Characterizing and Predicting Value Degree of Use

J. Adam Butts and Guri Sohi

University of Wisconsin–Madison

{butts,sohi}@cs.wisc.edu

MICRO-35

Istanbul, Turkey

November 20, 2002

Overview

Degree of use: number of times a dynamic value is used

- Indicates value's **communication characteristics**
- Predictable and exploitable
- Choose efficient communication method on a **per-value** basis

Degree of Use Prediction

- Low hardware cost, relaxed timing constraints
- **92%** of values receive correct predictions
- **<3%** misprediction rate

Outline

Overview

Degree of Use

- Motivation
- Characterization
- Predictability

Developing a Degree of Use Predictor

Evaluating Degree of Use Prediction

Summary

Value Communication

Observation: Value communication is expensive

- Large, multi-ported register files
- Complicated bypass networks
- Broadcast tag match for instruction wakeup

Why?: Communication structures are overly general

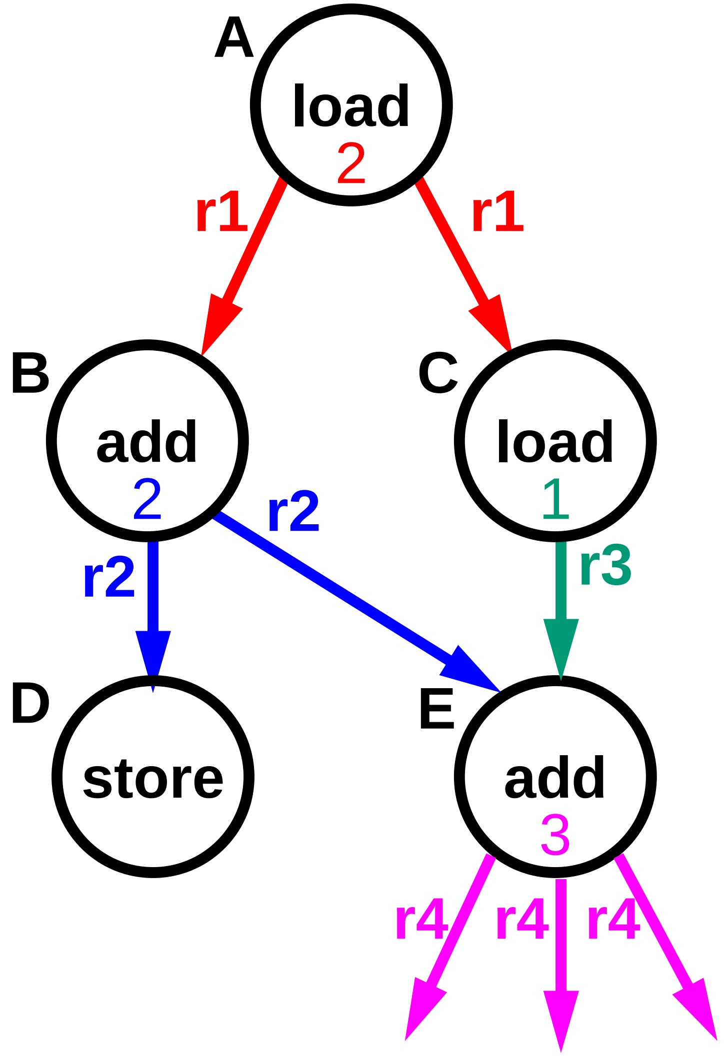
- Must support **all** possible communication patterns for **all** values
- Implicit assumption of complex communication patterns
- Optimize for the common case

How to determine the actual needs of a value?

Degree of Use

How many consumers does the value have?

- Direct indicator of the communication requirements of the value
- Answer: **degree of use**!



Focus on register degree of use

- **All** communicating instructions use at least one register
- Values not tracked through memory
  - Loads produce a **new** value
  - Stores produce **no** value

Applications of Degree of Use Knowledge

Degree of use: Zero

- Useless instruction elimination (avoid scheduling, execution)

Degree of use: One

- Collapsing dependent operations (intermediate value not needed)
- Direct consumer wakeup (no tag broadcast)
- Bypassing the register file (no writeback)

Degree of use: Few (< ~3)

- Selective instruction duplication (avoid cross-cluster communication)

