

BRANCH PREDICTORS

Mahdi Nazm Bojnordi

Assistant Professor

School of Computing

University of Utah

Branch Prediction

- **Goal:** avoiding stall cycles caused by branches
- **Solution:** static or dynamic branch predictor
 - ▣ 1. prediction
 - ▣ 2. validation and training
 - ▣ 3. recovery from misprediction
- **Performance** is influenced by the frequency of branches (**b**), prediction accuracy (**a**), and misprediction cost (**c**)

$$Speedup = \frac{Old\ Time}{New\ Time} = \frac{CPI_{old}}{CPI_{new}} = \frac{1 + bc}{1 + (1 - a)bc}$$

Problem

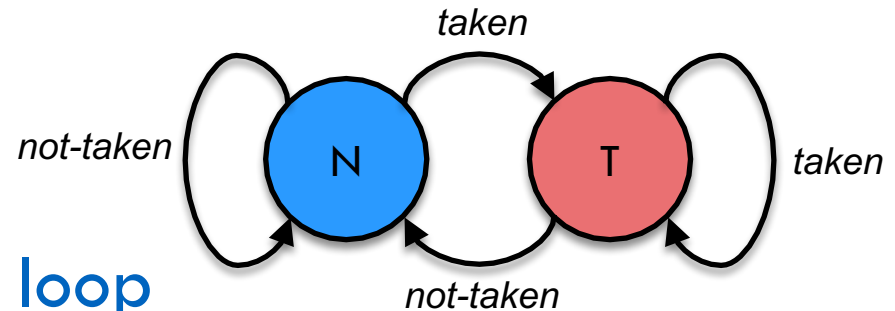
- A pipelined processor requires 3 stall cycles to compute the outcome of every branch before fetching next instruction; due to perfect forwarding/bypassing, no stall cycles are required for data/structural hazards; every 5th instruction is a branch.
- Compute speedup gained by a branch predictor with 90% accuracy

$$\text{Speedup} = (1 + 0.2 \times 3) / (1 + 0.1 \times 0.2 \times 3) = 1.5$$

Bimodal Branch Predictors

- One-bit branch predictor
 - ▣ Keep track of and use the outcome of last branch

- Shared predictor
- Two mispredictions per loop



$$\text{Accuracy} = 26/30 = 0.86$$

```
while(1) {  
    for(i=0; i<10; i++) {  
    }  
    for(j=0; j<20; j++) {  
    }  
}
```

branch-1

branch-2

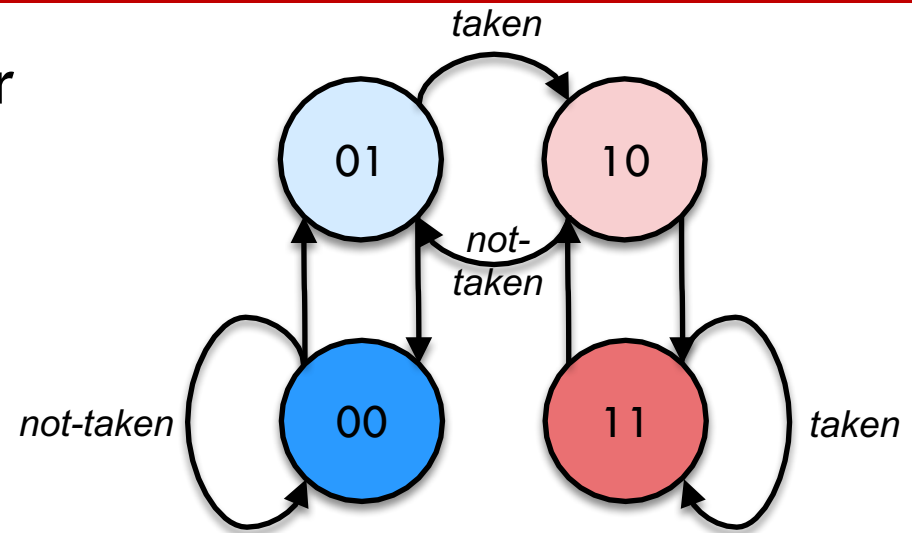
Bimodal Branch Predictors

□ Two-bit branch predictor

▣ Increment if taken

▣ Decrement if untaken

- A single predictor shared among many branches
- One misprediction on loop exit
- Accuracy = $28/30 = 0.93$
- **3-bit predictor?**



