ECE 4750
Computer Architecture

Prof. José F. Martínez

Dynamic Branch Prediction

- Control hazards limit performance
  - must figure out (1) it’s a branch, (2) target address, (3) outcome
  - gets worse when *(a)* issue ↑, *(b)* pipeline depth ↑
- Static schemes
  - delayed branches, predict (not) taken
  - do not capture dynamic behavior
- Dynamic schemes (today)
  - predictive hardware
  - action dependent on run-time behavior

Dynamic Branch Prediction

- Goal: prevent control dependences from causing stalls
- Means: predict branch outcome based on run-time behavior
- Effectiveness determined by
  - prediction accuracy
  - cost when prediction is correct
  - penalty when prediction is not correct
- Factors that affect effectiveness
  - pipeline structure
  - type of predictor
  - misprediction recovery scheme

Pattern History Table (PHT)

- Small memory indexed by lower portion of branch PC (why lower?)
    - similar to instruction cache, only every access is a hit
        - small, tagless, direct mapped
- Simplest: remember last outcome – one-bit PHT
    - targets highly biased branches
    - however, mispredictions come in pairs!

      ex.: loop branch taken 9/10 times; what is the misprediction rate?

Pattern History Table (PHT)

- Small memory indexed by lower portion of branch PC (why lower?)
  - similar to instruction cache, only every access is a hit
    - small, tagless, direct mapped
- Simplest: remember last outcome – one-bit PHT
  - targets highly biased branches
  - however, mispredictions come in pairs!

    ex.: loop branch taken 9/10 times; what is the misprediction rate?

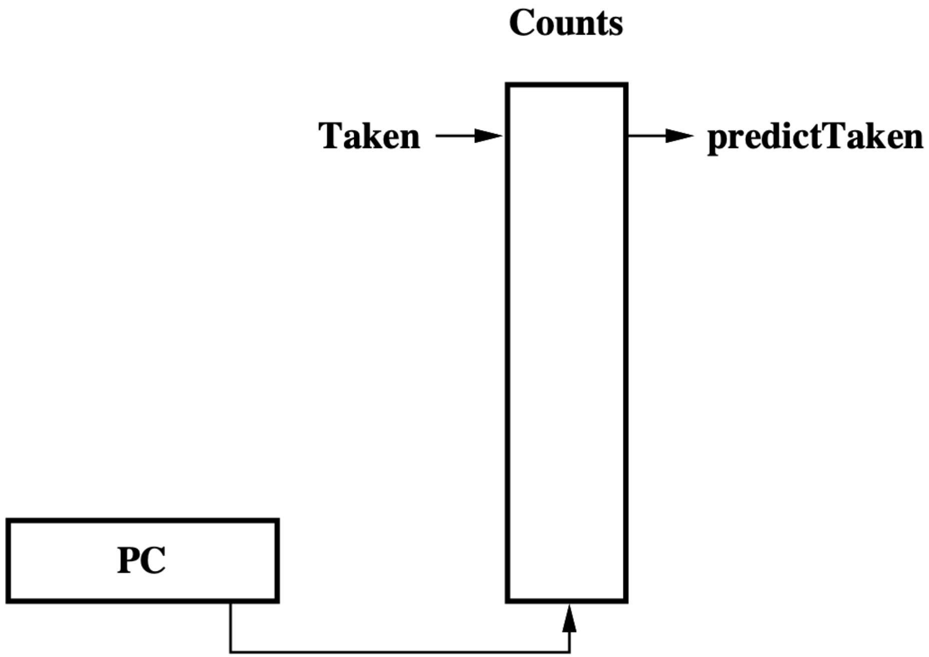
	Branch Instance										
	2	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
Prediction	T	T	T	T	T	T	T	T	T	NT	T
Outcome	T	T	T	T	T	T	T	T	NT	T	T

N-bit PHT

- Obvious solution: two-bit prediction (*saturating counter*)
  - taken increments (up to 11), not taken decrements (down to 00)
  - 0X predict not taken, 1X predict taken
- Generalization: n-bit prediction
  - 0XX…X predict not taken, 1XX…X predict taken
  - empirically, n=2 often good enough
- Complication vs. one-bit: requires more updates

