# CSE 240A Branch Predictor Project

**Due : May 16 2022, 11:00 PM**

**Chi-Hsin Lo, PID : A53311981**

1. Branch Predictor Name : Perceptron Branch Predictor
   I implemented the Perceptron in paper "Dynamic Branch Prediction with Perceptrons" (https://ieeexplore.ieee.org/document/903263). The code is on my github repository (https://github.com/chhiilnos1996/CSE240-BranchPredictor).

2. Branch Predictor Description/Implementation

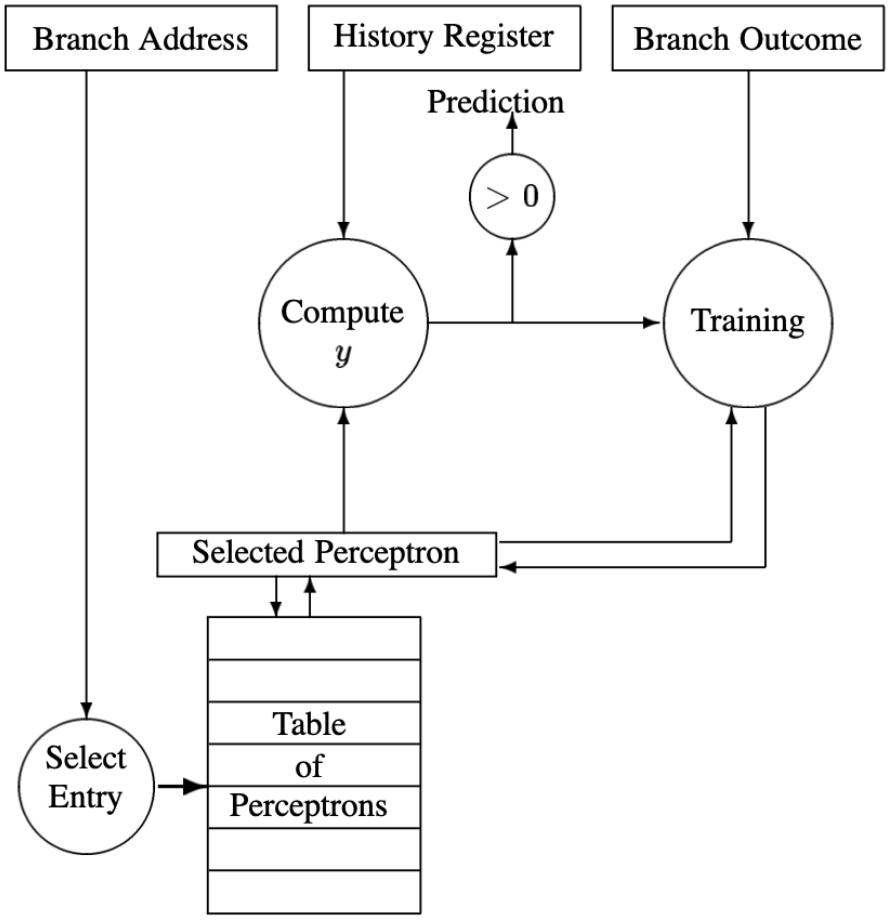


Figure 1: The Perceptron Branch Predictor

(a) Design Space

I used $h = 28$ bits for global history length. TOP (table of perceptrons) contains $s = 112$ perceptron models, each has $h + 1 = 29$ weights and $b + 1 = 10$ bits to save each signed weight. Hash function is simply the pc module the size of the TOP.

(b) Initialization

Initialize the weight to $(floor(1.93 \times h) + 14)/4$, this is different from the paper by a factor of 4 because I discovered that this threshold is adequate. And with smaller threshold, fewer bits is required to store the weight and thus I can increase my TOP size for less aliasing.Each weight of TOP $(112 * (b+1) * (h+1)$ of them) is initialized to weakly not taken (WN). Global history register is initialized to zero.

(b) Prediction

The prediction scheme is identical to what written in the paper. PC is fed into the hash function to obtain the index of TOP, and then the weights is obtained by $TOP[index]$, the weight is then multiplied with $(2 \times x_i - 1)$ and summed as the output. The prediction is taken if $y_{out} > 0$, not taken otherwise.