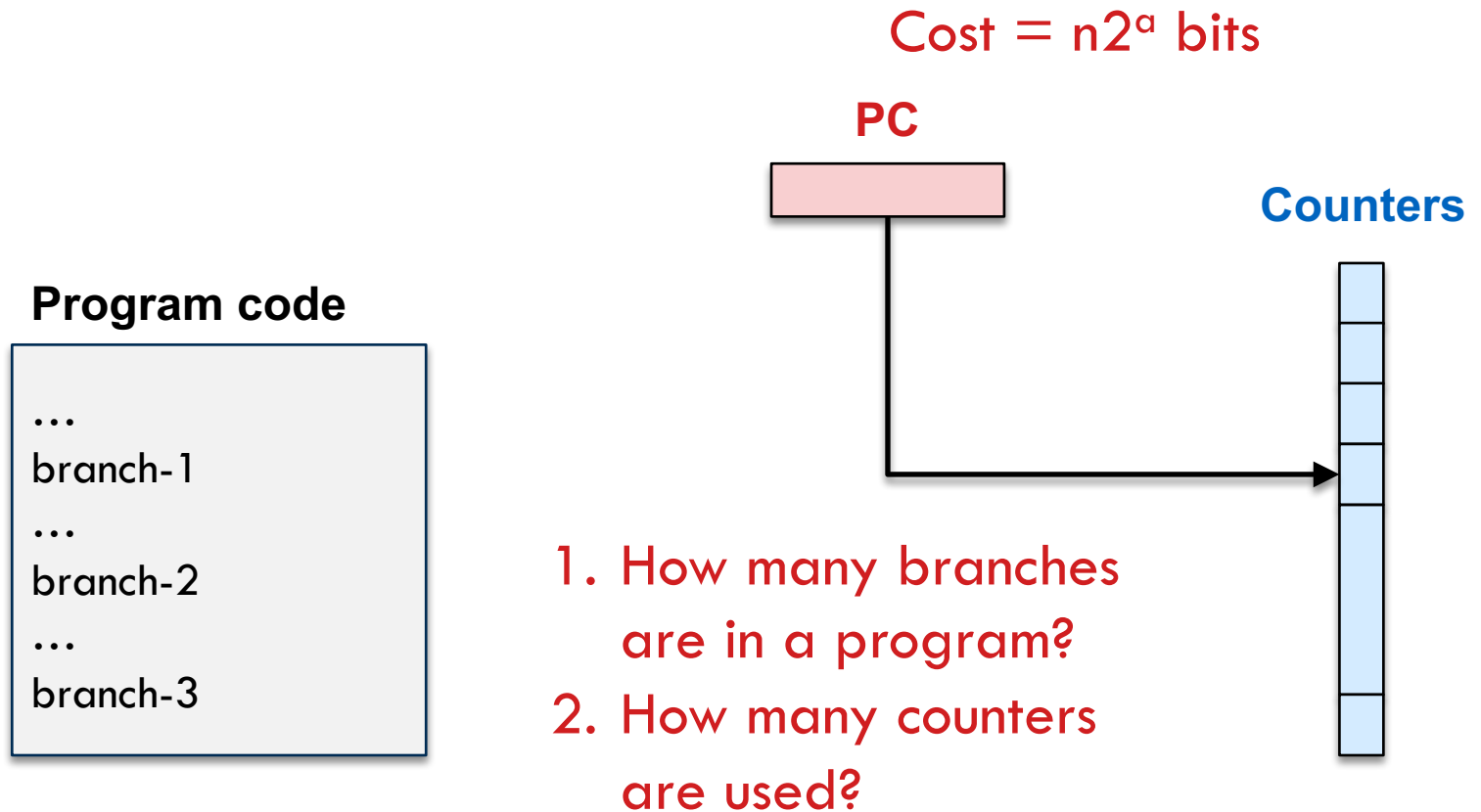
```
while(1) {
    for(i=0; i<10; i++) {
    }
    for(j=0; j<20; j++) {
    }
}
```

branch-1

branch-2

Using Multiple Counters

- How to assign a branch to each counter?



