Assigning Confidence to Conditional Branch Predictions

Erik Jacobsen, Eric Rotenberg[**], Jim Smith

Departments of Electrical and Computer Engineering and
[**]Computer Science
University of Wisconsin — Madison
http://www.cs.wisc.edu/~ericro/ericro.html

Introduction

- Branch prediction and speculative execution is becoming common in high performance processors
- Typically, all branch predictions are acted upon
- May not want to speculate if the likelihood of a branch misprediction is high
- More generally:

Vary how we act upon a branch prediction depending on the likelihood of a misprediction.

- Goal: develop hardware for assessing likelihood that a conditional branch prediction is correct
  - branch prediction confidence mechanisms

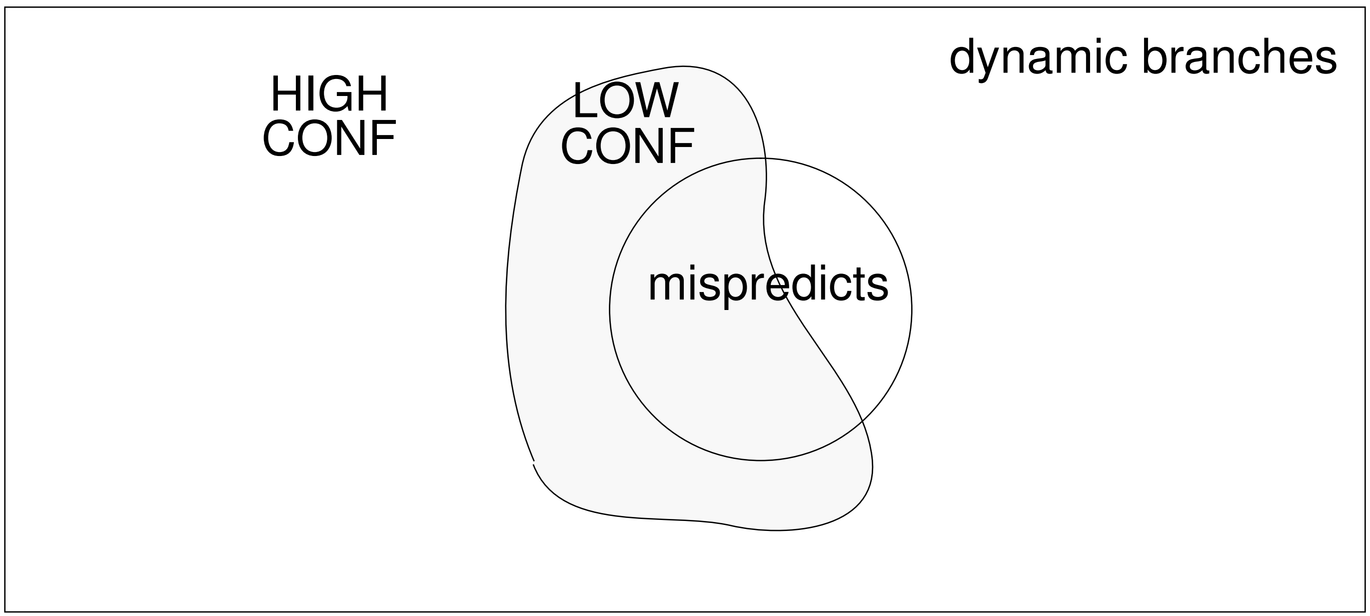
Applications

- Selective Dual Path Execution (Tim Heil)
- Guide instruction fetching in simultaneous multithreading
- Dynamic selector for N-way hybrid branch predictor (Jong-Hoon Shin)
- Branch prediction reverser (Karen Wells)

This talk: confidence mechanisms underlying these applications.

Objective

- Divide branch predictions into high and low confidence sets.
- Concentrate as many of the mispredicted branches as possible into the low confidence set, while keeping the low confidence set relatively small.



Simulation Methodology

- Trace-driven simulation using IBS benchmarks
  - both user and kernel code
  - benchmarks use X libraries
- Results are shown in aggregate
- Underlying predictor: gshare, 64K entries
  - ~4% aggregate mispredict rate
- Notes:
  - all results are ideal because they apply to the given input datasets only
  - choice of underlying predictor affects results
  - did not exhaust design space (must qualify “best”)

Static Confidence Experiment

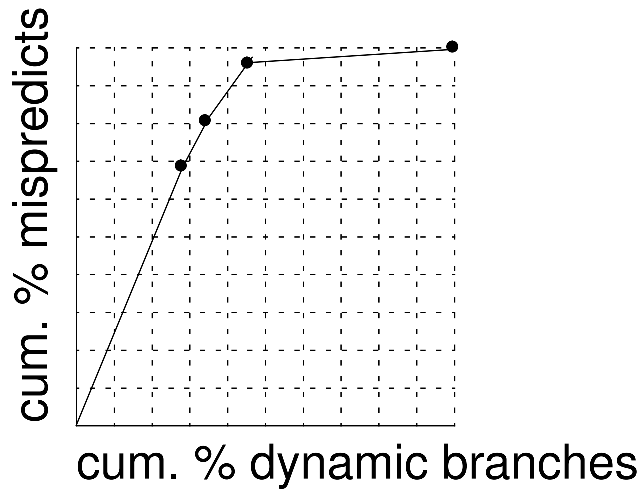
1. Profile every static branch:
   - count the number of dynamic occurrences
   - count the number of mispredicts