Key: early knowledge of value's behavior

Characterizing Degree of Use

Degree of use statistics

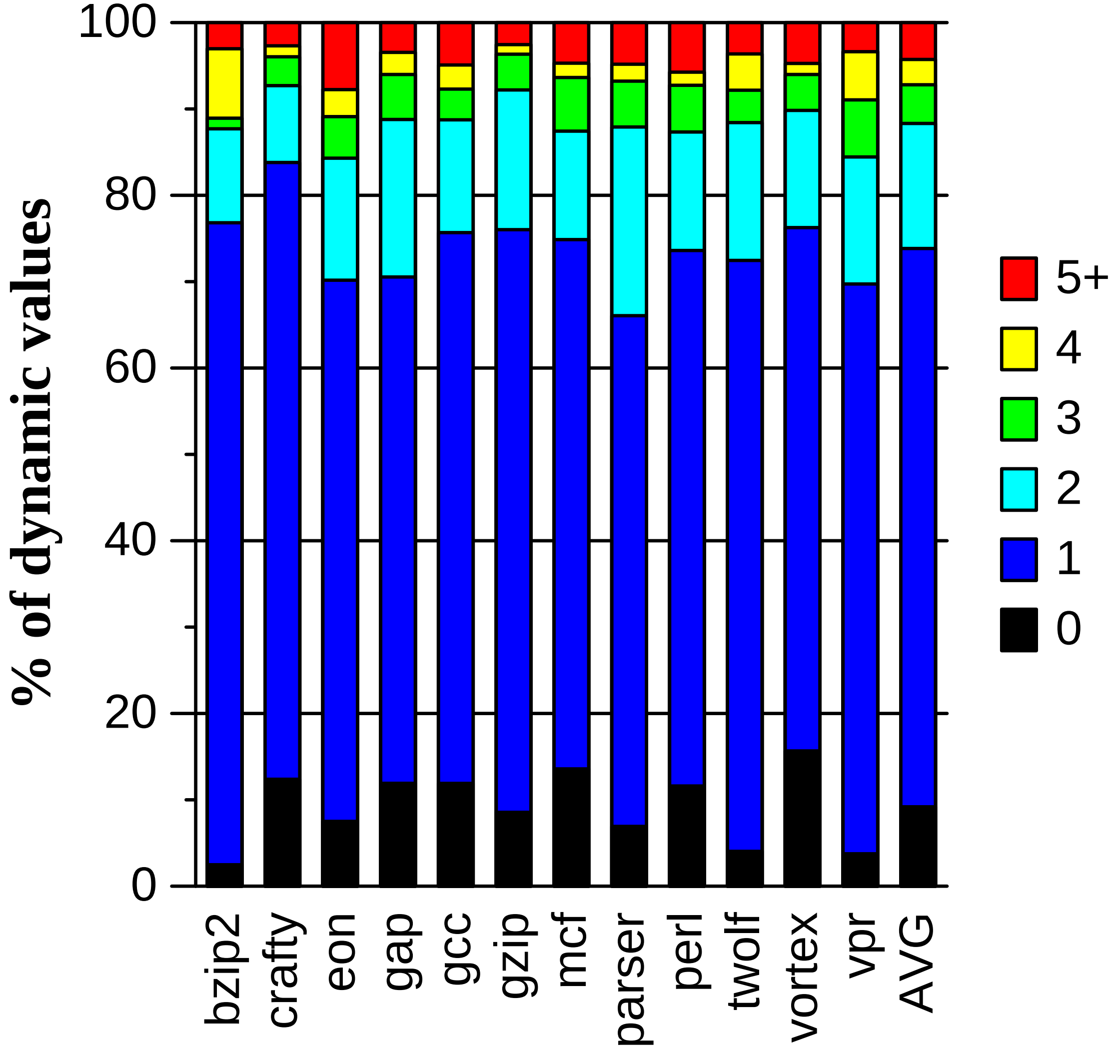
- Mode (most frequent): **1**
- Average: **1.66**
- Maximum: ~**330 M**

FP benchmarks

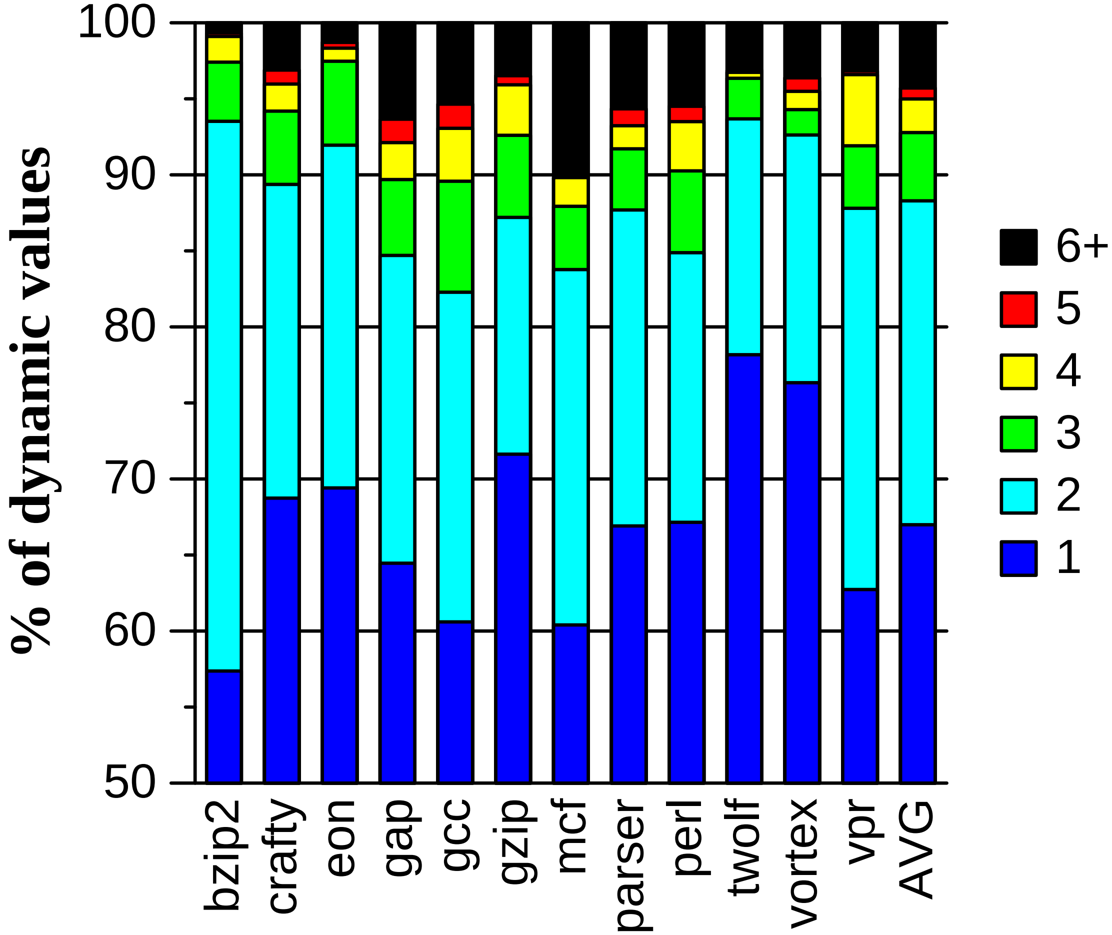
- Higher average: **1.83**
- Fewer 0, more 1, 2

