ECE 565 Project Report

Comparision of Performance of Prophet -Critic and Piecewise-Linear Branch Prediction Mechanisms in an Out-of-Order Processor

|  |  |
| --- | --- |
| Linfeng Zhang: Prophet/critic hybrid | Goutham Anand: Piecewise-Linear |

Abstract

In this report, we are going to discuss the implementation of two branch predictor, the performance and comparison of what scenario that each implementation is good at.

Introduction

Motivation:

Branch prediction is a good technique to minimize the negative impact of handle branch condition. It will have several impacts in performance, area and power. Modern branch predictor can achieve more than 90 percent accuracy which boost performance. To figure out which branch prediction algorithm will be important to improve the flow in instruction pipeline.

Description:

In this project we will implement two branch prediction algorithms to figure out which algorithm is better in performance. There are two papers describing Prophet/Critic and piece-wise linear branch predictors. We will implement these two algorithms in GEM5.

Survey of previous work

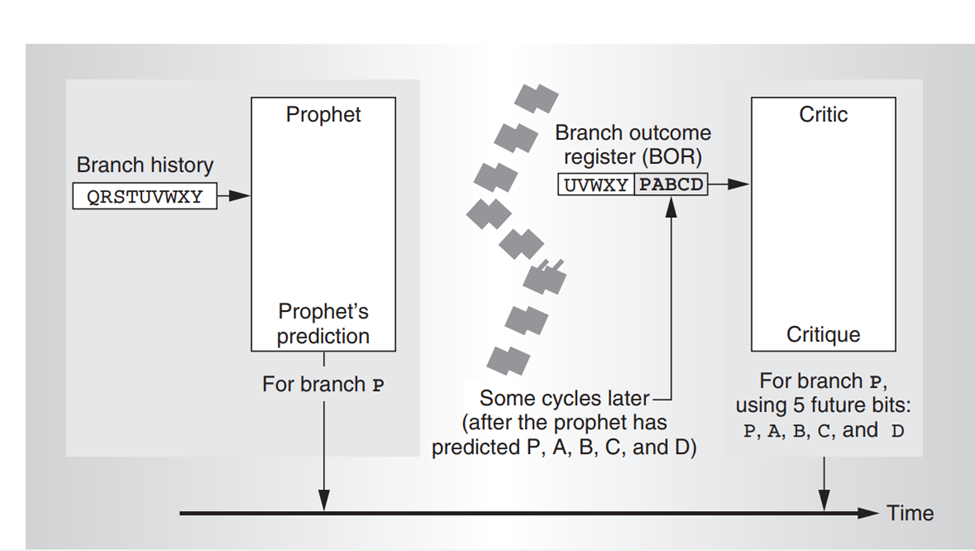


Figure 1 structure of prophet/critic hybrid branch predictor

Prophet predicts result based on the branch history, critic predicts result based on several bits generated by prophet, which is called future bits. These two branch predictors can be any of the traditional branch predictors. In addition they are running in serial rather than running parallel and need an arbiter to pick from one of two branch predictor. Critics can gather more information, since it starts a few cycles later than the prophet, so it can predict with higher accuracy. Critic either agrees or disagrees with the prophet prediction and determines the final prediction for the branch. Usually, prophet produce several future bits to provide information to critic to make a decision.

Experimental technique

As for prophet-critic hybrid branch predictor, I pick a simple 2-bit local predictor as prophet branch predictor and a simple 1-bit branch predictor as shown in figure 2.

Table

Description automatically generated with low confidence

Figure 2 structure of critic branch predictor

In this figure, it uses least used pointer to go down the table and enter the case which prophet makes a wrong prediction. Once prophet is asking critic to make a decision to process the branch outcome to icache and fetch. The critic will search the case(address + 4 future bits) in the table, and once it find out the case in the table with a valid bit. The critic will overwrite the outcome produced by the prophet with the taken flag in the critic table. Once the least used pointer reaches the last entry of the table, it will go over again and overwrite the first entry.

Before I implement the prophet/critic hybrid branch predictor, I take a look at the available predictor in pred folder in O3 Gem5. All the local predictor does not take history taken flag as an entry of the table, which is illustrated in Table 1.

|  |  |
| --- | --- |
| Index | Counter |
| 0x000 | 2 bit counter |
| 0x001 | 2 bit counter |
| 0x002 | 2 bit counter |
| 0x003 | 2 bit counter |

Table 1 original local predictor table

In order to increase the prediction accuracy for critic branch predictor. I revised 2-bit prophet branch predictor to table 2.

|  |  |
| --- | --- |
| Index | Counter |
| 0x000+taken | 2 bit counter |
| 0x000+not taken | 2 bit counter |
| 0x001+taken | 2 bit counter |
| 0x001+not taken | 2 bit counter |

Table 2 revised local predictor table

Since critics take four inputs of future bits, if prophet is the original local predictor, the input for the critic branch predictor will either be TTTT or FFFF.(if there are four future bits)

• If the first future bit is T, the counter will increase by 1 and following future bits stick to T.