2. Sort the static branches from worst to best mispredict rate

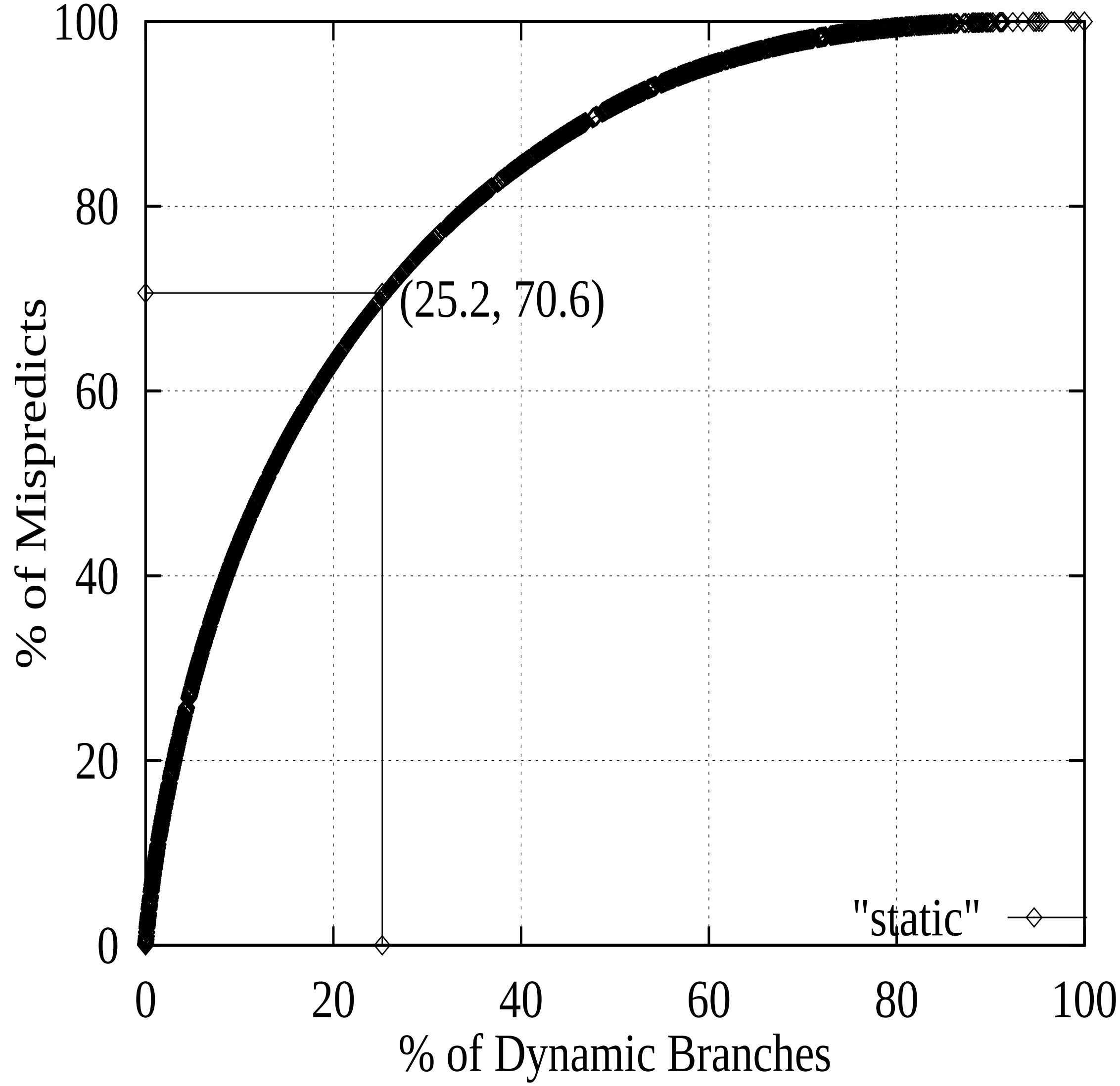
3. Going down the list of static branches, construct a graph:
   - Y-axis: cumulative % of total mispredicts
   - X-axis: cumulative % of total dynamic branches

Static Confidence (Example)

Address	occurrences: # / cum. %	mispredicts: # / cum. %	mispredict rate
`0x2020`	100 / 27%	98 / 69%	0.98
`0xAF04`	20 / 32%	18 / 81%	0.90
`0x1234`	50 / 46%	25 / 99%	0.50
`0x982F`	200 / 100%	2 / 100%	0.01