Characteristics conserved

- gcc results similar
- Consequence of program structure and ISA



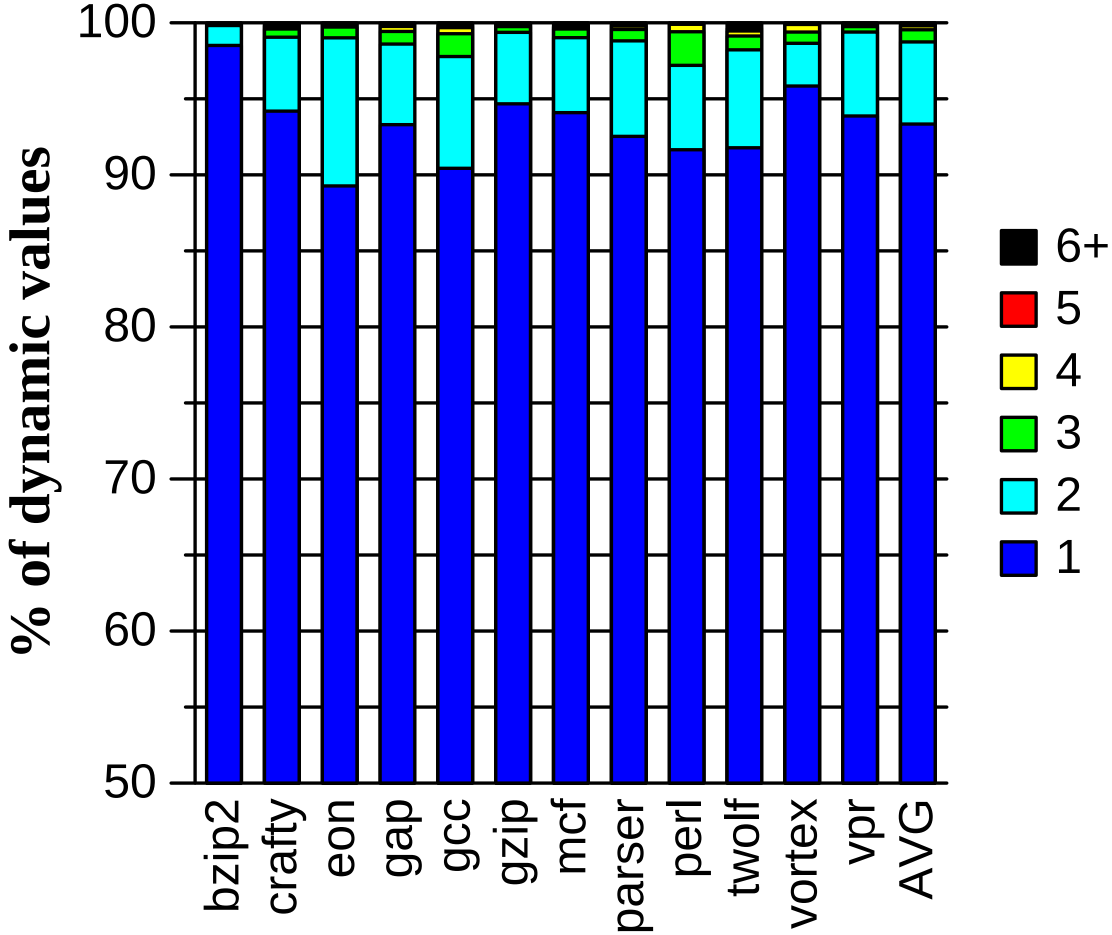
Predictability of Degree of Use



67% of values from instructions generating **one degree of use**

What about temporal locality?

Predictability of Degree of Use



93% of values have the same degree of use as the **last** value from the same instruction

