Lab 2 Write Up

Saturday, November 16, 2024 11:18 PM

Sanjay Krishna

306258619

- So for my design I used a TAGE implementation with majority function to decide my branch predictor

Implementation Specs: Methodology

-

```
// Keep the same history lengths matching table sizes
#define HISTORY_LENGTH 22
#define HISTORY2_LENGTH 20
#define HISTORY3_LENGTH 18
#define HISTORY4_LENGTH 16
#define HISTORY5_LENGTH 14
#define HISTORY6_LENGTH 12
#define HISTORY7_LENGTH 10
#define HISTORY8_LENGTH 8
#define HISTORY9_LENGTH 6
#define HISTORY10_LENGTH 4
#define HISTORY11_LENGTH 2  // New history length for the eleventh table

#define TABLE_BITS 22
#define TABLE2_BITS 20
#define TABLE3_BITS 18
#define TABLE4_BITS 16
#define TABLE5_BITS 14
#define TABLE6_BITS 12
#define TABLE7_BITS 10
#define TABLE8_BITS 8
#define TABLE9_BITS 6
#define TABLE10_BITS 4
#define TABLE11_BITS 2  // New table bits for the eleventh table
```

Varying History Lengths

Varying Table Bits

My design has 11 Tables, to give us more variance in our searches

- All of my Tables are also initialized to 2 which is ID to signal weak true branch

- Weighted Tables, to take into account history lengths

```
// Adaptive weights based on counter confidence
int weight1 = (cnt1 == 0 || cnt1 == 3) ? 7 : 5;
int weight2 = (cnt2 == 0 || cnt2 == 3) ? 6 : 4;
int weight3 = (cnt3 == 0 || cnt3 == 3) ? 5 : 3;
int weight4 = (cnt4 == 0 || cnt4 == 3) ? 4 : 2;
int weight5 = (cnt5 == 0 || cnt5 == 3) ? 3 : 1;
int weight6 = (cnt6 == 0 || cnt6 == 3) ? 3 : 1;
int weight7 = (cnt7 == 0 || cnt7 == 3) ? 2 : 1;
int weight8 = (cnt8 == 0 || cnt8 == 3) ? 2 : 1;
int weight9 = (cnt9 == 0 || cnt9 == 3) ? 1 : 1;
int weight10 = (cnt10 == 0 || cnt10 == 3) ? 1 : 1;
int weight11 = (cnt11 == 0 || cnt11 == 3) ? 1 : 1;  // New weight for the eleventh table

// Enhanced voting system
int weighted_vote = 0;
```

Longer histories have more weight

- Those 2 above basically summarize my TAGE Majority Predictor implementation roughly

- To optimize change history sizes has the most important along with table sizes accordingly

Trade offs:

- My main trade off is with my memory storage and history storage capabilities
- For example as stated above, my table sizes vary on large sizes ↙

```
#define TABLE_BITS 22
#define TABLE2_BITS 20
#define TABLE3_BITS 18
#define TABLE4_BITS 16
#define TABLE5_BITS 14
#define TABLE6_BITS 12
#define TABLE7_BITS 10
#define TABLE8_BITS 8
#define TABLE9_BITS 6
#define TABLE10_BITS 4
#define TABLE11_BITS 2  // New table bits for the eleventh table
```

Storage Overhead:

- Calculate each Table

$$2^{22} + 2^{20} + 2^{18} + 2^{16} + 2^{14} + 2^{12} + 2^{10} + 2^{8} + 2^{6} + 2^{4} + 2^{2}$$

$$\approx 2^{22} + 2^{20} + 2^{18}$$

Negligible size