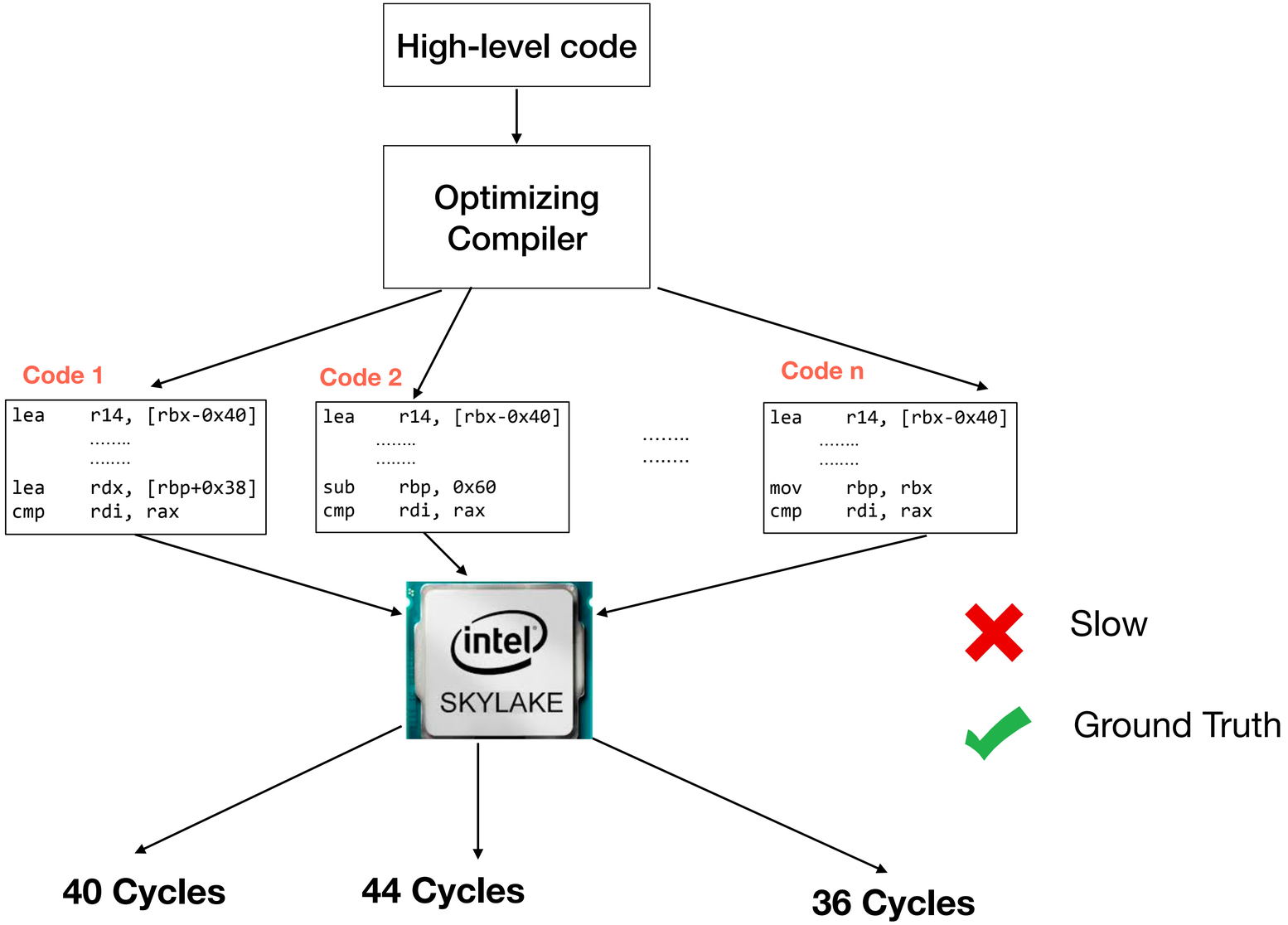
How many cycles does it take to run?