Static Confidence Experiment



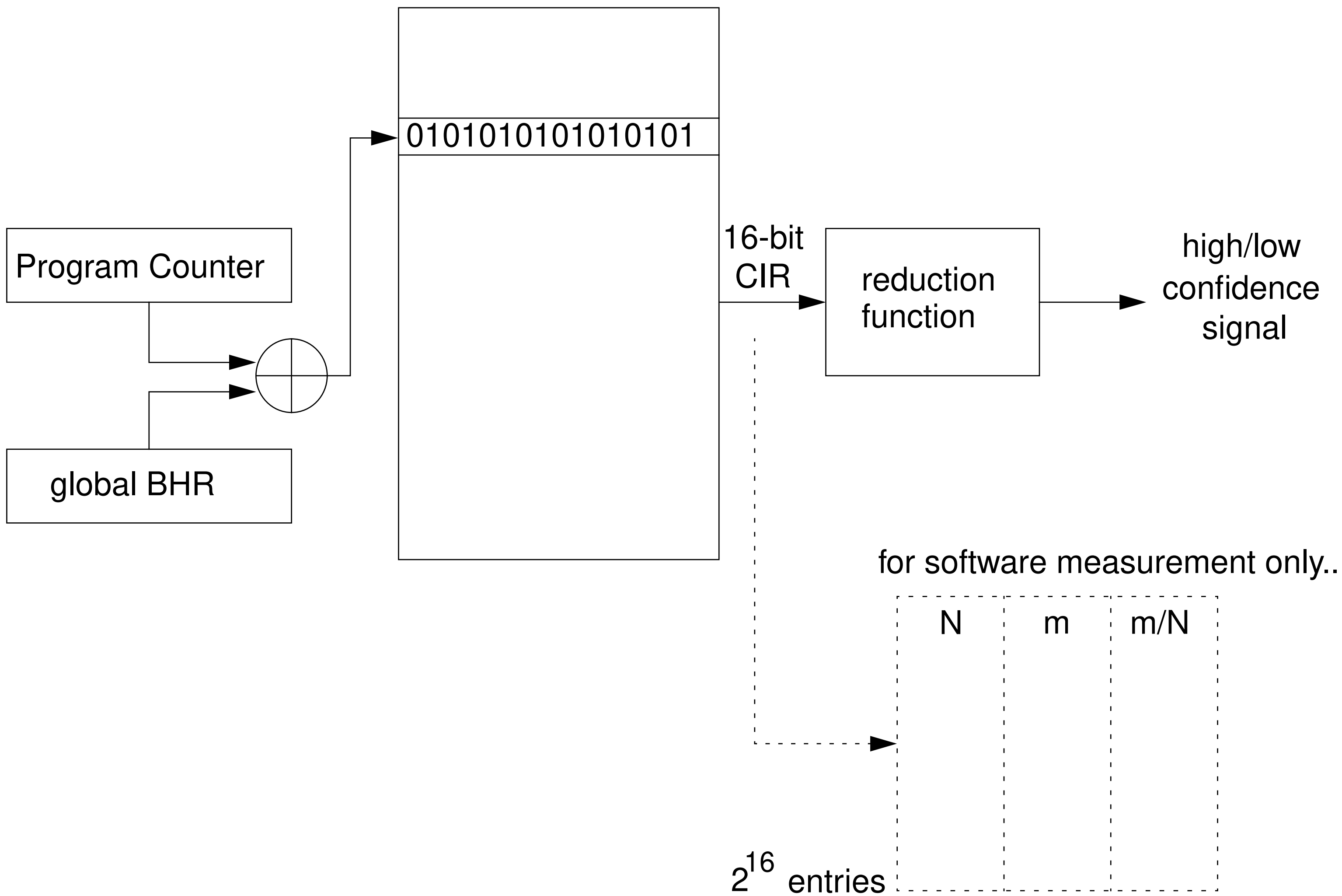
Dynamic Confidence Mechanisms

1. Index into a hardware table using branch PC.

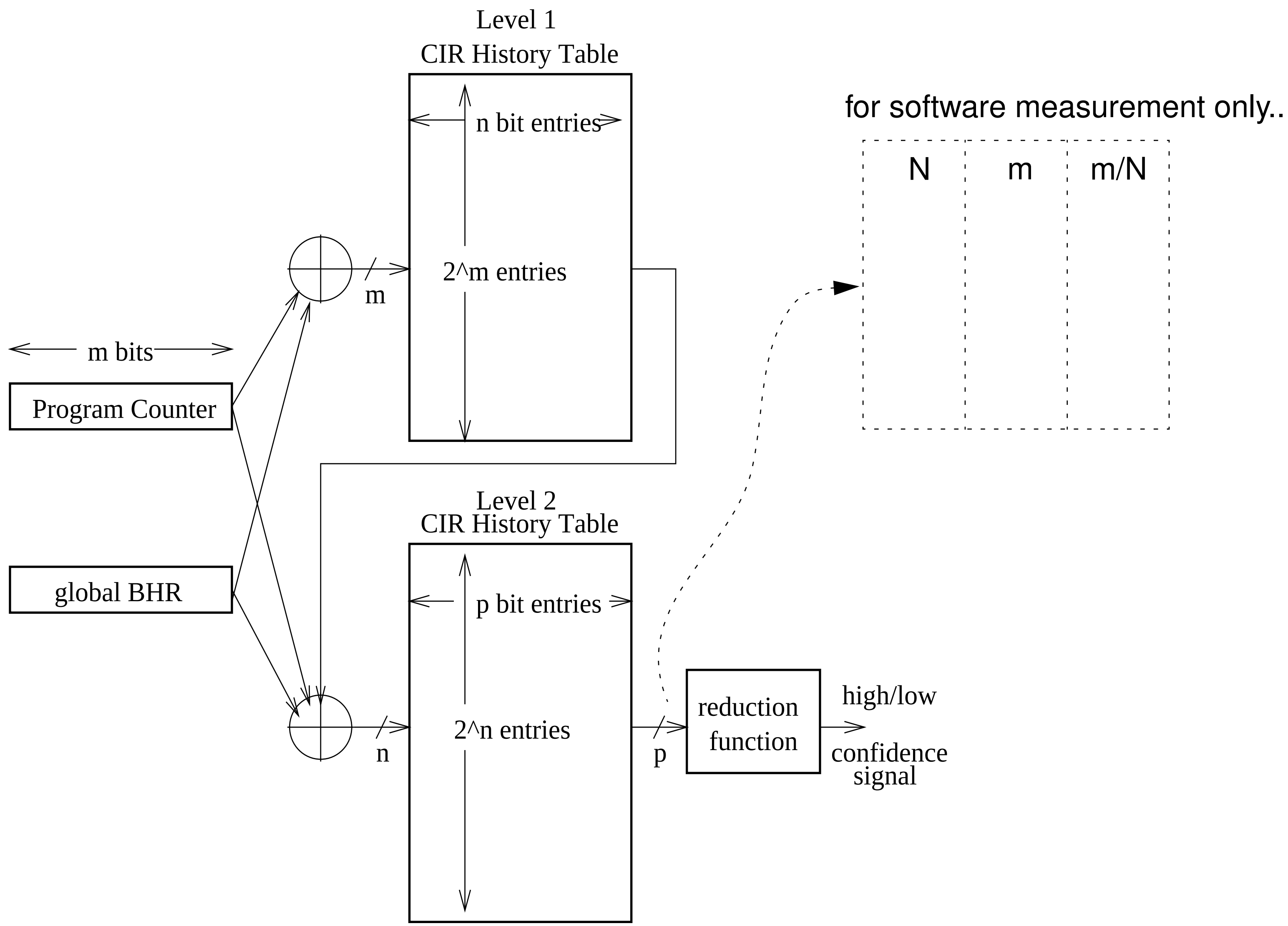
2. Each entry in the hardware table is a shift register containing history of prediction correctness:
   - CIR (“sir”): correct/incorrect register
   - on a mispredict, shift in a ‘1’; else shift in a ‘0’

3. Similar to static experiment, maintain a software table that keeps a profile for each possible CIR value:
   - count number of occurrences of each CIR value (N)
   - count number of mispredicts upon seeing that CIR (m)
   - construct graph similarly (sort CIRs, plot cum. %...)

