Towards a More Efficient Trace Cache

Amit Saha
Jerry Yen
Rajnish Kumar
ELEC/COMP 525   April 24, 2001
Motivation

- Exploiting ILP
- Current limitations of instruction fetch mechanisms

Hypothesis

Trace cache implemented by:

- Giving weights to entries based on past use and future usage prediction (branch prediction) and
- Using the weights for the line fill and replacement buffer logic

will enhance processor performance
Architecture

- Trace Cache
  - 1024 or 32 entries
  - Max 3 blocks per entry
  - Max 16 instruction per entry
Branch Predictor

- Two Level Adaptive Branch Predictor
Weight Parameters

- Number of basic blocks
- Non-contiguity of the line
  - Zero-count in branch-prediction values
- Expected future use
  - 2-bit hit counter
  - Active-window-size field
Implementation

- Separate fields for different parameters
- Total weight of trace cache line is sum of
  - Basic_block_count weight
  - Branch prediction values mapped to weights
  - Number of hits in last $x$ number of cycles
    - $x$ is active_window_size.
Redundancy in Trace-Cache

- Line-fill-buffer logic changed:
  - If a block is the point of multiple entry, like B here, start a new trace cache line with B.
Implementation

Example:

\[ [ABC] \rightarrow [ABC, DE] \]

\[ [ABC, DE, BCD] \]

\[ [BCD] \]
Methodology

- Baseline
  - Increased execution resources
- Baseline with TC
- Baseline with modified TC
- Unmodified Trace Cache
  - LRU replacement policy
Ideal case

Possible IPC Improvement with 1024 Entry Trace Cache
Small sized trace cache

IPC Improvement With 64 Entry Trace Cache

SPEC2000 Benchmark

ammp  mcf*  vpr  mean

Baseline
TC(1024)
TC(64)
Possible IPC Improvement with 1024 Entry Trace Cache

Test Benchmark:
- anagram
- test-fmath
- test-llong
- test-lswlr
- test-math
- test-printf
- mean

IPC

Baseline
TC(1024)
Small sized trace cache

IPC Improvement with Trace Cache

Test Benchmark

- anagram
- test-fmath
- test-llong
- test-lswir
- test-math
- test-printf
- mean

IPC

Baseline
TC(1024)
TC(32)
Various weights used

IPC Improvement with Trace Cache Using Various Weights

Test Benchmark

- anagram
- test-fmath
- test-lon
- test-lswlr
- test-math
- test-printf
- mean

IPC

- TC(32)
- TC(32) + Distance Weight
- TC(32) + Weights
Modified LFB logic

IPC Improvement with Modified Line-Fill Buffer Logic

- TC(32) + Weights
- TC(32) + Weights + Modified Line-Fill Buffer Logic

Test Benchmarks:
- anagram
- test-fmath
- test-llong
- test-lswlr
- test-math
- test-printf
- mean
In conclusion

- Fetch Q is the bottleneck
- Hypothesis partially valid
  - Better results for Spec2000?
- Better combination of proposed weights?
- New weights?
- Same weights to work across multiple benchmarks?
Learning experience

- Difficult to increase IPC beyond what a base trace cache offers.
- How to proceed with such research projects
- Why man-months are so important in architecture research?