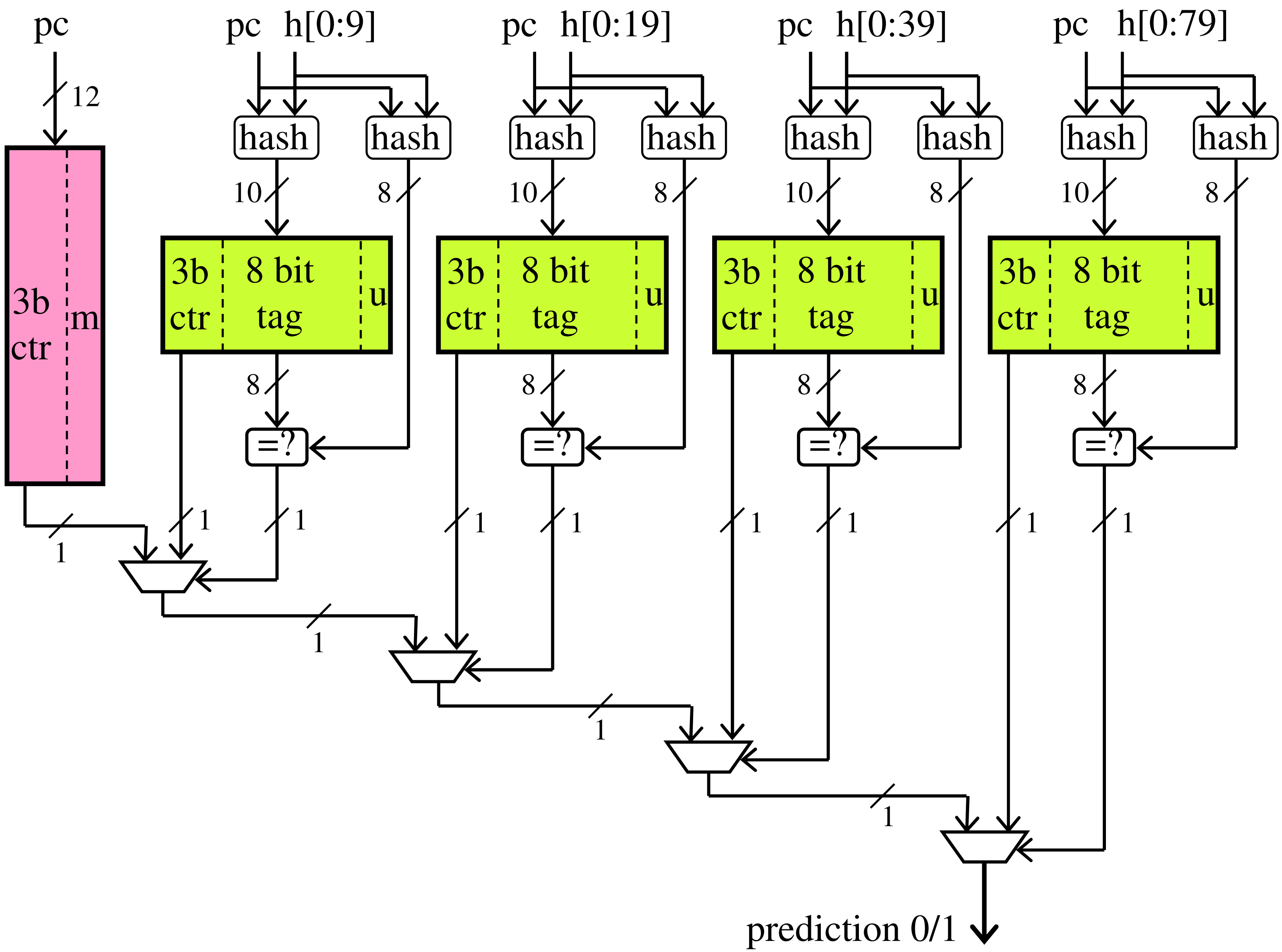
A PPM-like, tag-based predictor

Pierre Michaud

Main characteristics

- global history based
- 5 tables
  - one 4k-entry bimodal (indexed with PC)
  - four 1k-entry “global” (history length 10,20,40,80)
- “Global” tables are tagged (8-bit tags)
- Prediction given by the 3-bit up-down saturating counter associated with the longest matching history


pc
pc h[0:9]
pc h[0:19]
pc h[0:39]
pc h[0:79]
12
hash
10
8
3b ctr
m
3b ctr
8 bit tag
u
=?
1
prediction 0/1