Instruction identity is significant in determining degree of use

Outline

Overview

Degree of Use

Developing a Degree of Use Predictor

- Predictor organization
- Control flow signatures
- Predictor enhancements

Evaluating Degree of Use Prediction

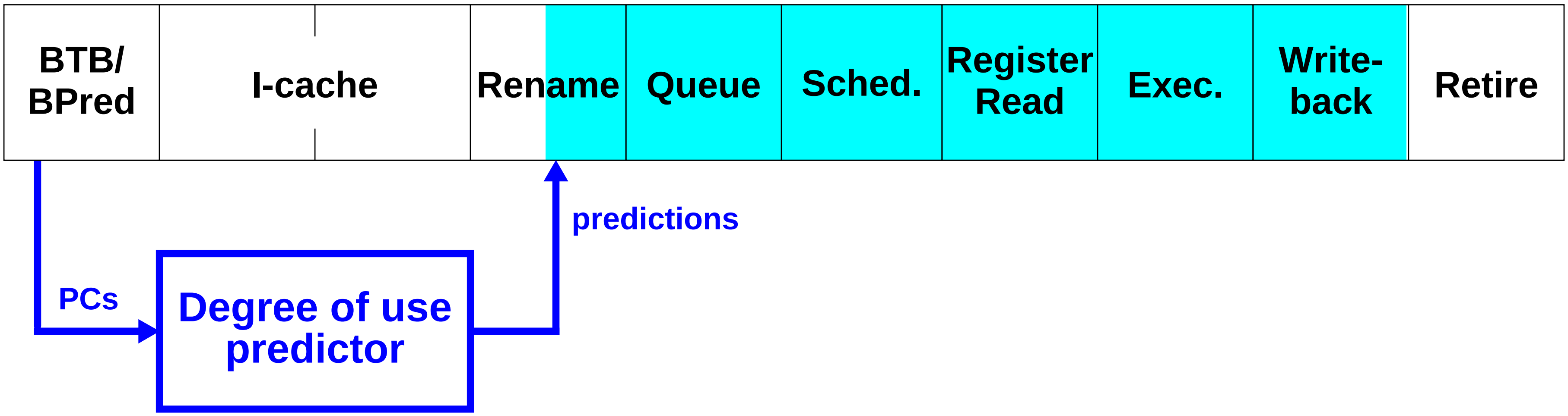
Summary

Degree of Use Prediction

BTB/ BPred	I-cache	Rename	Queue	Sched.	Register Read	Exec.	Write-back	Retire

Use predictions to optimize value communication

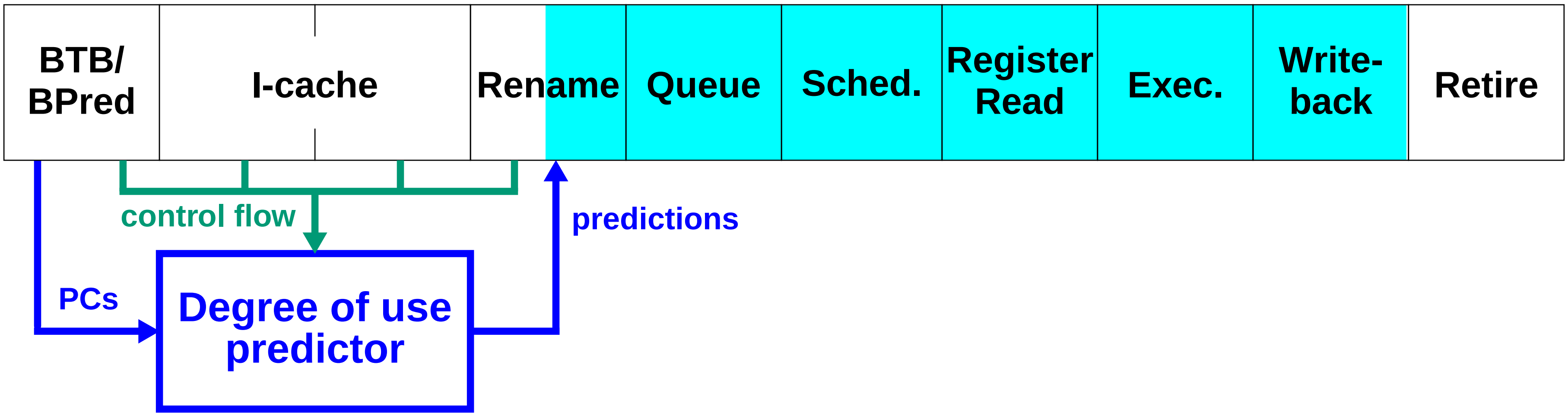
Degree of Use Prediction



Use predictions to optimize value communication

- Index predictor w/ instruction PC
- Receive timely predictions

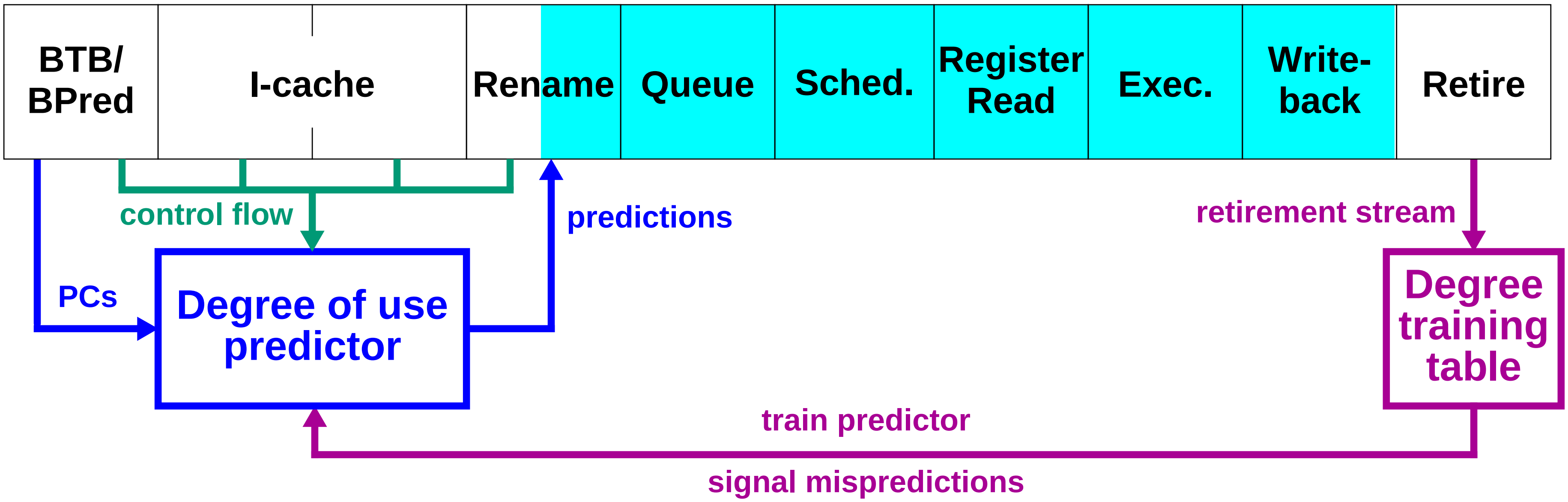
Degree of Use Prediction



Use predictions to optimize value communication

- Index predictor w/ instruction PC
- Receive timely predictions
- Exploit other pipeline information

Degree of Use Prediction



Use predictions to optimize value communication

- Index predictor w/ instruction PC
- Receive timely predictions
- Exploit other pipeline information

Observe instruction stream for training/misprediction detection

- Refer to paper

Basic Predictor

Associate static instructions w/ degree of use of last instance

- Cache-like predictor indexed with low-order instruction PC
- Parameters: capacity, associativity, tag length, maximum degree

Tag entries to reduce aliasing

- Use high order PC bits
- Associative organizations require tagging

Maximum predictable degree of use?

- Application dependent
- Small contribution from high degrees of use
- Group all predictions >= limit (6)

Multiple Degrees of Use

How to differentiate multiple possible degrees of use?