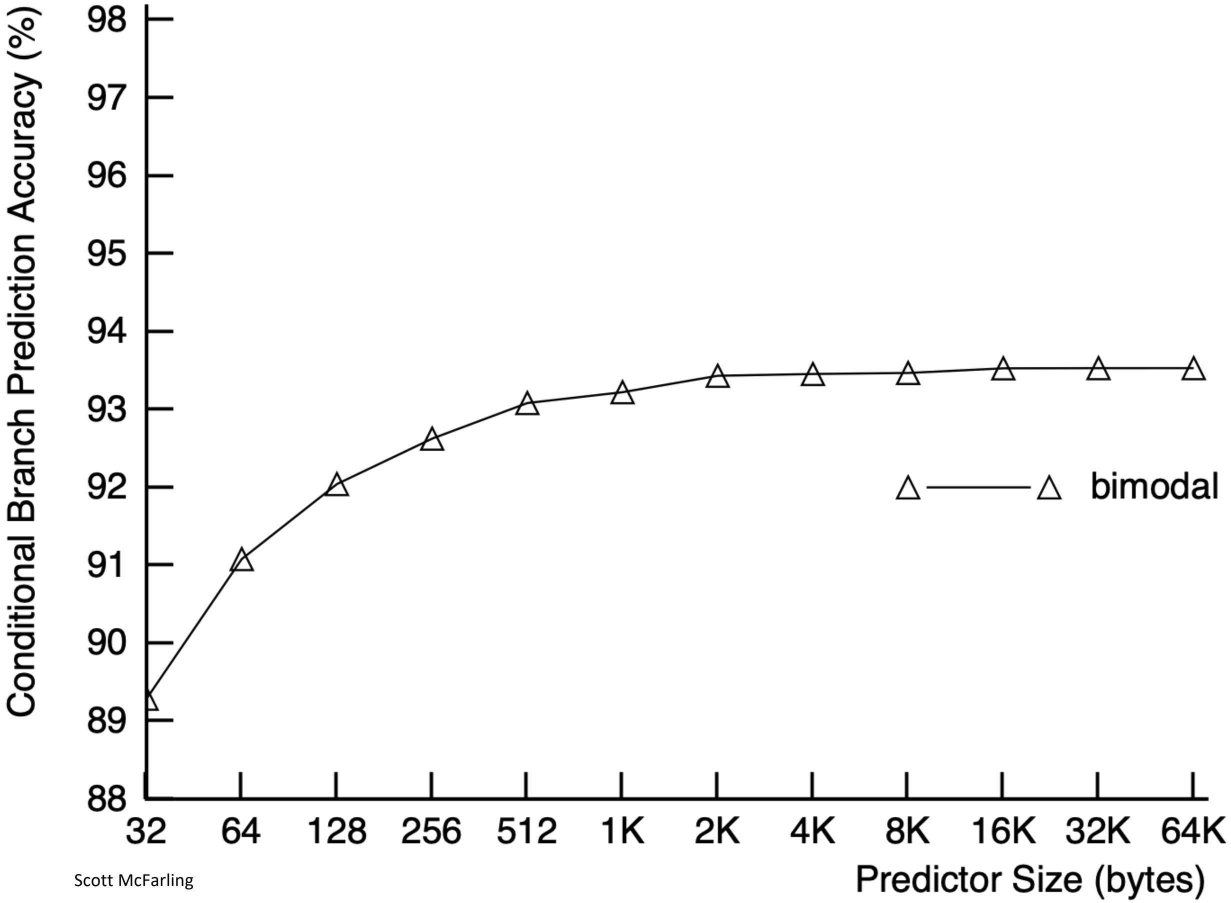
Bimodal

Smith, ISCA ’81



Scott McFarling

SPEC ’89 average