Using Multiple Counters

□ How to assign a branch to each counter?

▣ Decode History Table (DHT)

■ Reduced HW with aliasing

Program code

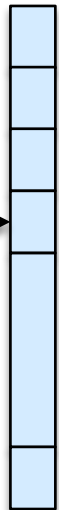
```
...  
branch-1  
...  
branch-2  
...  
branch-3
```

PC



Cost = $n2^b$ bits

Counters



Least significant bits are
used to select a counter

(+) Reduced hardware

(-) Branch aliasing

Using Multiple Counters

□ How to assign a branch to each counter?

▣ Decode History Table (DHT)

■ Reduced HW with aliasing

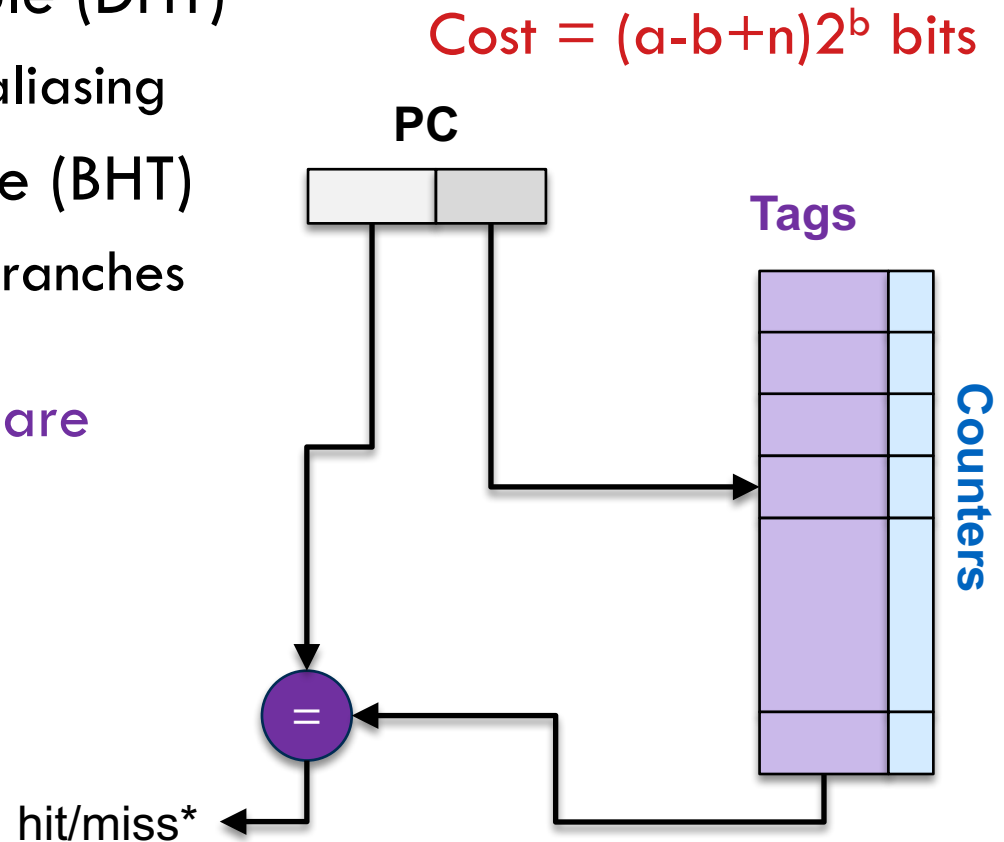
▣ Branch History Table (BHT)

■ Precisely tracking branches

Most significant bits are
used as tags

(+) No aliasing

(-) Missing entries



Using Multiple Counters

□ How to assign a branch to each counter?