Future **control flow uniquely determines degree of use**

- All uses occur after value is generated
- Observed uses depend solely on which path is taken

Predicted future control flow is available

- Degree of use predictions needed in **middle** of pipeline
- Control flow predictions are made in **early** pipeline stage

Use a forward control-flow signature

- Proposed as part of dead-instruction predictor [ASPLOS-X]

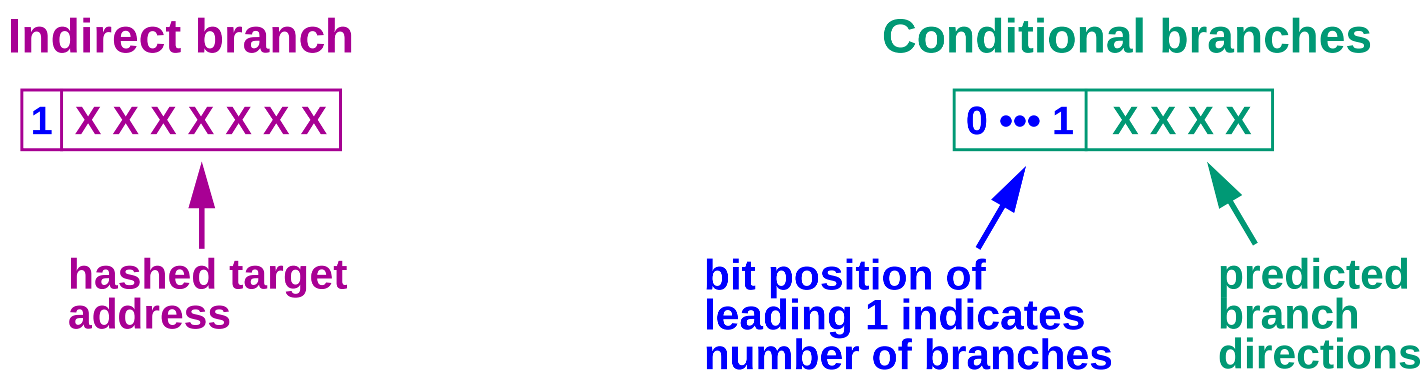
Forward Control Flow Signatures

Signature encodes predictions for upcoming indirect or conditional branches

- If next branch is indirect (e.g., return), encode **target address**
- Otherwise, encode available predicted branch **directions**
- Type and **number** of predictions is also encoded

Generating a prediction requires a PC and signature match

Signature Formats



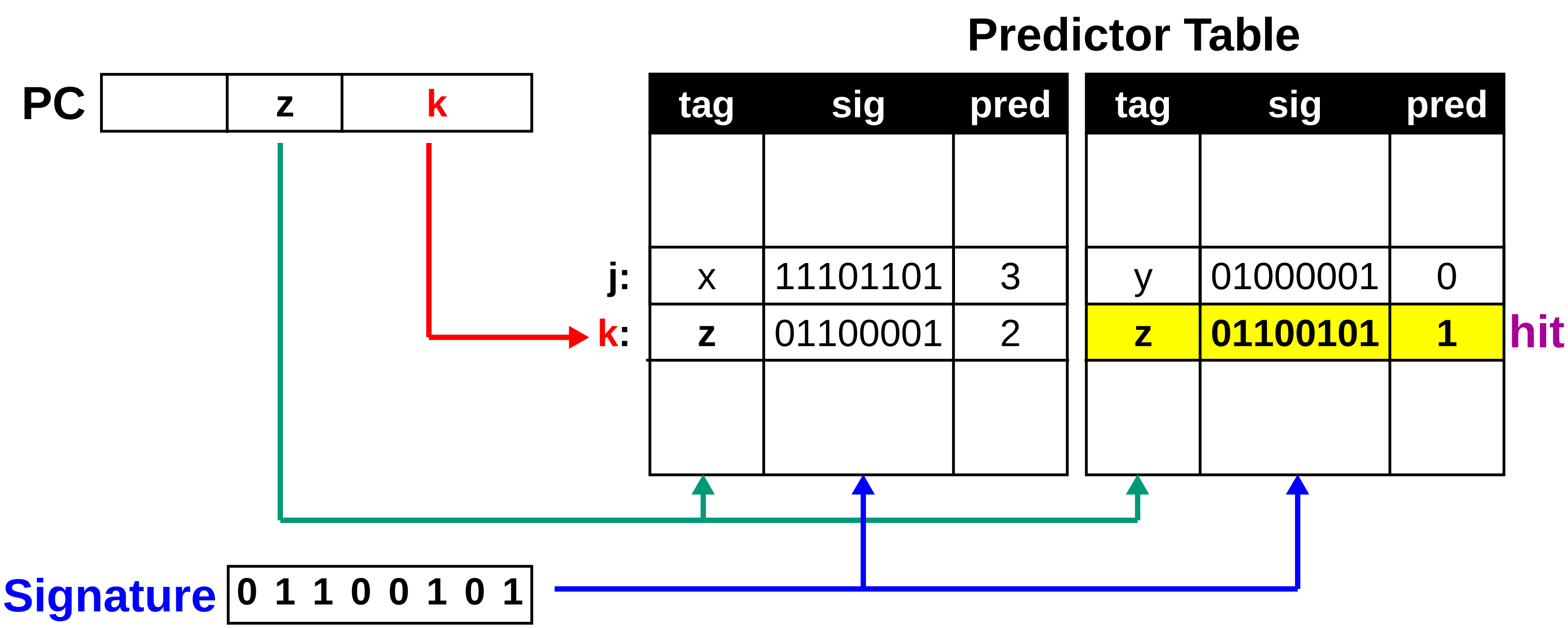
Predictor Microarchitecture

Associates predictions with instruction identity

- Index with low-order PC bits, tag entries with higher-order bits

Support multiple predictions per static instruction

- Use a set-associative predictor organization
- Use control flow signature as part of tag



Enhancements to Basic Predictor

Confidence bits

- 2-bit saturating counter per entry: ↑ on correct pred., ↓ on mispred.
- Hysteresis helps

Signature prefix matching

- Require match only to **length of stored signature**
- Assumes stored signature is “necessary and sufficient detail”

Alternative replacement algorithms

- True LRU costs **many bits**
- Not-MRU, random replacement

Default to maximum-predictable degree of use

- No explicit storage for this case: increases effective capacity
- Diminishing returns
- Needs of applications