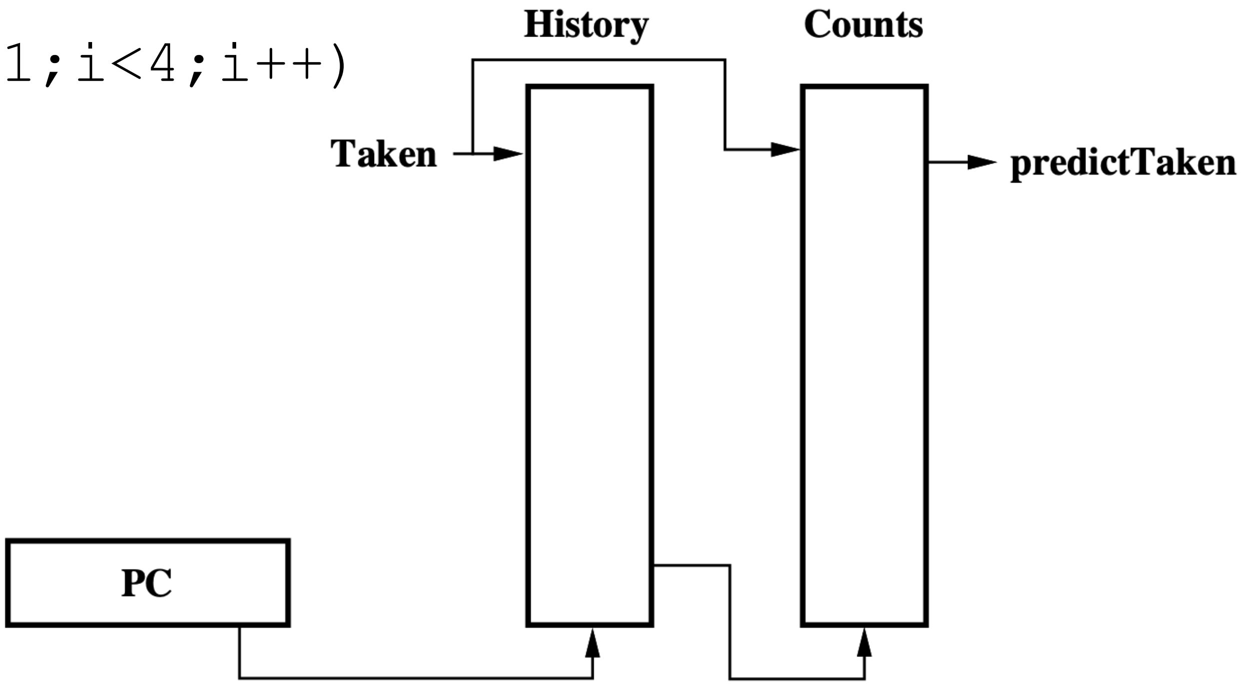


Scott McFarling

Local

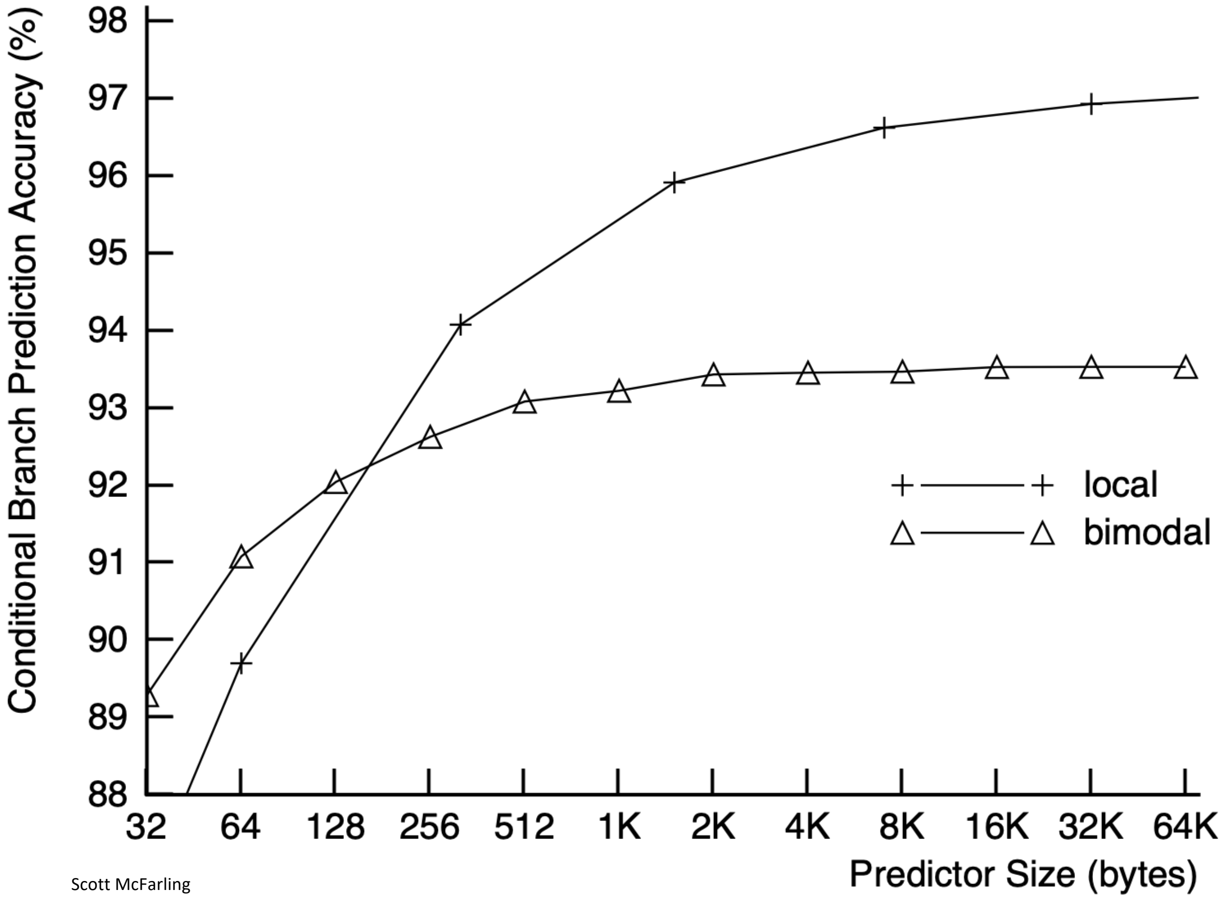
```
for (i=1;i<4;i++)
   …
```