References

- Perceptron predictor
  - Jiménez, Lin, HPCA 2001
  - ➔ benefit from a very long global history
- PPM (*prediction by partial matching*)
  - text compression: Cleary,Witten, IEEE Trans. on Communications, 1984
  - branch prediction "limit": Chen, Coffey, Mudge, ASPLOS 1996
  - ➔ spectrum of history lengths, prediction from longest matching history
  - ➔ permits using a very long global history with limited table space
- YAGS: bimodal table + 1 global table
  - Eden, Mudge, MICRO 1998
  - ➔ (short) tags do not waste table space
  - ➔ allocate entry in global table only if bimodal prediction is wrong

Predictor update

- X = longest matching history at prediction time
- Update 3-bit counter associated with X, and only that counter
  - Increment if taken, decrement otherwise
- If prediction was correct, we are done
- If prediction was wrong, try to steal entries for history lengths > X
  - Write the branch tag
  - Reinitialize 3-bit counter to a new value

New update method

- Bit *u* in each global table entry ➔ selective entry stealing
  - (*u* is for *useful entry*)
  - if we steal all entries > X, up to 4 entries stolen on each mispredict ➔ ☹
  - try to distinguish entries that we should avoid stealing
  - heuristic:
    - useful when prediction correct and bimodal wrong
    - not useful when prediction wrong and bimodal correct

- Bit *m* in each bimodal table entry ➔ 3-bit counter initialization
  - (*m* is for *meta-predictor*)
  - many entries deliver few predictions before being stolen
  - ➔ 3-bit counter initialization is important
  - if there is some correlation, better to initialize according to branch outcome
  - otherwise, better to initialize with bimodal prediction = most likely outcome

Precisely:

- If prediction was wrong and X < 80
  - Choose entries to steal
    - Read bit *u* for all entries > X
    - If at least one bit *u* is reset, steal only entries which bit *u* is reset
    - If all bits *u* are set, choose a random Y > X and steal only entry Y
  - Read bit *m* from bimodal
  - Steal entries
    - Write tag
    - Reset bit *u*
    - If *m* is set, initialize 3-bit counter according to branch outcome
    - Otherwise, initialize 3-bit counter according to bimodal prediction
- If prediction from X different from bimodal prediction
  - if X is correct, set both bit *m* in bimodal and bit *u* in entry X
  - Otherwise, reset both *m* and *u*

Why 3-bit counters ?

- Example: stream of random branch outcomes with 70% taken and 30% not-taken
  - predict *always taken* ➔ mispredict rate = 30%
  - 2-bit counter ➔ mispredict rate = 36 %
  - ➔ 20% higher

- In the proposed predictor, on the distributed traces, 3-bit counters are better than 2-bit counters.
  - Average: -3.3% mispredicts
  - Hard-to-predict traces: up to -6%

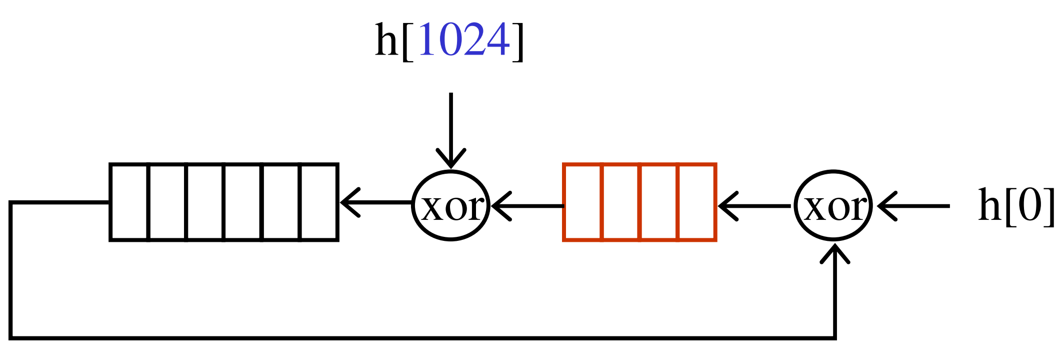
Hashing functions

Based on global history folding

Example: fold a 1024-bit history onto 10 bits

➔ use a cyclic shift register and a couple of XORs

1024 % 10 = four



More explanations…

- *Analysis of a tag-based branch predictor*, P. Michaud, IRISA research report PI-1660, Nov. 2004.
  - start from an ideal predictor, and introduce successive degradations corresponding to hardware constraints

- There is room for improvement
  - the problem bits *u* and *m* try to solve is not completely solved
  - in the ideal predictor, global table space is shared by all history lengths