Outline

Overview

Degree of Use

Developing a Degree of Use Predictor

Evaluating Degree of Use Prediction

- Methodology
- Parameter sensitivity
- Performance of enhancements

Summary

Methodology

Execution-driven simulation

- 4-wide fetch, issue, retire
- 256-entry ROB, 64-entry scheduling window
- 12 KB YAGS branch predictor, RAS, cascaded indirect predictor
- 64 KB 2-way set-associative L1 caches, unified 2MB 4-way L2

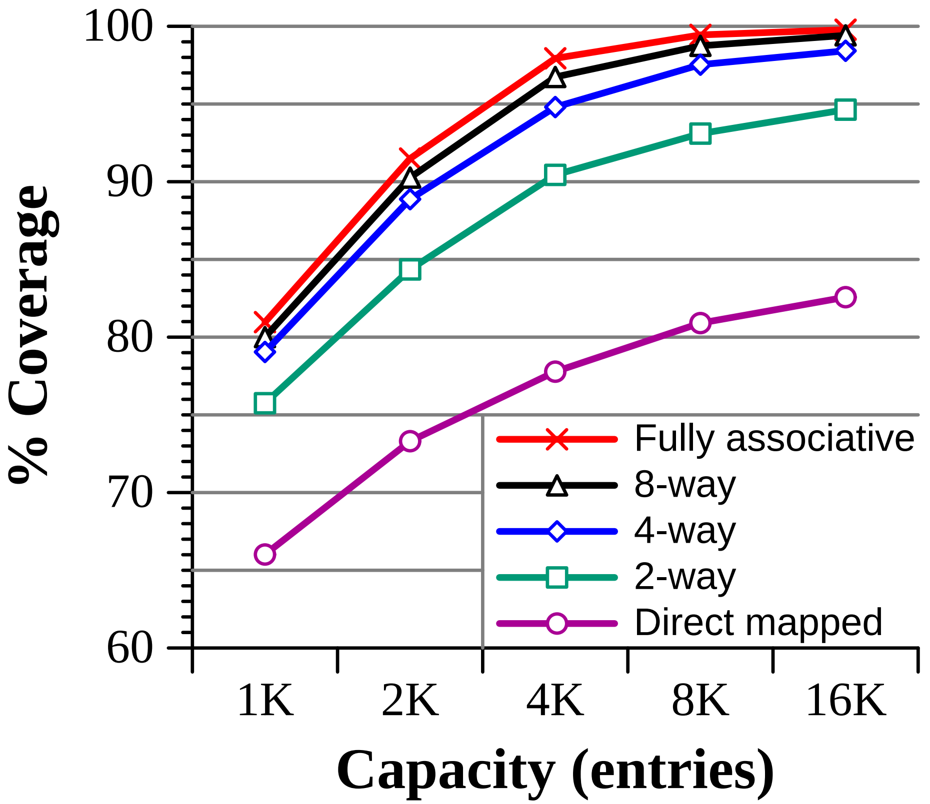
Coverage

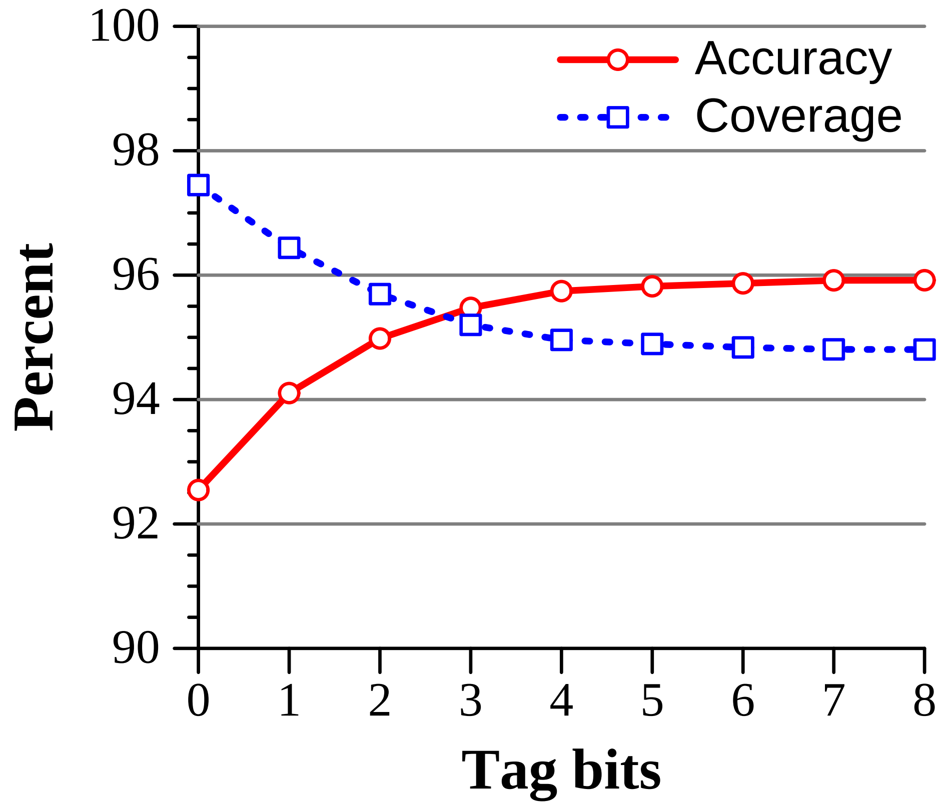
- Percentage of all values predicted
- Higher coverage = less lost opportunity

Accuracy

- Percentage of **covered** values correctly predicted
- Higher accuracy = fewer mis-predictions