Basic Block Throughput

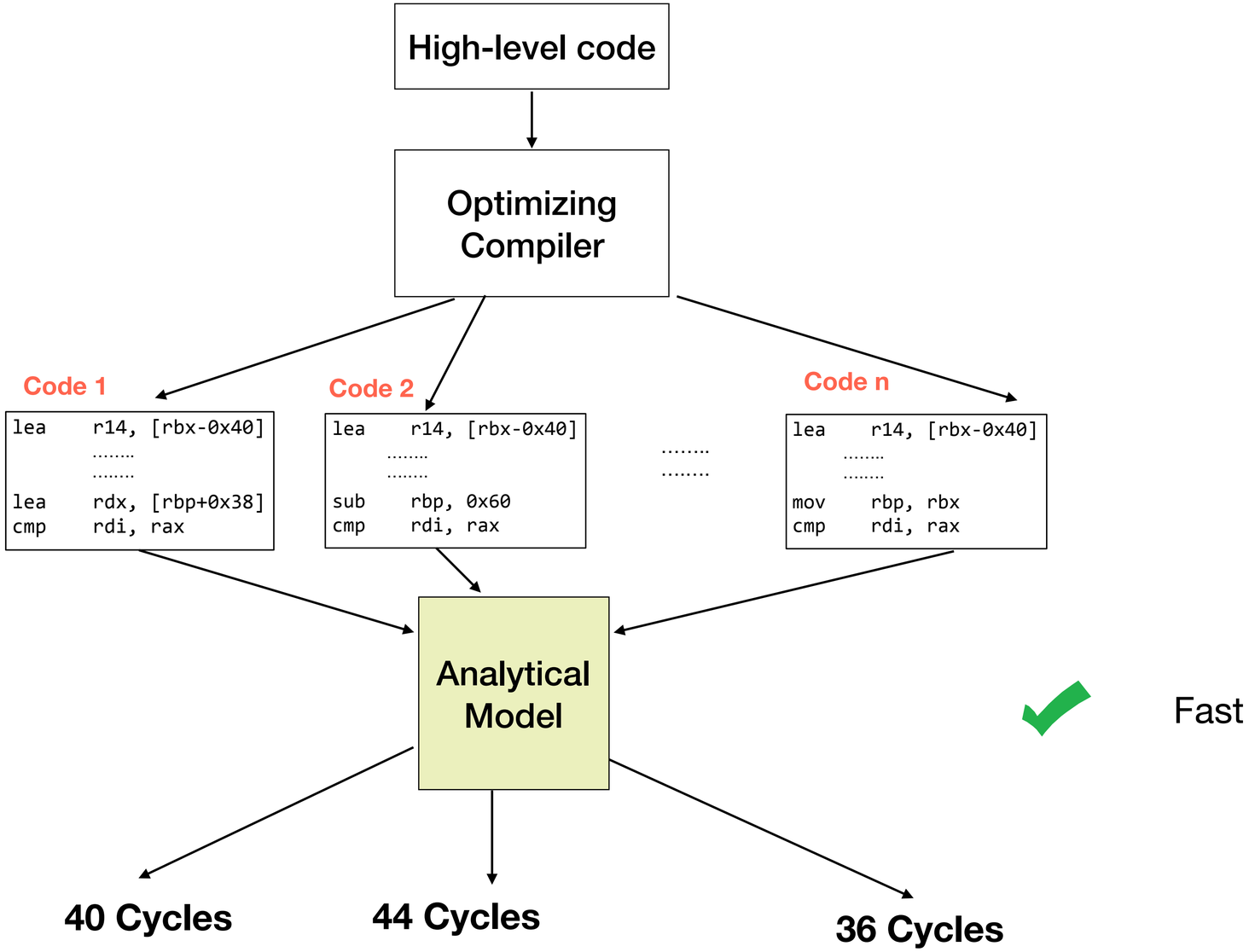
Compilers need to search through code Sequences