Yeh and Patt, ISCA ’92



Scott McFarling

SPEC '89 average

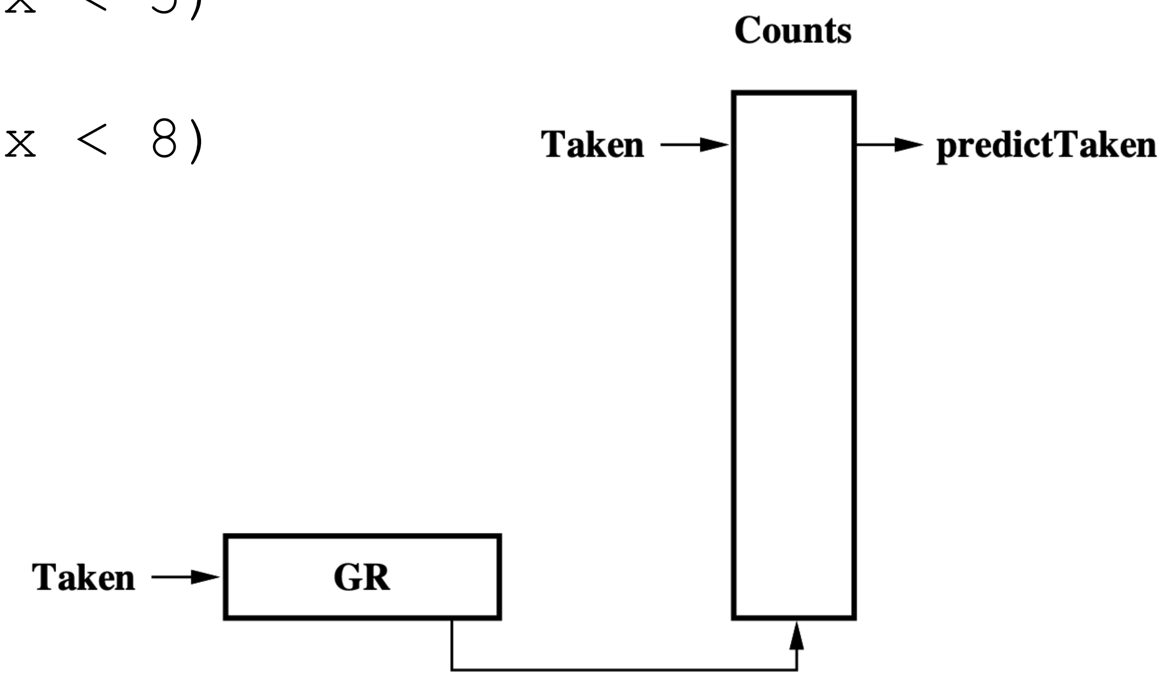


Scott McFarling

Global