Predictor Parameter Sensitivity





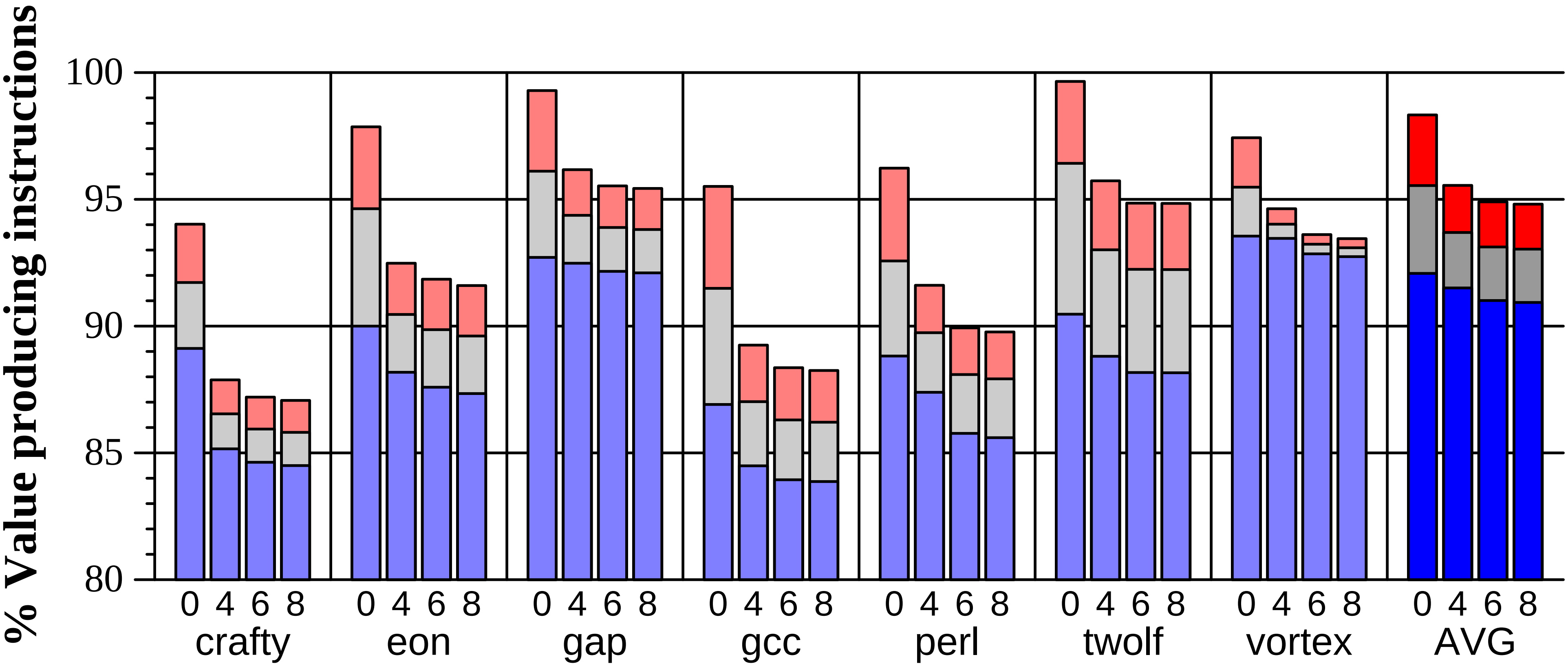
Coverage is a strong function of capacity and organization

- Similar to other caches
- Accuracy nearly independent

Increasing tag bits reduces destructive aliasing

- 6 bits OK for these programs
- Signature helps

Predictor Parameter Sensitivity



Control flow signature increases accuracy

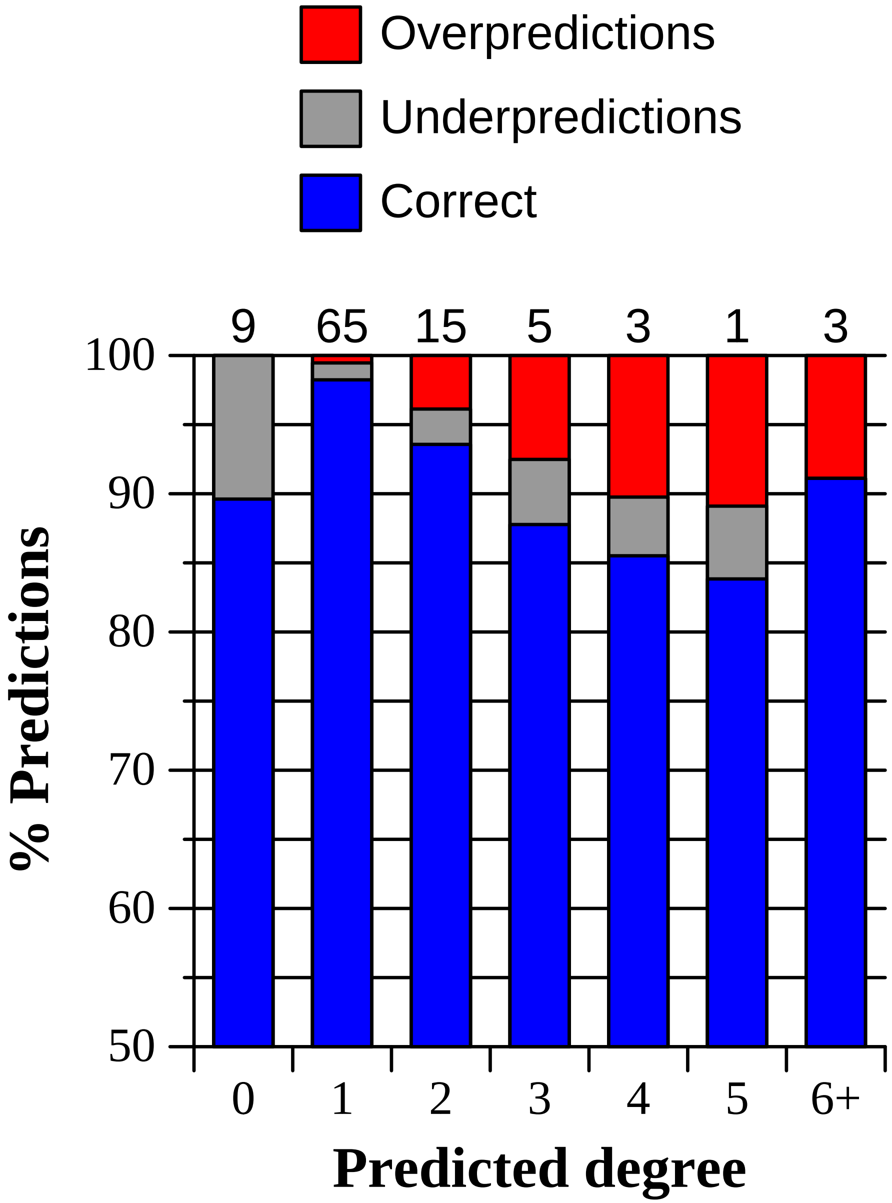
- Little benefit beyond 6 signature bits (== 4 branches)
- 98% of instructions have fewer than four branch directions available