Compilers need to search through code Sequences



Compilers need to search through code Sequences



Analytical models are inaccurate

$\approx$

Analytical Model

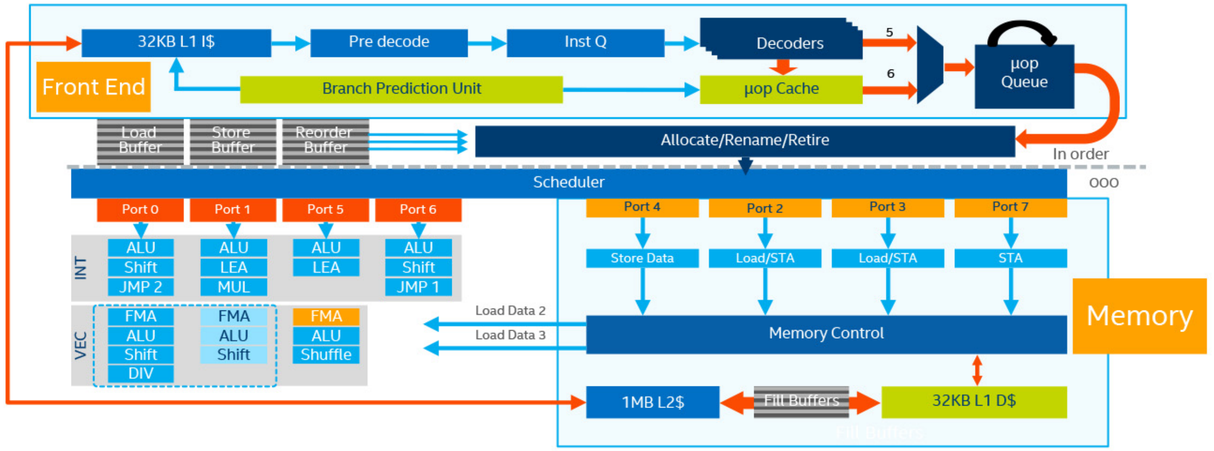
~20% error

- out-of-order
- pipelined
- super-scalar
- bypassed
- stateful components
- complicated and inaccurate manuals
- **opaque implementations (vendor specific)**

Analytical models are inaccurate

≈

Analytical Model



	Broadwell uArch	Skylake uArch
Out-of-order Window	192	**224**
In-flight Loads + Stores	72 + 42	72 + **56**
Scheduler Entries	60	**97**
Registers – Integer + FP	168 + 168	**180** + 168
Allocation Queue	56	**64/thread**
L1D BW (B/Cyc) – Load + Store	64 + 32	**128 + 64**
L2 Unified TLB	4K+2M: 1024	4K+2M: **1536** **1G: 16**

prediction problem is highly non-linear

Motivating Example - Zero Idioms

```
vxorps xmm0, xmm0, xmm0
```

Intel Architecture Optimization Reference Manual 662 of 672

```
vxorps xmm1, xmm2, xmm3
```

Throughput: 1 clock cycle

Intel Architecture Optimization Reference Manual 51 of 672

Dependency Breaking Idioms

Instruction parallelism can be improved by using common instructions to clear register contents to zero. The renamer can detect them on the zero evaluation of the destination register.

Use one of these dependency breaking idioms to clear a register when possible.

- XOR REG,REG
- SUB REG,REG
- PXOR/VPXOR XMMREG,XMMREG
- PSUBB/W/D/Q XMMREG,XMMREG
- VPSUBB/W/D/Q XMMREG,XMMREG
- XORPS/PD XMMREG,XMMREG
- VXORPS/PD YMMREG, YMMREG

Since zero idioms are detected and removed by the renamer, they have no execution latency.

Special Case Throughput: 0.33 clock cycles

Motivating Example - Zero Idioms

llvm-mca

- Part of LLVM compiler infrastructure
- Uses industry standard compiler (LLVM) scheduling models
- e.g., more than >230 commits spread over 2 years for x86 Haswell Scheduling model

IACA

- Intel Architecture Code Analyzer
- Developed in-house at Intel

```
vxorps xmm0, xmm0, xmm0
```