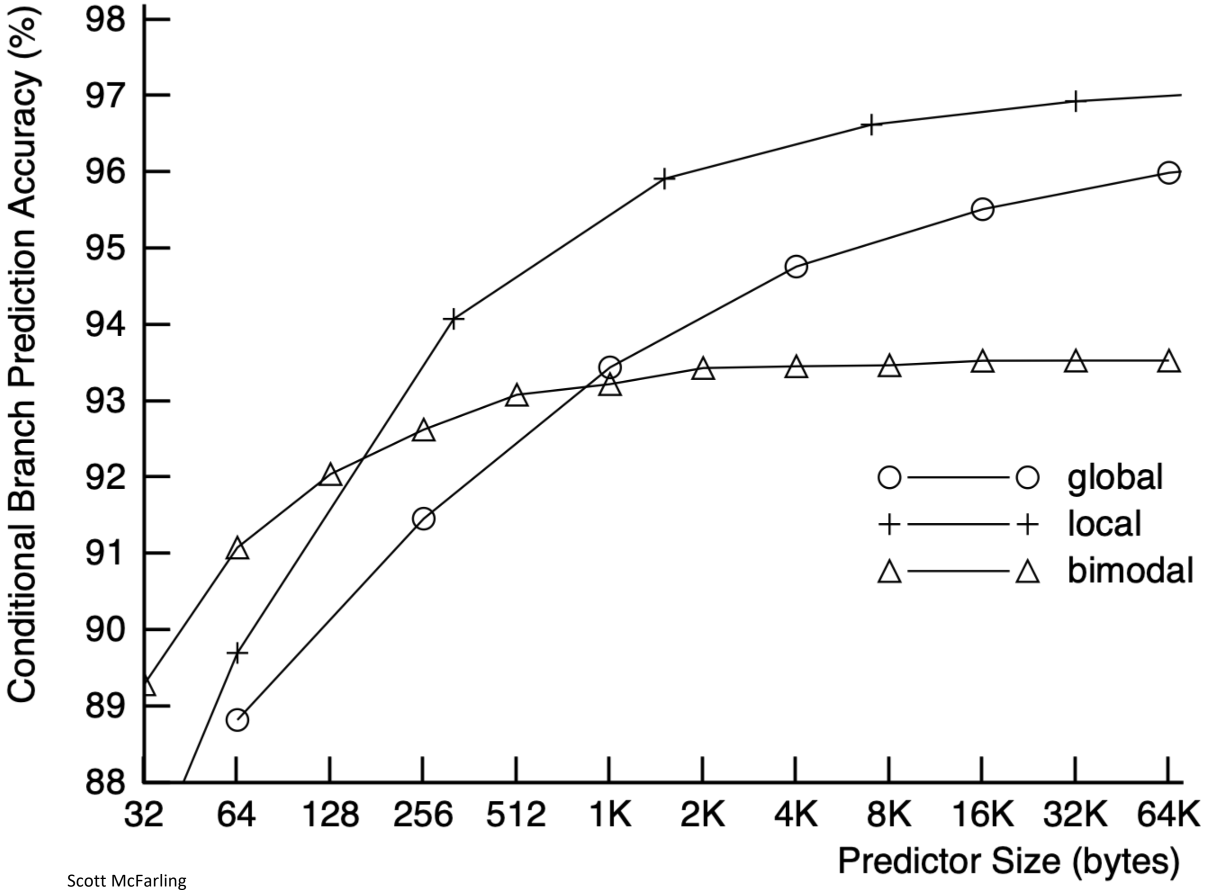
```
if (x < 5)
   …
if (x < 8)
   …
```

Yeh and Patt, ISCA ’92



Scott McFarling