▣ Decode History Table (DHT)

■ Reduced HW with aliasing

▣ Branch History Table (BHT)

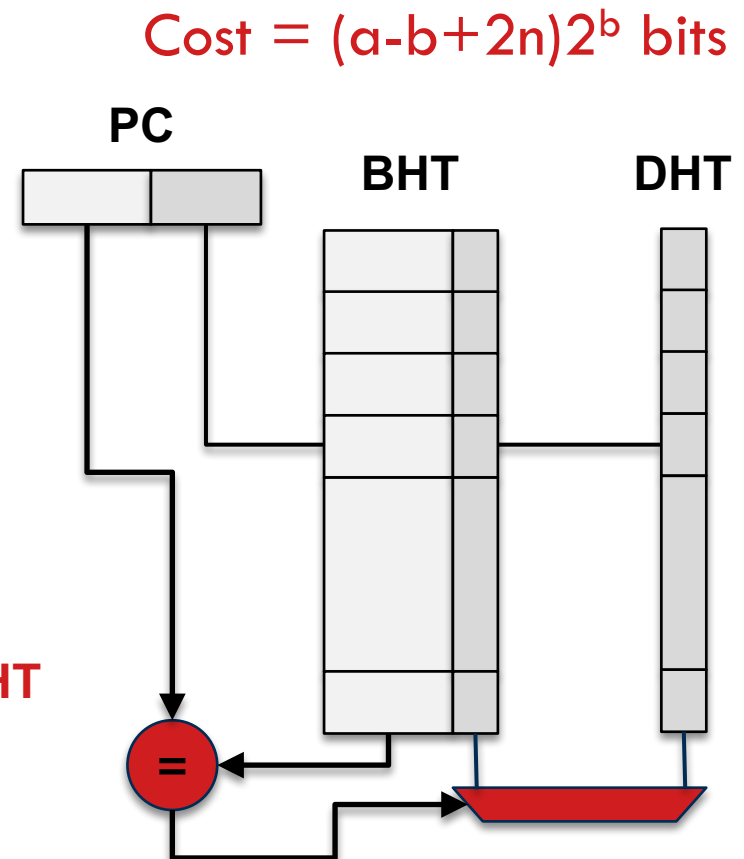
■ Precisely tracking branches

▣ Combined BHT and DHT

■ BHT is used on a hit

■ DHT is used/updated on a miss

DHT typically has more entries than BHT



Correlating Branch Predictor

- Executed branches of a program stream may be correlated

```
while (1) {  
    if(x == 0)  
        y = 0;  
    ...  
    if(y == 0)  
        x = 1;  
}
```

branch-1

branch-2

while:

```
BNEQ R1, R0, skip1  
ADDI R2, R0, #0
```

skip1: ...

```
BNEQ R2, R0, skip2  
ADDI R1, R0, #1
```

skip2: J while

Correlating Branch Predictor

- Executed branches of a program stream may be correlated

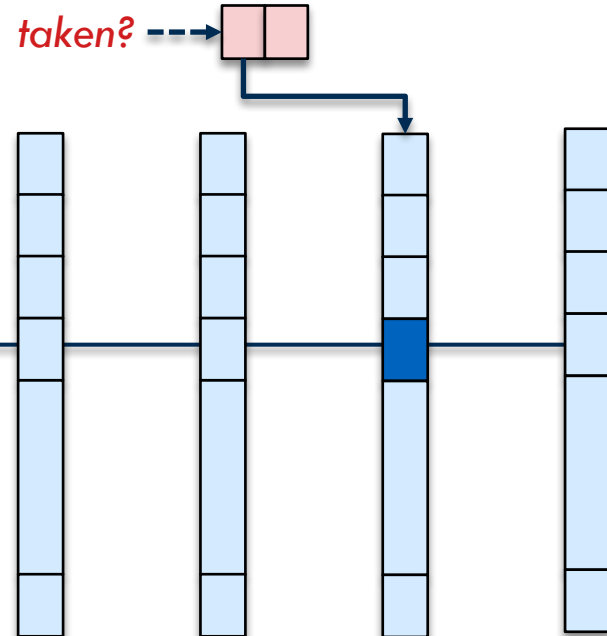
```
while (1) {  
    if(x == 0)  
        y = 0;  
    ...  
    if(y == 0)  
        x = 1;  
}
```