100 iterations

Method	Estimate
Measured	**32**
llvm-mca	100
IACA	24

Analytical models are not portable

≈

Analytical Model 1

≈

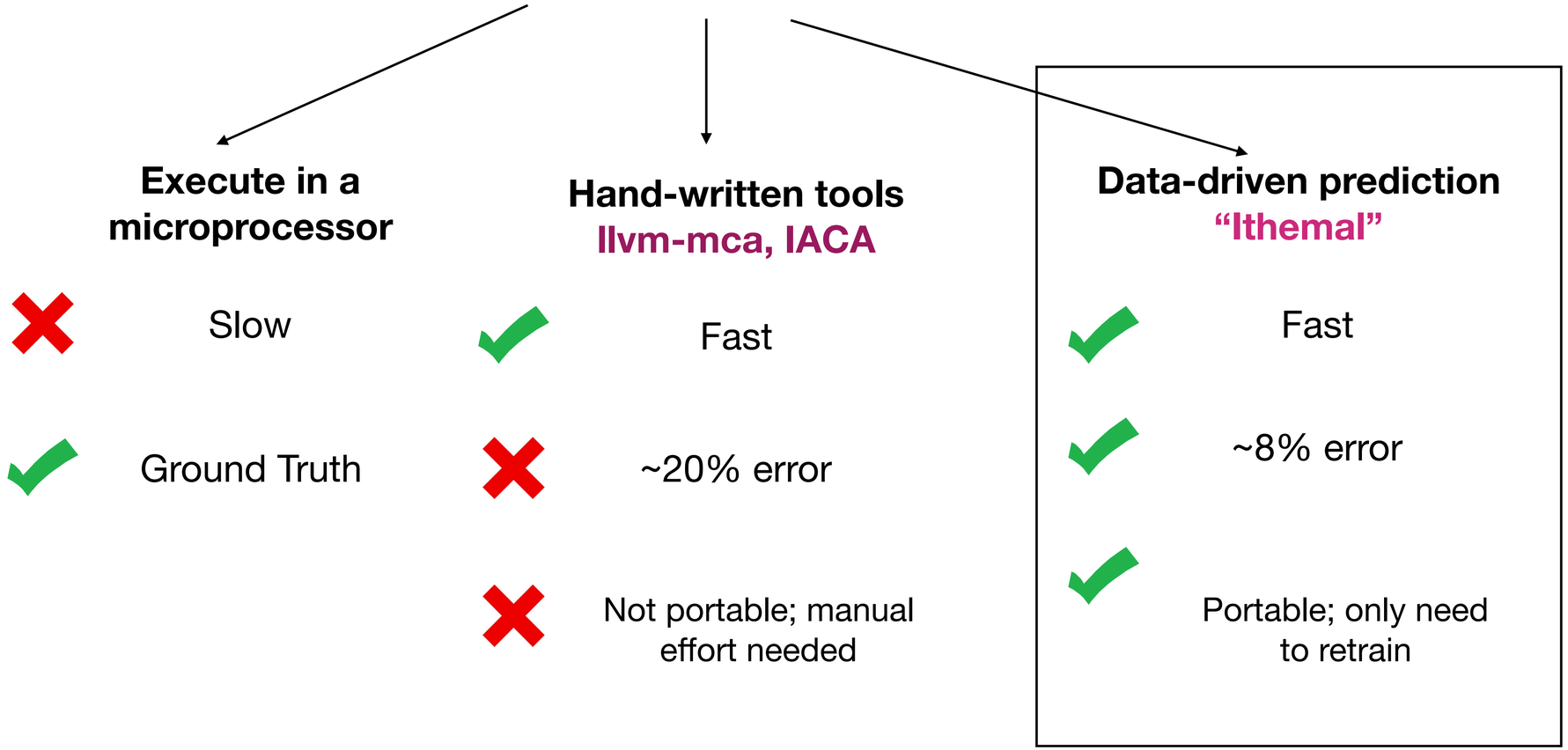
Analytical Model 2

≈

Analytical Model 3

Basic Block Throughput Estimation

```
mov ebx, [ecx]
add ebx, ecx
```



Motivating Example - Zero Idioms

llvm-mca

- Part of LLVM compiler infrastructure
- Uses industry standard compiler (LLVM) scheduling models
- e.g., more than >230 commits spread over 2 years for x86 Haswell Scheduling model