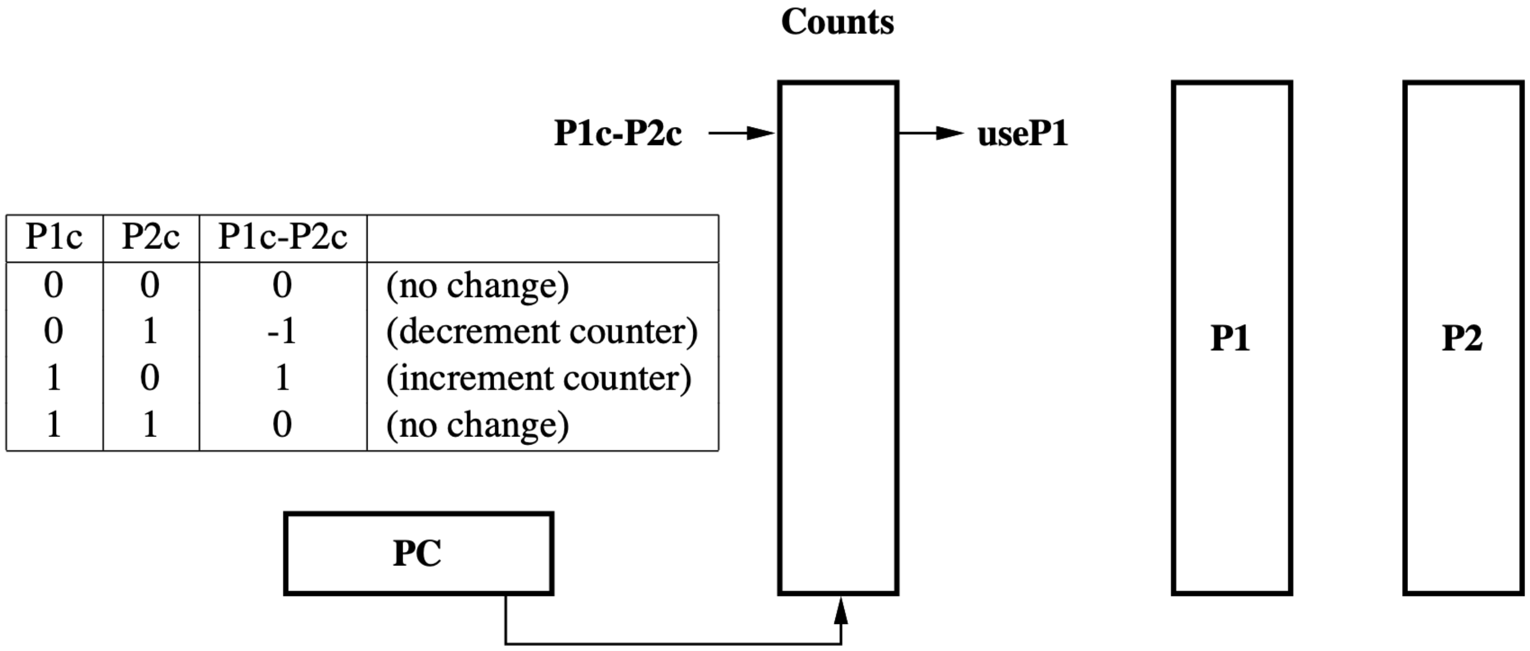
SPEC '89 average



Scott McFarling

Combined (aka Tournament or Selection)