branch-1

branch-2

PC

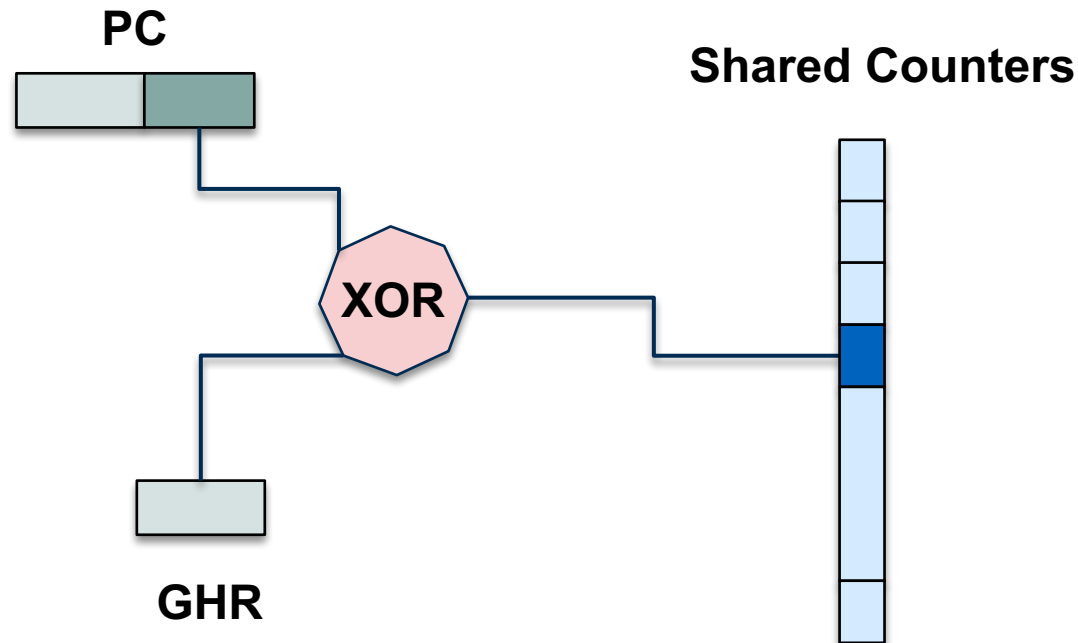
Global History Register: an r-bit shift register that maintains outcome history