IACA

- Intel Architecture Code Analyzer
- Developed in-house at Intel

Ithemal

- Data-driven model

```
vxorps xmm0, xmm0, xmm0
```

100 iterations

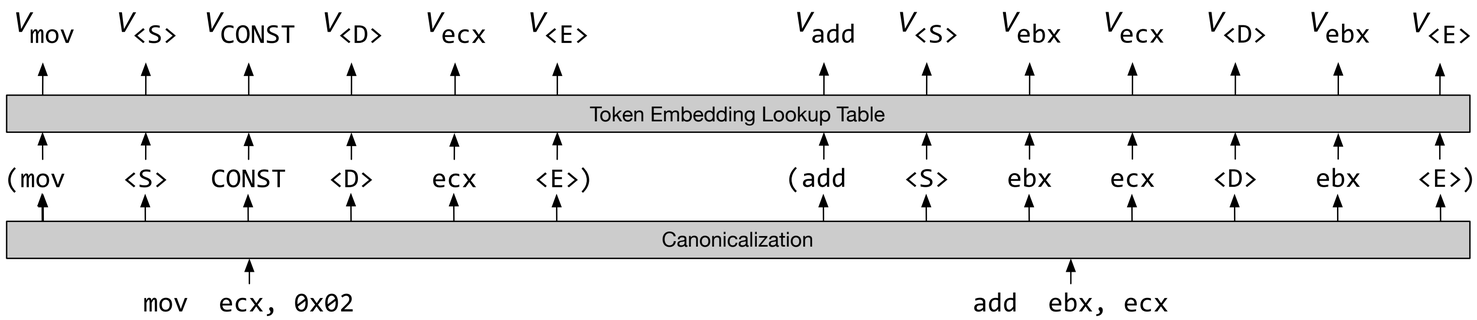
Method	Estimate
Measured	**32**
llvm-mca	100
IACA	24
Ithemal	**35**