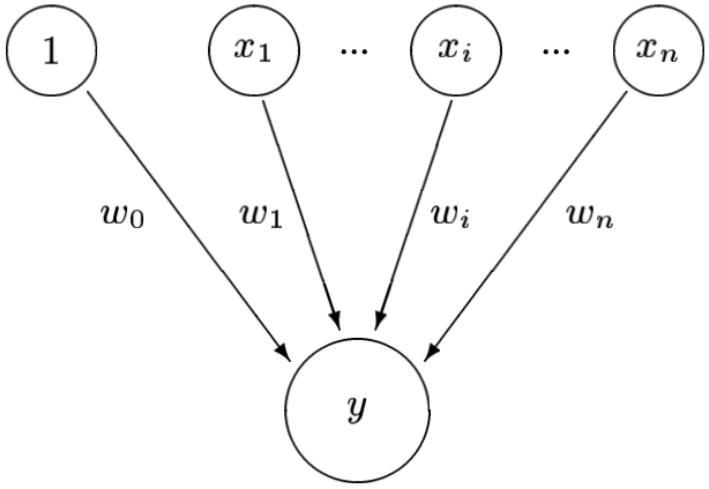


Figure 2: Perceptron Branch Predictor Prediction Scheme

(c) Training

The update policy is identical to what written in the paper. If the outcome and prediction disagrees or the absolute value of $y_out$ is smaller than threshold, then the weight is updated. The weight is updated by adding 1 if the outcome agrees with $x_i$, updated by decreasing 1 if the outcome disagrees with $x_i$. The weight is updated only if the weight doesn't exceed the range that can be expressed by the number of bits. The global history register is updated by adding the newest outcome to the leas significant bit.

3. Size Analysis

(a) Tournament

Assume we use $k$ bits of pc, $m$ bits for local branch pattern, $l$ bits for global history. Then the

local PHT size is $2^k \times m$, local BHT size is $2^m \times 2$, global BHT size is $2^l \times 2$, and choice predictor size is $2^l \times 2$. This should add up to no more than $32K = 2^{15}$ bits. I wrote a Python code to generate the (k,m,l) tuples and chose (11, 10, 12) for my tournament implementation.

```
for l in range(10, 14):
    for k in range(5, 14):
        for m in range(5, 14):
            result = pow(2,l+2)+pow(2,m+1)+m*pow(2,k)
            if result<=pow(2,15):
              print(m,k,l)
```

Figure 3: Python code to generate design parameter for Tournament predictor

(b) Perceptron Assume we use $b+1$ bits for each weight , $h$ bits for global history register, $s$ is the size of the TOP(table of perceptrons). Then the TOP size is $s \times (h+1) \times (b+1)$, this should be no more than $32K = 2^{15}$ bits. I wrote a Python code to compute the maximum $s$ value can be given $b, h$ in a range and printed out (b,h,s) tuples. I chose (9, 28, 112) for my perceptron implementation.

```
import math

for b in range(9, 14):
    for h in range(12,34):
        result = (b+1)*(h+1);
        s = math.floor(pow(2,15)/result);
        print("# ", b,h,s, "->")
```

Figure 4: Python code to generate design parameter for Perceptron predictor

4. Performance Analysis

(a) Tournament : Gradescope score 2.072833

| trace | $fp_1$ | $fp_2$ | $int_1$ | $int_2$ | $mm_1$ | $mm_2$ |
|---|---|---|---|---|---|---|
| Branches | 1546797 | 2422049 | 3771697 | 3755315 | 3014850 | 2563897 |
| Incorrect | 15131 | 48933 | 386084 | 13487 | 47908 | 182847 |
| Misprediction Rate | 0.978 | 2.020 | 10.236 | 0.359 | 1.589 | 7.132 |

(b) Perceptron : Gradescope score 2.29866

| trace | $fp_1$ | $fp_2$ | $int_1$ | $int_2$ | $mm_1$ | $mm_2$ |
|---|---|---|---|---|---|---|
| Branches | 1546797 | 2422049 | 3771697 | 3755315 | 3014850 | 2563897 |
| Incorrect | 12668 | 34100 | 374004 | 12767 | 72703 | 217276 |
| Misprediction Rate | 0.819 | 1.408 | 9.916 | 0.340 | 2.411 | 8.474 |

5. Design Space
   (a) Tournament
   A set of (b, h) combinations is exploited and the result (Gradescope score) is listed below.

| (h, b-1) | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|
| 12 | 2.62 | 2.65 | 2.68 | 2.65 | - |
| 14 | 2.59 | - | - | - | - |
| 16 | 2.47 | 2.57 | 2.52 | 2.52 | - |
| 18 | 2.53 | - | - | - | - |
| 20 | 2.38 | 2.38 | 2.37 | 2.36 | 2.53 |
| 22 | 2.37 | - | - | - | - |
| 24 | 2.40 | 2.49 | 2.39 | 2.42 | 2.51 |
| 25 | 2.40 | - | - | - | - |
| 26 | 2.29 | - | - | - | - |
| 27 | 2.35 | - | - | - | - |
| 28 | 2.29 | 2.36 | 2.32 | 2.45 | 2.36 |
| 29 | 2.32 | - | - | - | - |
| 30 | 2.37 | 14.7 | - | - | - |
| 31 | 2.34 | - | - | - | - |
| 32 | 14.34 | 14.7 | 15.15 | 14.97 | - |
| 33 | 10.21 | - | - | - | - |

   (b) Perceptron
   A set of (k, m, l) combinations is exploited and the result (Gradescope score) is listed below.

| (k,m,l) | score |
|---|---|
| (13,9,11) | 2.07 |
| (11,11,10) | 2.23 |
| (12,10,11) | 2.07 |
| (10,11,11) | 2.41 |
| (11,10,12) | 2.07 |
| (12,9,12) | 2.21 |
| (10,10,12) | 2.34 |