Cost = $r + 2^{b+r}$ bits

Global Branch Predictor

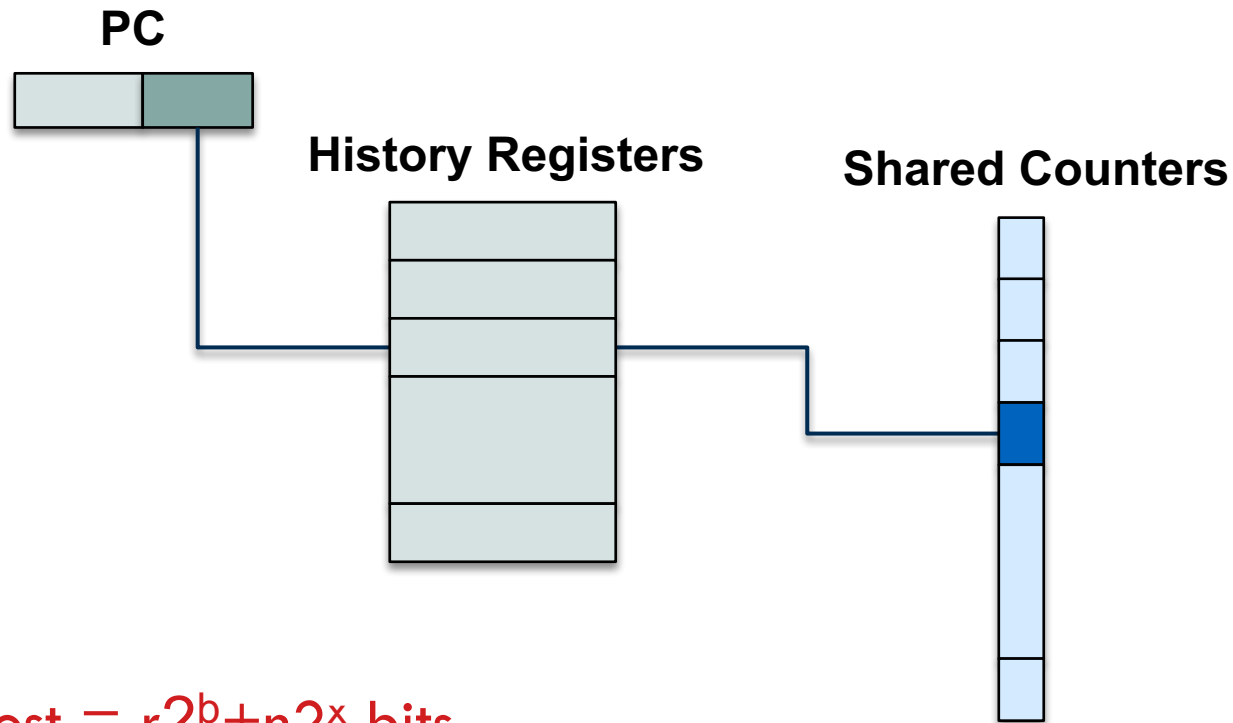
- GHR is merged with PC bits to choose a counter