Ithemal model architecture

Hierarchical Embeddings

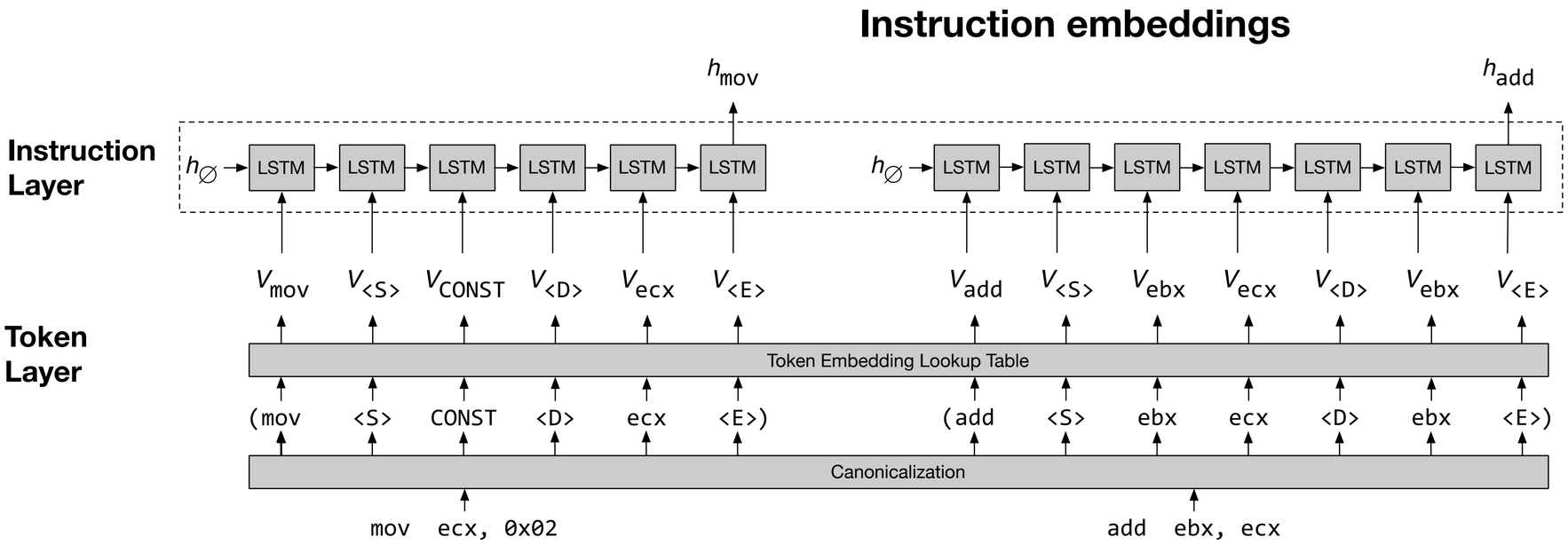
Token embeddings

Token Layer



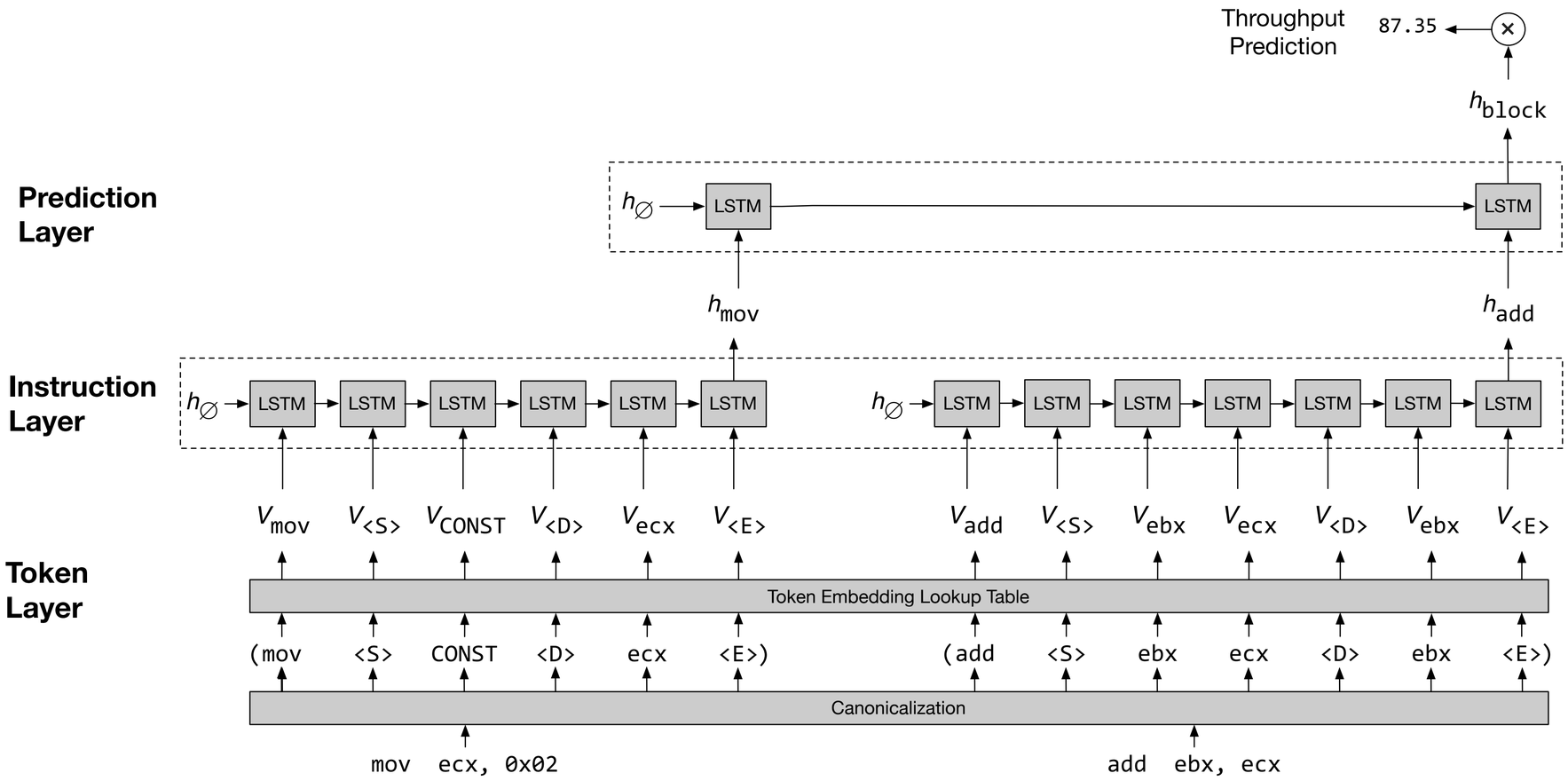
Ithemal model architecture

Hierarchical Embeddings

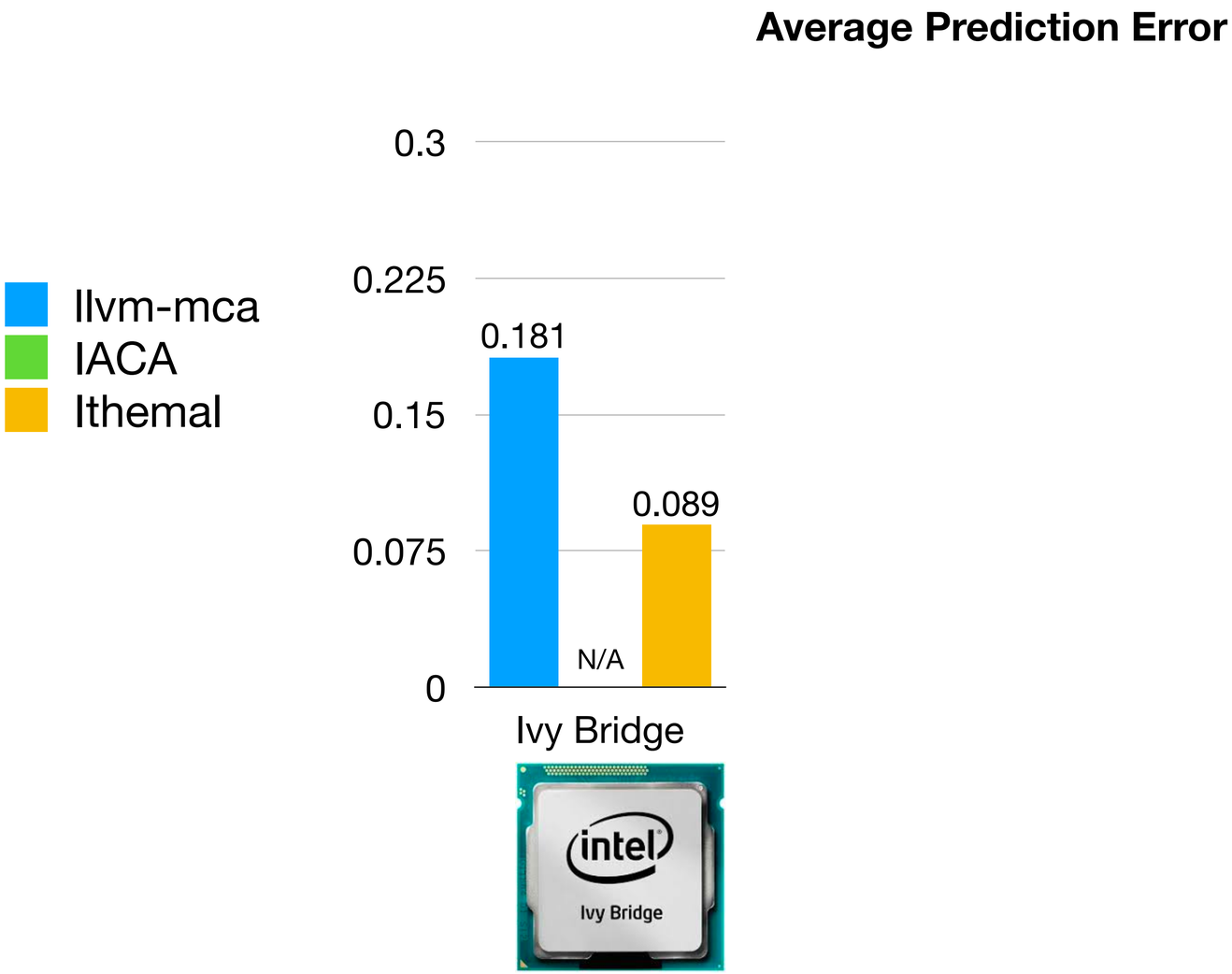


Ithemal model architecture

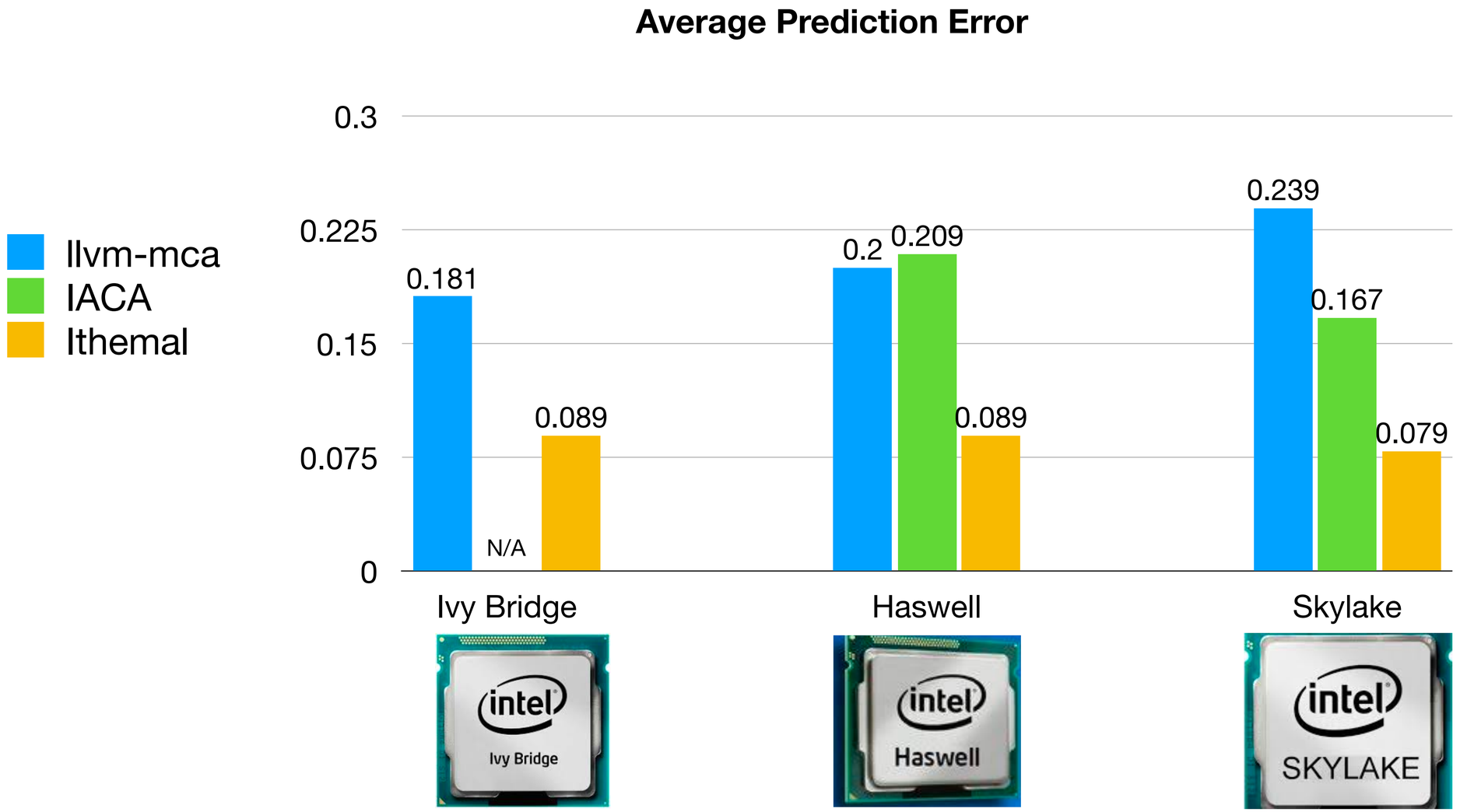
Hierarchical Embeddings



Ithemal halves the error rate

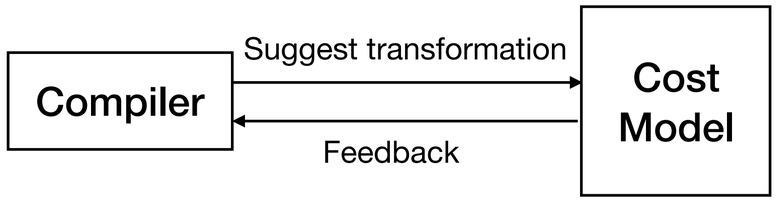


Ithemal halves the error rate across multiple microarchitectures



Conclusion and Future Work

- Potential to replace or augment traditional systems with data-driven counterparts
- Can Ithemal be made more robust?
- Continuous improvement in compiler optimization - cost model guided learnt optimizations



Download and use!

- Dataset
  - Over 1 Million-timed basic blocks
- Code **https://github.com/psg-mit/Ithemal**
- Try live demo **http://3.18.198.23/predict**

Come visit our poster

Today (Jun 11th Tuesday) from 06:30 to 09:00 PM
at Pacific Ballroom **#241**