ECEN 5593 – Advanced Computer Architecture

**Adarsh Kulkarni**

**Checkpoint-1A – Branch Prediction**

1. Determine the prediction accuracy of each predictor for FOUR of the program applications using PIN's ability to run : 400.perlbench 401.bzip2 403.gcc 429.mcf 445.gobmk 456.hmmer 458.sjeng 462.libquantum 464.h264ref 471.omnetpp 473.astar

**1 Bit Results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **401.bzip2** | **403.gcc** | **464.h264** | **429.mcf** |
| **Total Branches** | 10000000000 | 77185887 | 10000000000 | 4674974049 |
| **Count Replaced** | 9391469 | 5258211 | 108908363 | 3725262 |
| **Count taken** | 86479712250 | 50325302 | 8639730661 | 3425228829 |
| **Count Correct** | 9204380950 | 65184356 | 8934992135 | 4232041017 |
| **Accuracy** | 0.920438 | 0.844511 | 0.893499 | 0.905254 |

**2 bit Results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **401.bzip2** | **403.gcc** | **464.h264** | **429.mcf** |
| **Total Branches** | 10000000000 | 77196828 | 10000000000 | 4674974045 |
| **Count Replaced** | 9391469 | 5256511 | 108908496 | 3725261 |
| **Count taken** | 6479712246 | 50331164 | 8639730876 | 3425228824 |
| **Count Correct** | 9374530661 | 66542076 | 9336698622 | 4340842591 |
| **Accuracy** | 0.937453 | 0.861979 | 0.93367 | 0.928528 |

**8bit GAg Results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **401.bzip2** | **403.gcc** | **464.h264** | **429.mcf** |
| **Total Branches** | 10000000000 | 77189077 | 10000000000 | 4674974040 |
| **Count Replaced** | 9391470 | 5255965 | 108908553 | 3725260 |
| **Count taken** | 6479712244 | 50325721 | 8639730291 | 3425228823 |
| **Count Correct** | 9189439493 | 62973495 | 9127739242 | 4372736063 |
| **Accuracy** | 0.918944 | 0.815846 | 0.933483 | 0.93535 |

**12 bit PAg Results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **401.bzip2** | **403.gcc** | **464.h264** | **429.mcf** |
| **Total Branches** | 10000000000 | 77165976 | 10000000000 | 4674974053 |
| **Count Replaced** | 9391470 | 5254663 | 108908488 | 3725260 |
| **Count taken** | 6479712249 | 50311309 | 8639730571 | 3425228828 |
| **Count Correct** | 9416926522 | 68136405 | 9673543003 | 4413568039 |
| **Accuracy** | 0.941693 | 0.882985 | 0.967354 | 0.944084 |

**2. Develop a new unique predictor, explain the predictor, and report and the predictor's accuracy. There could be a number of new branch predictor schemes. You have to create on your own idea, you do not have to do one of the PAg, SAg, SAs, etc, types. You can invent your own (hybrid, etc).**

I have implemented a hybrid predictor consisting of a Gshare predictor and a 12 bit Pag predictor. The Gshare predictor indexes into the pattern history table by XOR'ing the branch history register and the program counter. This helps to prevent the aliasing problem encountered in the Gag predictor. The selection of the prediction mechanism is made by another table of 2 bit counters. This table is updated when the prediction given by the Gshare and Pag is not the same. If the value from the table is > 1, then the Pag predictor is chosen or the Gshare is chosen if < 1.

Hybrid predictor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **401.bzip2** | **403.gcc** | **464.h264** | **429.mcf** |
| **Total Branches** | 5288013611 | 77182152 | 10000000000 | 4674974053 |
| **Count Replaced** | 9391470 | 5254299 | 108908552 | 3725260 |
| **Count taken** | 6479712249 | 50326371 | 8639730496 | 3425228828 |
| **Count Correct** | 9416918732 | 69617345 | 9759795675 | 4416081648 |
| **Accuracy** | 0.941692 | 0.901908 | 0.97598 | 0.944029 |

**3- Answer the following by performing experiments.**

**What is the prediction accuracy of each scheme, for each application?**

The prediction accuracy of each scheme is as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **401.bzip2** | **403.gcc** | **464.h264** | **429.mcf** |
| **1 bit** | 92.0438 | 84.4511 | 89.3499 | 90.5254 |
| **2 bit** | 93.7453 | 86.1979 | 93.367 | 92.8528 |
| **8 bit GAg** | 91.8944 | 81.5846 | 93.3483 | 93.535 |
| **12 bit PAg** | 94.1693 | 88.2985 | 96.7354 | 94.4084 |
| **Hybrid** | 94.1692 | 90.1908 | 97.598 | 94.4029 |

**What is the miss rate (not finding a branch in the BTB) for each application?**

The miss rate (not finding a branch in the branch target buffer) for each application is as found out by dividing the BTB misses by the total count of the instructions seen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Miss rate for 401.bzip2** | **Miss rate for 403.gcc** | **Miss rate for 464.h264** | **Miss rate for 429.mcf** |
| **1 bit** | 5.05 % | 27.2 % | 6.8 % | 1.5 % |
| **2 bit** | 5.051% | 27.1 % | 6.7% | 1.48 % |
| **8 bit GAg** | 5.056 % | 27.04 % | 6.7% | 1.46 % |
| **12 bit PAg** | 5.09 % | 27.05 % | 6.67% | 1.41 % |
| **Hybrid** | 5.08 % | 27.15 % | 6.61% | 1.4% |

**Does using a longer global history for the GAg scheme, provide better prediction accuracy for the different application benchmarks?**

Using a longer global history does provide better prediction as evidenced from the table below. A longer history helps in reducing the aliasing issue, since a longer history register will also have a larger pattern history table.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **8 bit GAg** | **10 bit GAg** | **12 bit GAg** |
| **401.bzip2** | 0.918944 | 0.927641 | 0.933976 |
| **403.gcc** | 0.815846 | 0.830596 | 0.880504 |
| **464.h264** | 0.933483 | 0.926932 | 0.941174 |
| **429.mcf** | 0.93535 | 0.944719 | 0.95028 |

**Which applications have the most global correlation and why?**

403.gcc, 401.bzip2 and 429.mcf have the most global correlation since increasing the history register size increases the accuracy in an almost linear manner.