Five poTAGEs and a COLT for an unrealistic predictor

Pierre Michaud

June 2014
Competition track:

Unlimited size
I did not modify the predictor after the submission
Two-level history branch predictors

First level = context

Second level

E.g., global branch history, local branch history

branch address

prediction

E.g., TAGE
PPM-like second level

• Search the longest context that already occurred at least once, and predict from the past history for that context
  – search with the maximum context length L1
  – if no past occurrence for L1, search with L2 < L1
  – if no past occurrence for L2, search with L3 < L2
  – and so on...

• One table per context length

• To know if a context already occurred, use tags
  – false hit probability divided by 2 every time we increase the tag length by 1 bit
TAGE

- PPM-like (TAgged) with GEometric context lengths
  - does not name a specific predictor but a predictor family

- Most of the tricks are in the update
  - allocation policy, \(u\) bit, selection counter,...
  - makes the difference between bad TAGE (e.g., PPM-like 2004) and good TAGE
Let’s tune TAGE for limit studies
PPM’s main weakness:

the *cold-counter* problem
Biased-coin tossing game

• The coin is biased, we don’t know which side is the bias

• We play repeatedly with the same coin

• At game N+1, we count how many times head occurred vs. tail in the N previous games ➔ we choose the side which occurred the most
  – if equal head and tail counts ➔ choice = outcome of last game
Biased-coin tossing game

• The coin is biased, we don’t know which side is the bias

• We play repeatedly with the same coin

• At game N+1, we count how many times head occurred vs. tail in the N previous games ➔ we choose the side which occurred the most
  – if equal head and tail counts ➔ choice = outcome of last game

similar to TAGE’s taken/not-taken counters
Cold-counter problem

bias = 90%

<table>
<thead>
<tr>
<th>game</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>win proba.</td>
<td>0.500</td>
<td>0.820</td>
<td>0.820</td>
<td>0.878</td>
<td>0.878</td>
<td>0.893</td>
<td>0.893</td>
<td>0.898</td>
<td>0.898</td>
<td>0.899</td>
</tr>
</tbody>
</table>

bias = 60%

<table>
<thead>
<tr>
<th>game</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>win proba.</td>
<td>0.500</td>
<td>0.520</td>
<td>0.520</td>
<td>0.530</td>
<td>0.530</td>
<td>0.537</td>
<td>0.537</td>
<td>0.542</td>
<td>0.542</td>
<td>0.547</td>
</tr>
</tbody>
</table>
Cold counter problem in TAGE

- Limited storage ➔ allocate entry for longer context only upon misprediction

- ➔ counter likely to be initialized with least frequent outcome

- TAGE has a mechanism for reducing the cold counter problem
  - sometimes, second longest match entry more accurate than (cold) longest match entry
  - single global selection counter chooses between longest match and second longest
poTAGE: post-predicted TAGE

• TAGE tuned for limit studies

• Tackle cold counter problem

• Replace the selection counter with a post-predictor

• Aggressive update & allocation for fast ramp up
Selection counter ➔ post-predictor

• Selection counter is cost-effective, but does not solve the cold counter problem completely

• Post-predictor ➔ more effective solution
Post-predictor

TAGE

 ctr   ctr   ctr   u
3    3    3    1
third hit  second hit  first hit

1024 five-bit counters

T: increment
NT: decrement

T/NT prediction
Post-predictor

TAGE

ctrl   ctrl   ctrl  u
3      3      3      1
third hit  second hit  first hit

1024 five-bit counters

T: increment
NT: decrement

T/NT prediction

5% fewer mispredictions than selection counter
Ramp up

• Realistic TAGE ➔ careful policy allocates new entries only upon mispredictions
  – good use of limited storage by minimizing useless allocations