Cost = $r + 2^r$ bits

Local Branch Predictor

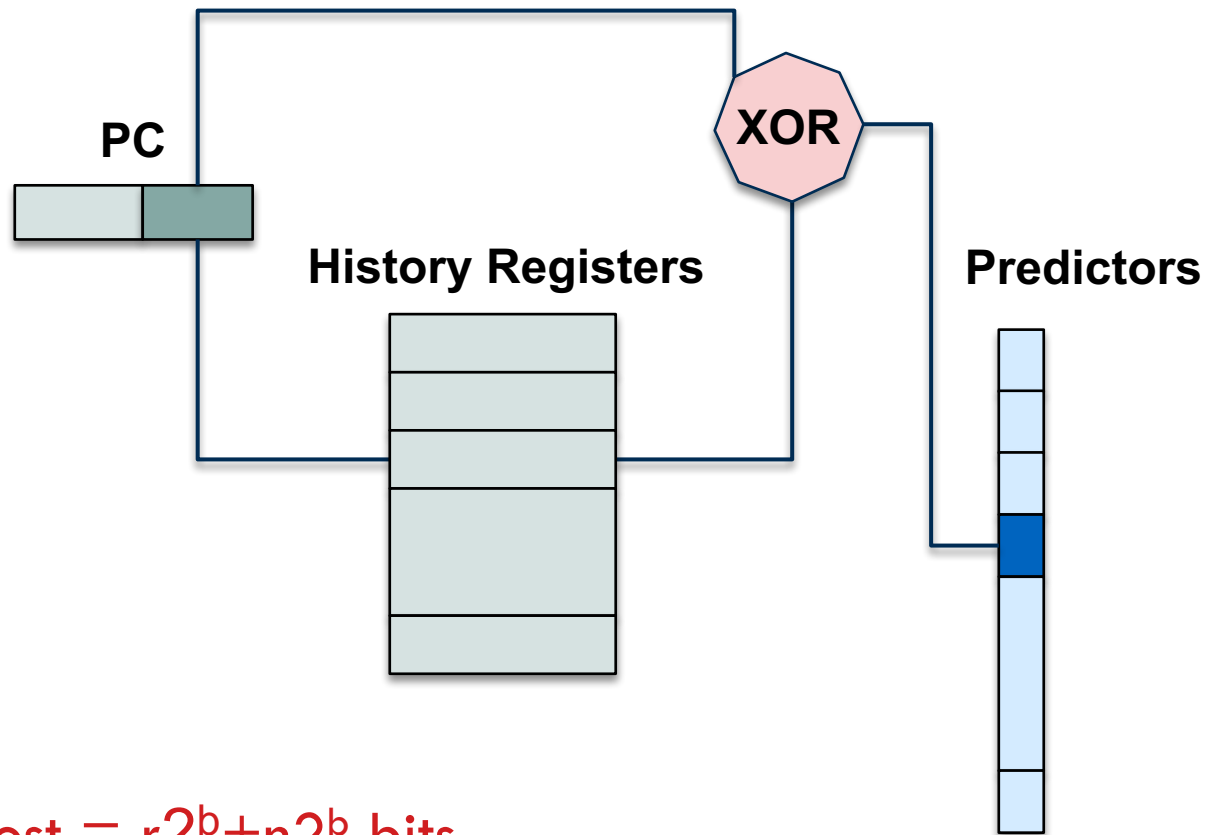
- One GHR per branch



$$\text{Cost} = r2^b + n2^x \text{ bits}$$

Local Branch Predictor

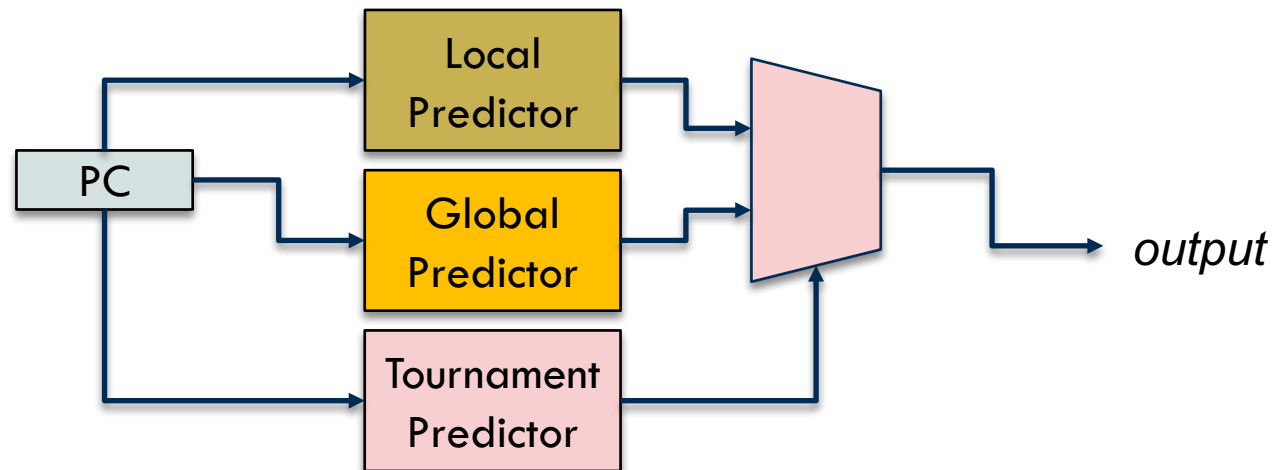
- One GHR per branch



$$\text{Cost} = r2^b + n2^b \text{ bits}$$

Tournament Branch Predictor

- Local predictor may work well for some applications, while global predictor works well for some other programs
- ▣ Include both and identify/use the best one for each branch



Two bit saturating counters

Branch Prediction Summary

- Dedicated predictor per branch
 - ▣ Program counter is used for assigning predictors to branches
- Capturing correlation among branches
 - ▣ Shift register is used to track history
- Predicting branch direction is not enough
 - ▣ Which instruction to be fetched if taken?
- Storing the target instruction can eliminate fetching
 - ▣ Extra hardware is required

Branch Target Buffer

- Store tags and target addresses for each branch