Predictor Performance by Degree

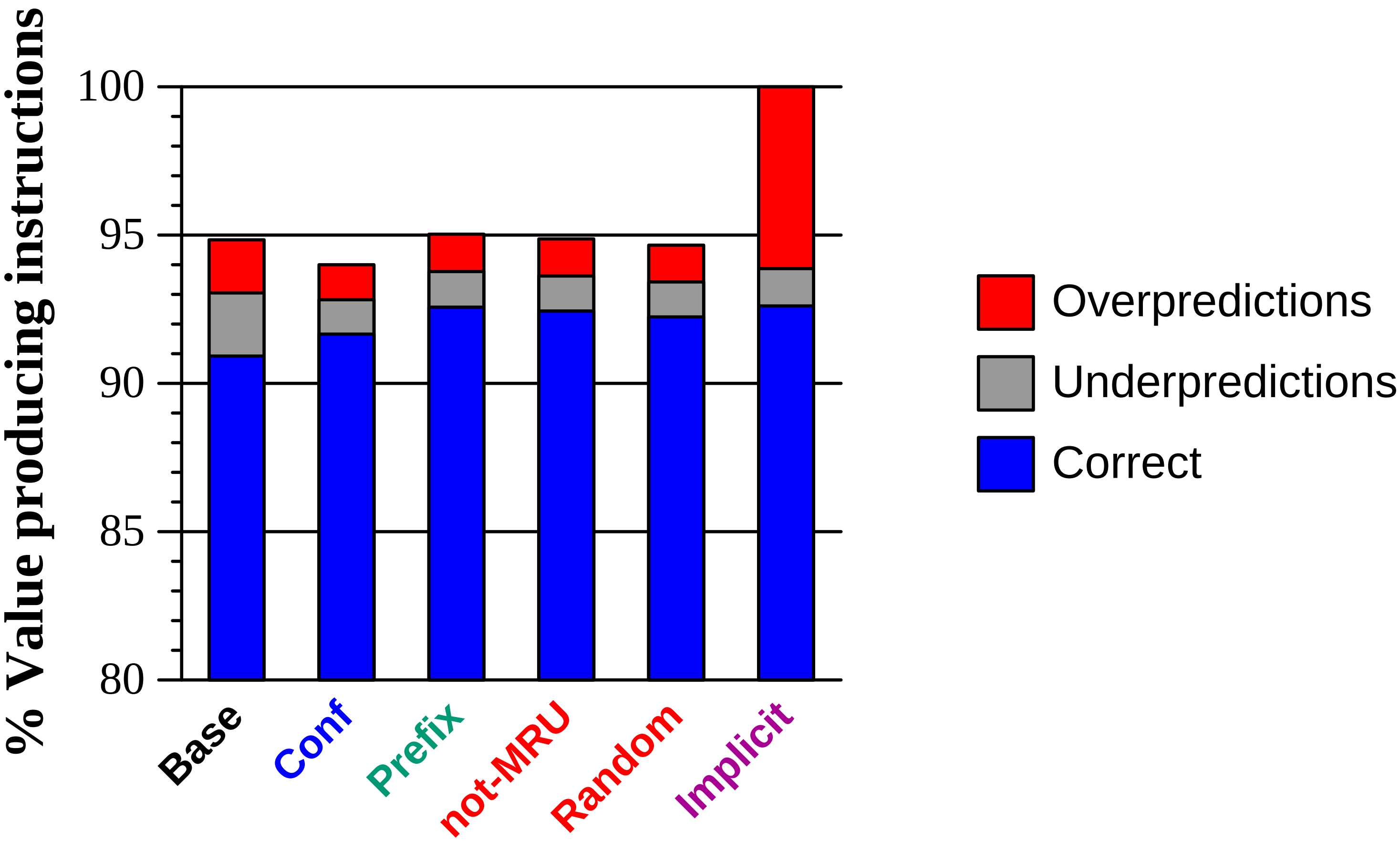
Prediction frequency reflects degree of use distribution

Accuracy depends on degree

- Accuracy diminishes with increasing predicted degree
- Degree of use 1 predictions
  - **Most accurate**
  - **Most mispredictions also**



Performance of Predictor Enhancements



Confidence bits, **prefix matching**: **big wins**

Alternative replacement (even random): reasonable cost

Implicit prediction: gain and cost are application-dependent

Outline

Overview

Degree of Use

Developing a Degree of Use Predictor

Evaluating Degree of Use Prediction

Summary

- Related work
- Conclusions

Related Work

Register traffic analysis (Franklin, Sohi)

- Initial examination of degree of use
- MIPS ISA, SPEC95 benchmarks
- Similar results
- Data used motivate many optimizations by other researchers

Analytical-statistical modeling (Eeckhout and Bosschere)

- Proposed power-law model for degree of use distribution
- Applied to microarchitectural modeling

Conclusions

Degree of use describes value communication

- Provides intuitive, direct knowledge of communication requirements
- Most values communicated to a small number of consumers

Accurate degree of use prediction is possible

- **92%** of values receive predictions with **<3%** misprediction rate
- Low overhead (8.5 KB), non-critical timing

Many potential applications

- Efficiency using mechanisms matched to actual communication
- Focus of our ongoing work

A degree of use predictor can be a key component of a communication-optimized architecture