• poTAGE ➔ aggressive policy for reducing cold-start mispredictions
  – update all hitting counters
  – allocate for all context lengths greater than the longest hitting context and for which $u$ bit is reset
  – stop aggressive allocation for context lengths greater than 200 when all hitting counters are saturated
  – switch to careful policy after a fixed number of mispredictions
Ramp up

• Realistic TAGE ➔ careful policy allocates new entries only upon mispredictions
  – good use of limited storage by minimizing useless allocations

• poTAGE ➔ aggressive policy for reducing cold-start mispredictions
  – update all hitting counters
  – allocate for all context lengths greater than the longest hitting context and for which \( u \) bit is reset
  – stop aggressive allocation for context lengths greater than 200 when all hitting counters are saturated
  – switch to careful policy after a fixed number of mispredictions

4% fewer mispredictions
Global-path TAGE: footprint problem

• Global path, if long enough, can (in theory) capture all branch correlations

• Problem: high-entropy branches grow the footprint (number of allocations)

• We could try to filter out of the global path branches that carry no useful correlation information
  – in practice, difficult to identify these branches
  – filtering them out does not necessarily reduce the footprint

• Alternative approach: intentional path aliasing
Intentional path aliasing

• Path aliasing = several distinct global paths aliased to the same predictor entry and tag
  – something we try to avoid in a global-path TAGE

• Intentional path aliasing reduces the footprint
  – we lose some correlation information ➔ only some branches benefit from it

• Local history can be viewed as intentional path aliasing

• Per-set history (Yeh & Patt, 1993) is intentional path aliasing
  – was used in the FTL++ predictor (Yasuo Ishii et al., CBP-3)
multi-poTAGE

• Combine several poTAGE predictors using different first-level histories
  – $P_0$: 1 global path
  – $P_1$: 32 local (per-address) subpaths
  – $P_2$: 16 per-set subpaths (128-byte sets)
  – $P_3$: 4 per-set subpaths (2-byte sets)
  – $P_4$: 8 frequency subpaths

• Combined through COLT Fusion
  – Loh & Henry, PACT 2002

• Better to have a few long subpaths than many short ones
  – Yasuo Ishii et al., CBP-3
multi-poTAGE

P0 (global) → P1 (local) → P2 (per set) → P3 (per set) → P4 (frequency)

branch address → COLT

T/NT prediction
multi-poTAGE

P0 (global) → P1 (local) → P2 (per set) → P3 (per set) → P4 (frequency)

branch address → COLT → T/NT prediction
Frequency-based first-level history

• Branch frequency = number of times the branch was executed
  – Branch Frequency Table ➔ one counter per branch address
  – increment counter on each dynamic occurrence

• Exploit correlations between branches with (roughly) same frequency

• Define 8 frequency bins
  – from high to low frequency

• Associate one subpath with each frequency bin

• Access poTAGE with subpath corresponding to the branch frequency
Global path: most accurate single component

- $P_0$ (global)
Global path: most accurate single component
2nd most important: 128-byte sets

P0 (global) — P2 (per set) — COLT

branch address

-5 %
3rd: local

-3 %

-5 %

P0 (global)

P1 (local)

P2 (per set)

branch address

COLT
4th: frequency

- P0 (global): -3%
- P1 (local): -5%
- P2 (per set): -2.5%

branch address → COLT
5th: 4-byte sets

-3 %  
P0  (global)  

-5 %  
P1  (local)  

-1 %  
P2  (per set)  

-2.5 %  
P3  (per set)  

branch address  

COLT  

P4  (frequency)
Total

-10 %

P0 (global)  P1 (local)  P2 (per set)  P3 (per set)  P4 (frequency)

branch address

COLT
Conclusion

• Post-predictor more effective than selection counter for reducing cold-counter problem

• Huge TAGE can use aggressive update & allocation

• Fundamental weakness of global-path TAGE: high-entropy branches grow the footprint

• Proposed solution: blind use of intentional path aliasing

• Is it possible to use intentional path aliasing in a cost-effective way?
Questions ?