Dynamic: Single-Level



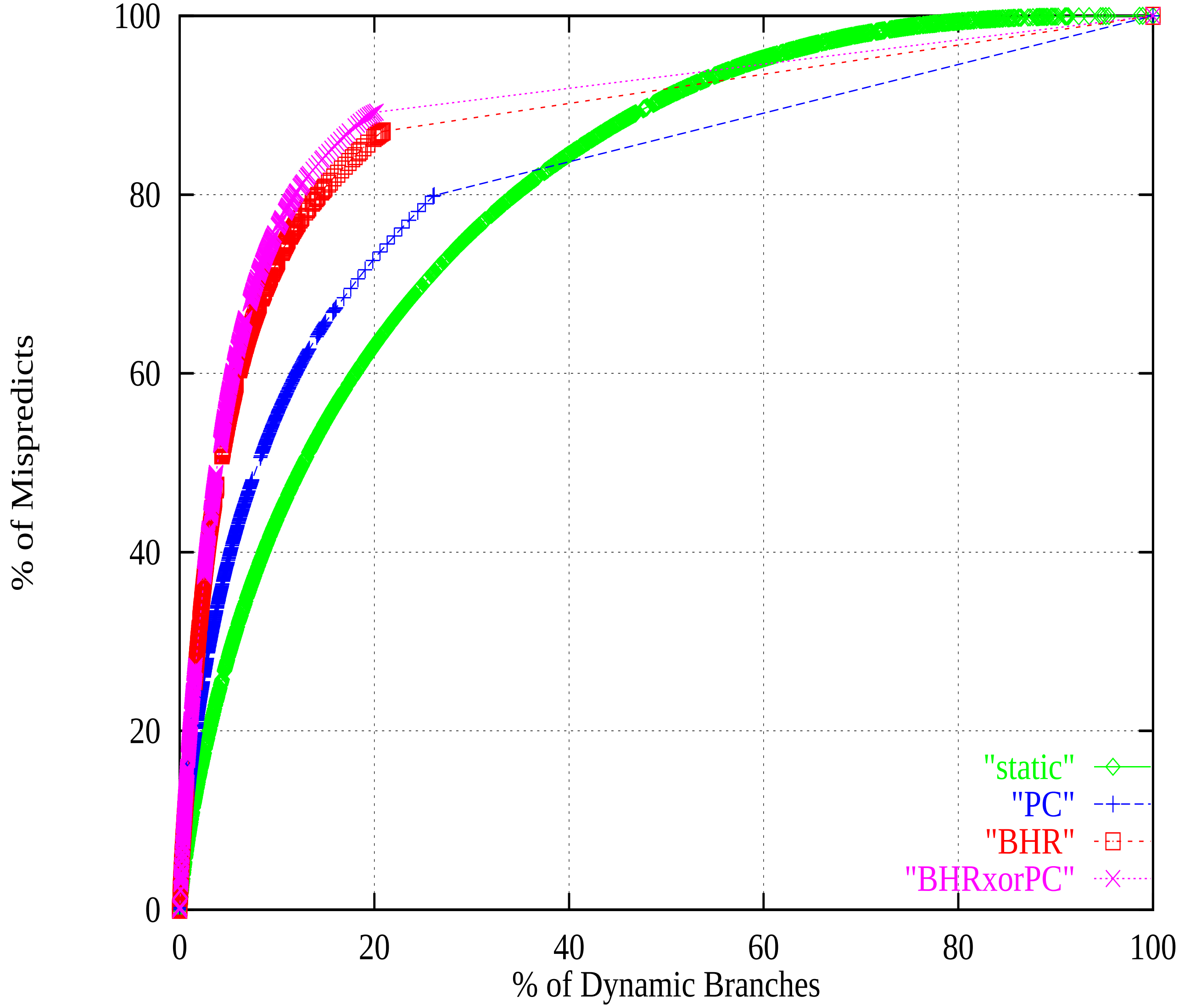
Dynamic: Two-Level



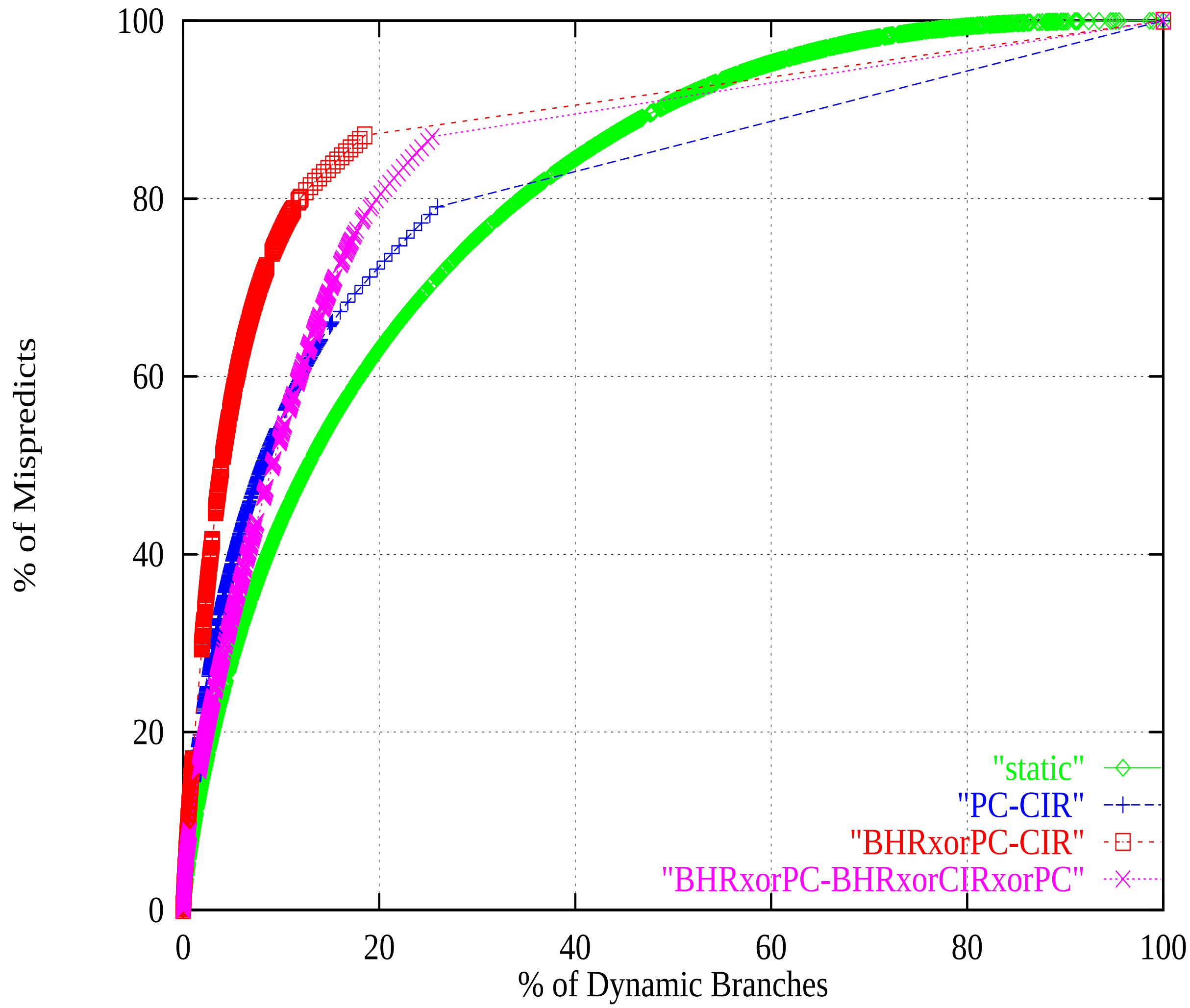
Parameters

- underlying predictor: 64K entry gshare
- CIR size (all levels) = 16 bits
- confidence table sizes:
  - first level: 64K entries
  - second level: 64K entries
- we vary indexing method, both levels
- Two sets of results:
  - limit study (“ideal”) => use CIR patterns
  - implementations => use reduction functions