McFarling, TR DEC-WRL TN-36



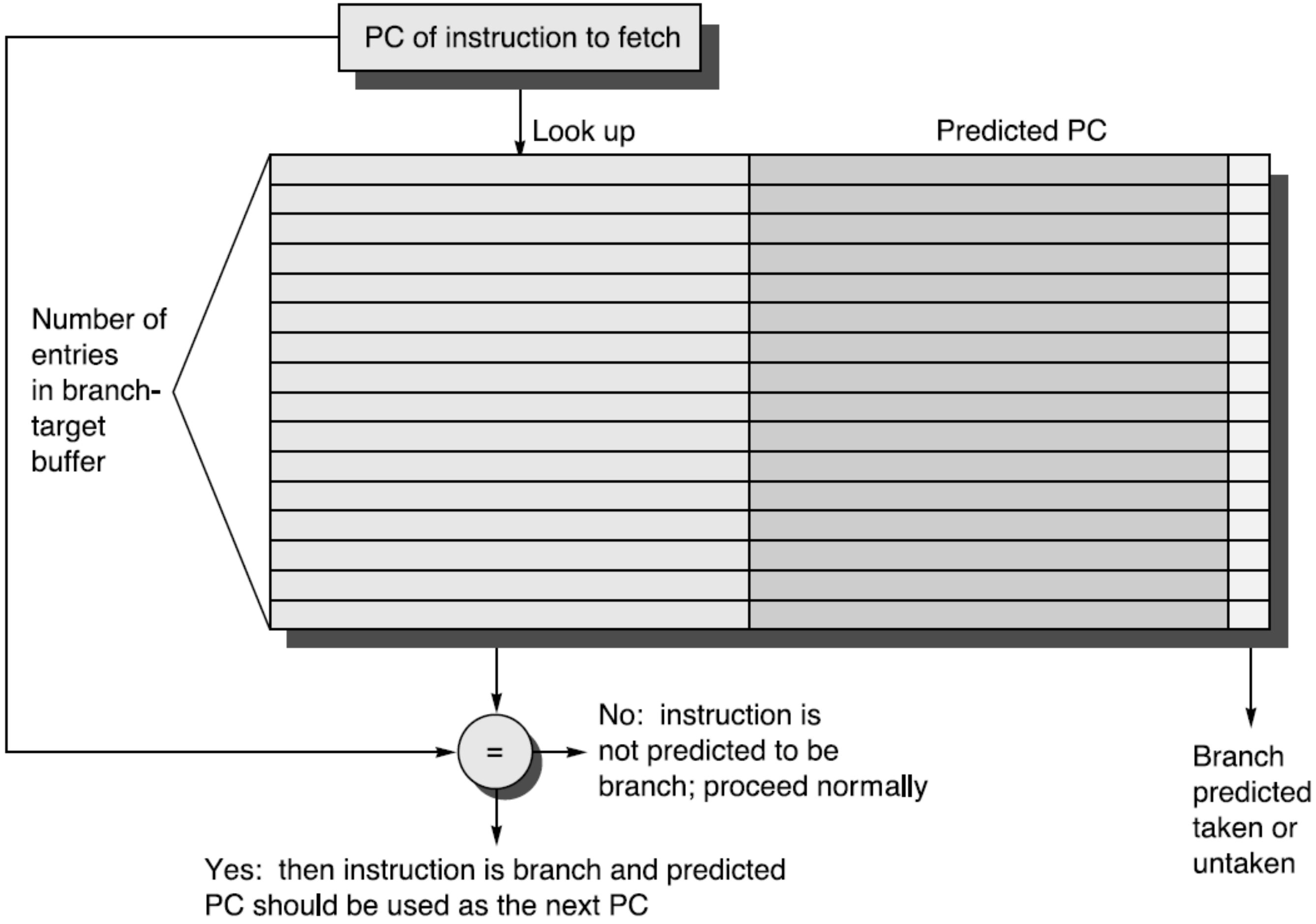
P1c	P2c	P1c-P2c	
0	0	0	(no change)
0	1	-1	(decrement counter)
1	0	1	(increment counter)
1	1	0	(no change)

Scott McFarling

Branch Target Buffer (BTB)

- PHT guesses direction; BTB tries to obtain target address
- If target determined by the end of IF, no penalty
  - PHT must be used at ID → one-cycle penalty
  - BTB can be used at IF → zero-cycle penalty
- Small cache indexed, tagged by PC
  - unlike PHT, should miss if not the right branch – or a branch at all!
- Allocate when branch and taken
- Problem: hit implies taken

BTB + PHT


PC of instruction to fetch
Look up
Predicted PC
Number of entries in branch-target buffer
=
No: instruction is not predicted to be branch; proceed normally
Yes: then instruction is branch and predicted PC should be used as the next PC
Branch predicted taken or untaken

Return Address Stack (RAS)

- Predict return address of a procedure
- Cannot use BTB – caller address may change every time
- Solution: use specialized stack
  - push PC+4 when `jal`
  - pop address, set PC upon returning
- Nested/recursive calls: remember last n return addresses