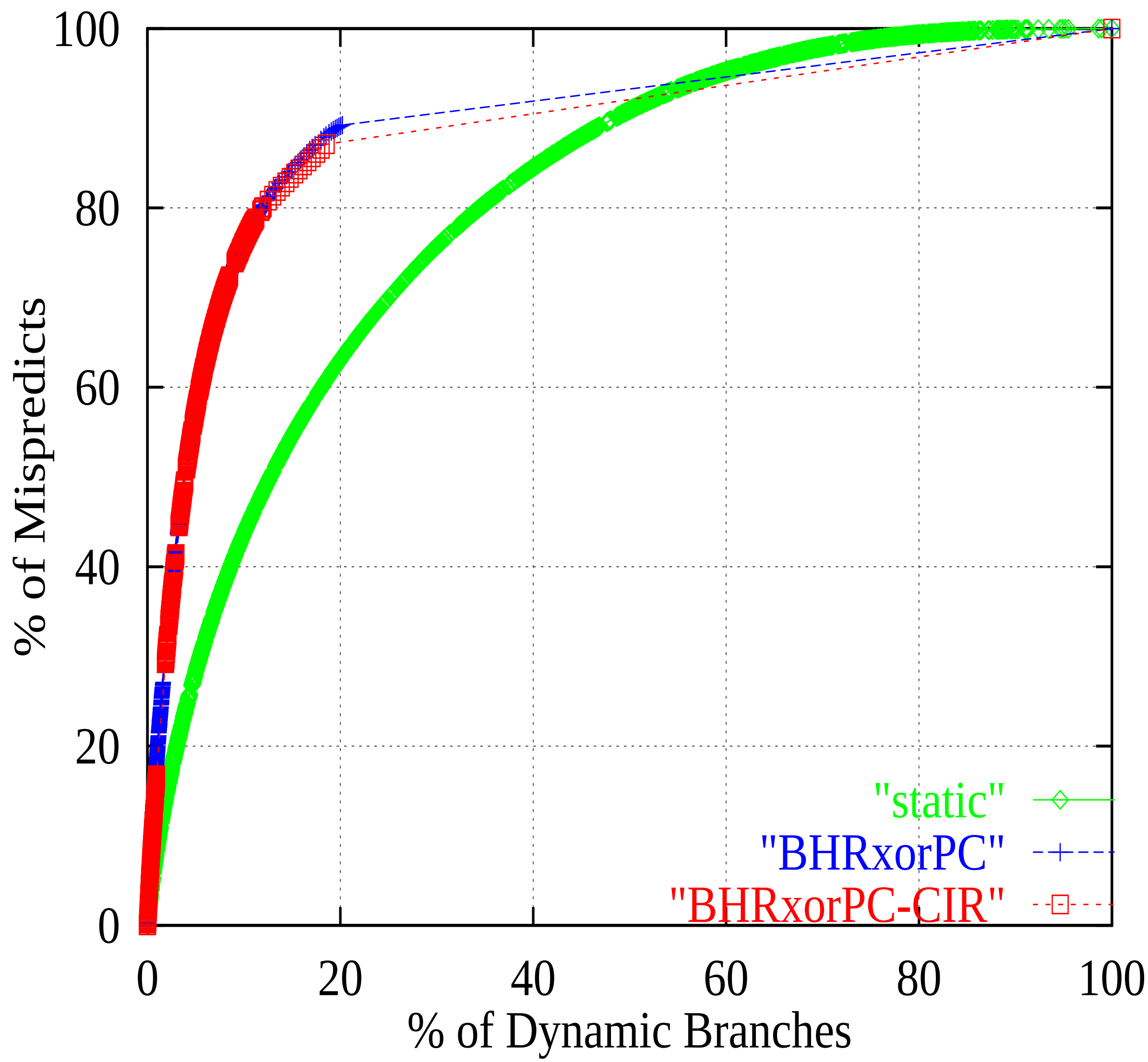
Single-Level Methods



Two-Level Methods



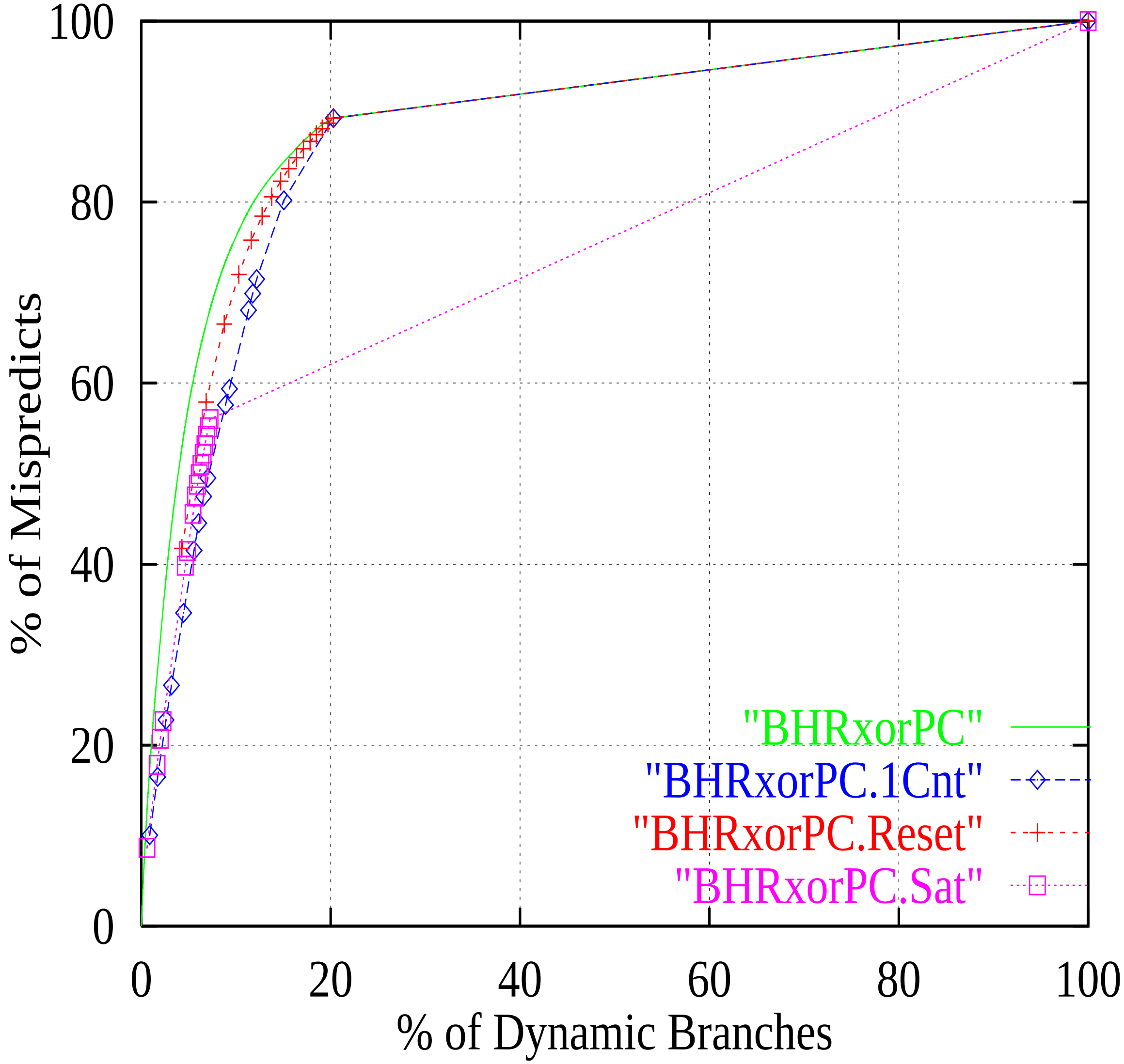
Best Single- and Two-Level



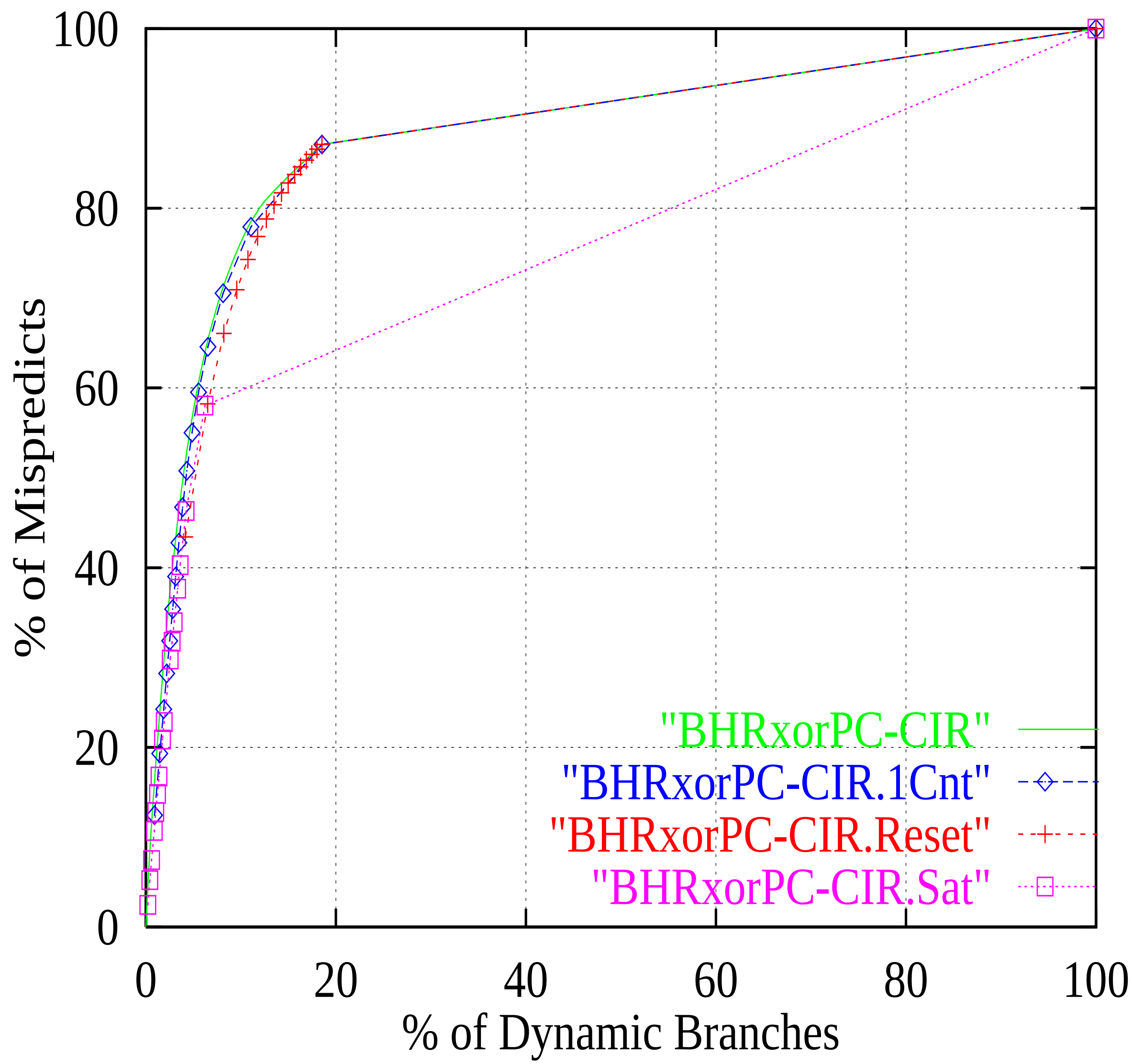
Real Implementations

- Reduction functions:
  - saturating counters
    Increment if correct, Decrement if incorrect.
  - resetting counters
    Increment if correct, Reset to 0 if incorrect.
  - ones counting
    Maintain CIRs in confidence table, count one bits.
- To limit graphs, we'll show only the "best" single- and two-level methods:
  - PC xor BHR
  - PC xor BHR / CIR

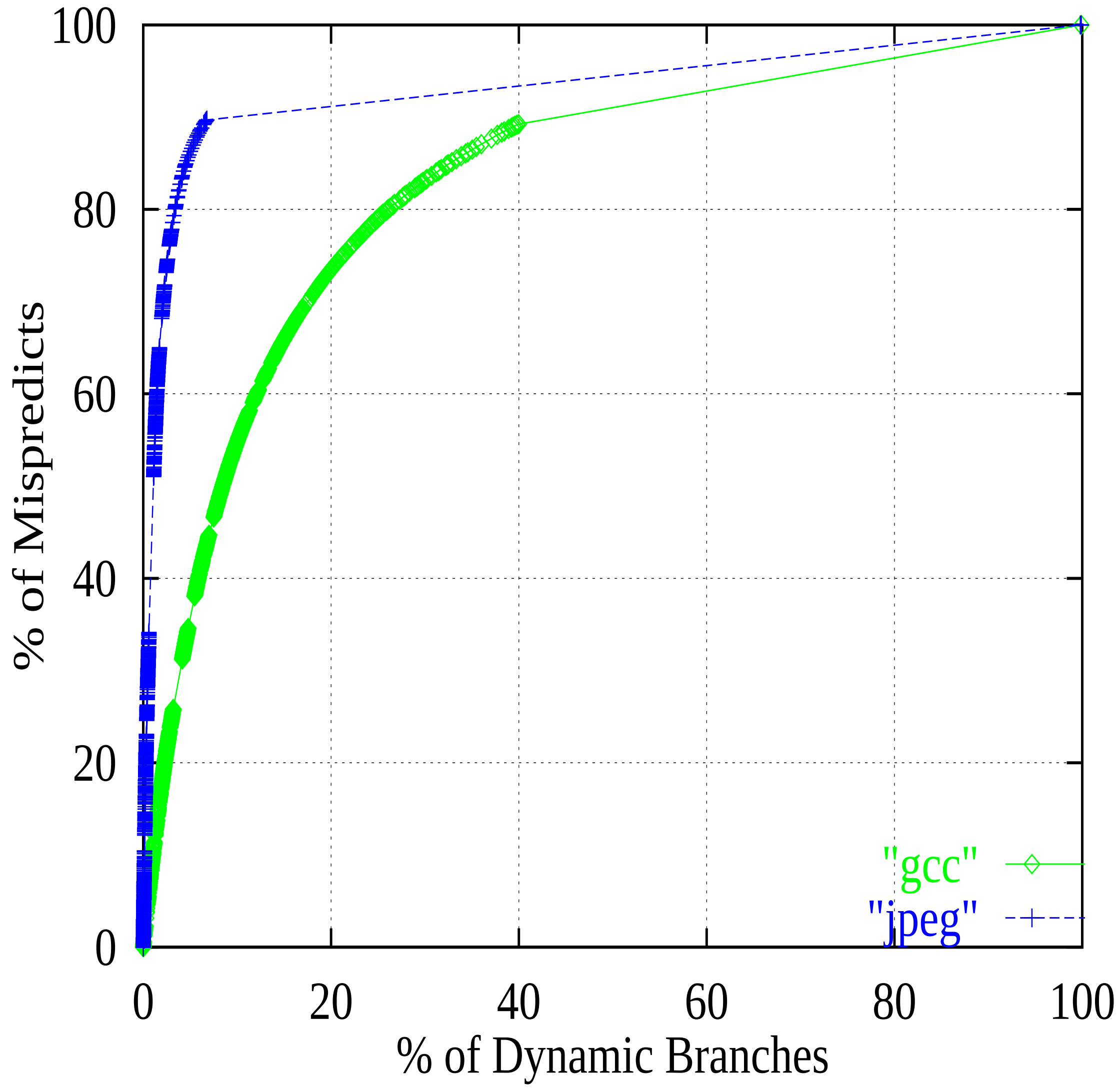
Single-Level, Real Implementations



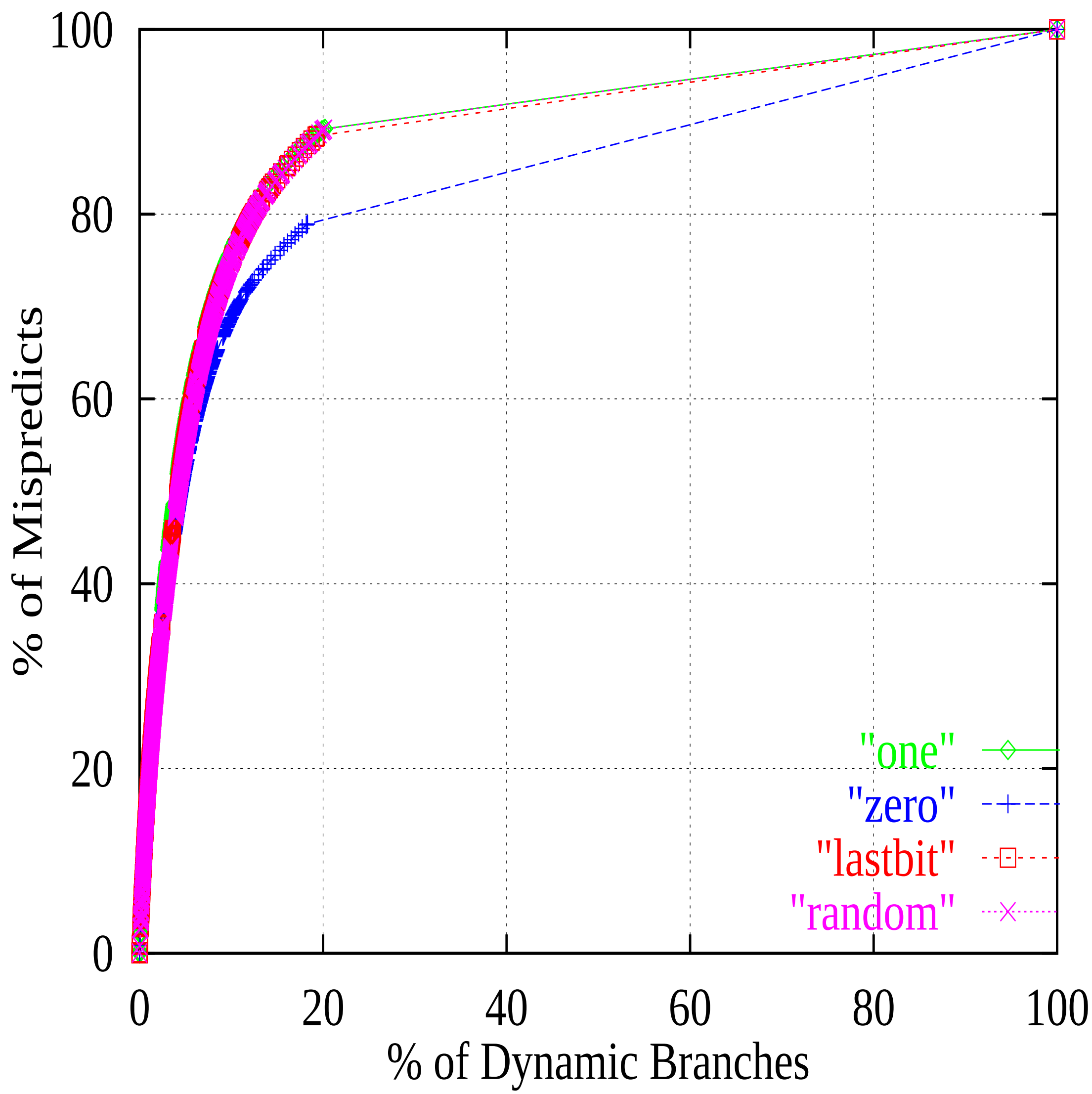
Two-Level, Real Implementations



Best/Worst Benchmarks



Effect of CIR Table Initialization



Summary

- Developed a technique for establishing high/low confidence sets (correct/incorrect history...)
- Can do much better than static confidence
- Introduced single-level and two-level dynamic schemes
  - CIR tables
  - found "best" indexing methods (BHRxorPC - CIR)
  - two-level gives little or no benefit over single-level
- Introduced reduction functions
  - resetting counters achieve close to ideal
- Result: a low confidence set of 20% covers 89% of all mispredicts

Using Predictor State for Confidence