• If the first future bit is NT, the counter will decrease by 1 and following future bits stick to NT.

In this case the critics cannot record the characteristics of the special case that prophet makes a wrong decision so that critics branch predictor cannot correct wrong edge case that prophet branch predictor cannot identify.

In order to further improve the performance of the prophet/critic hybrid branch predictor, I implement what paper suggested, a filtered critic.

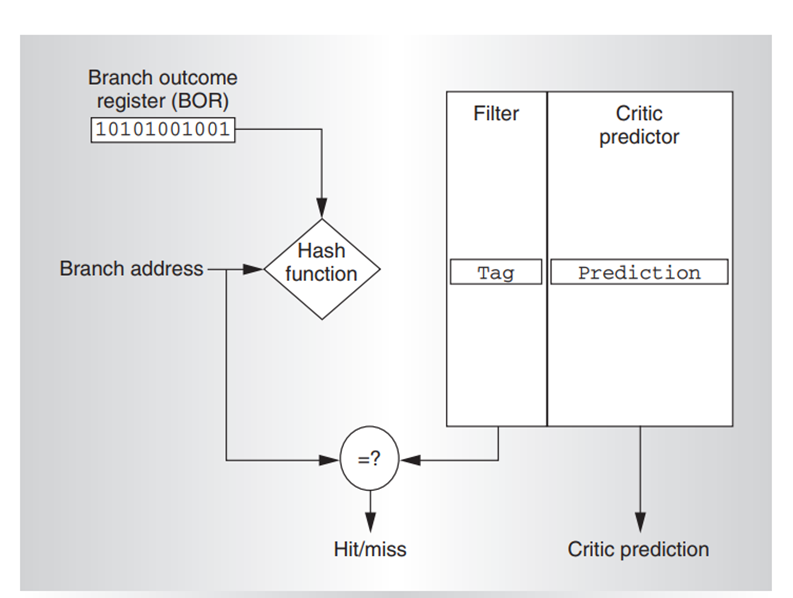


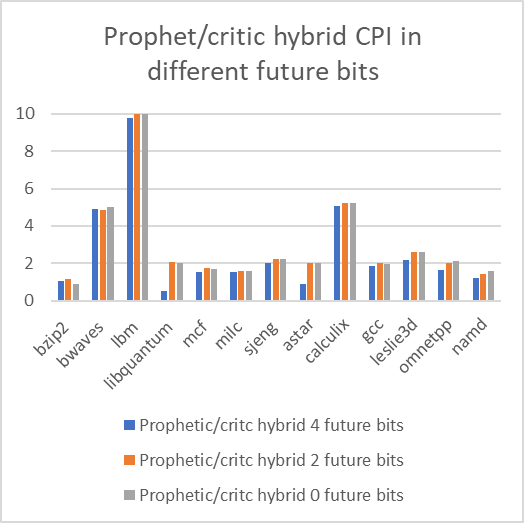
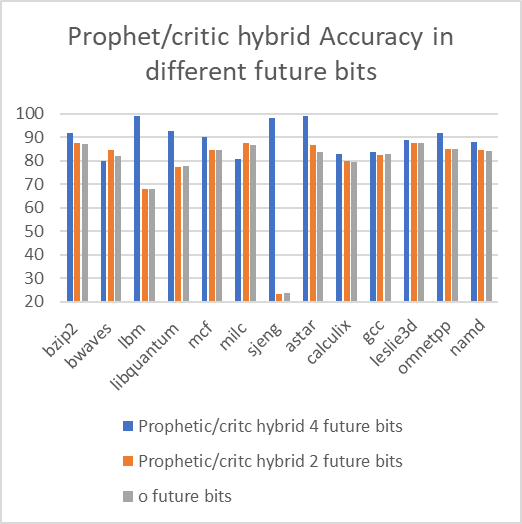
Figure 2 filtered critic

In this architecture, the table in critic branch predictor only records the branch that prophet mispredicts. Every time prophet predicts the outcome of the branch, if the branch pc and four future bits are available in the critic branch predictor, the critic branch predictor will overwrite the outcome generated by prophet branch predictor, otherwise the critic branch predictor will process the outcome of prophet branch predictor to I cache and fetch.

Presentation of result

First I made a comparison for single prophet branch predictor and prophet/critic hybrid branch predictor to figure out whether there is an improvement after using this implementation.

chart 1 Prophet/critic hybrid in different future bits

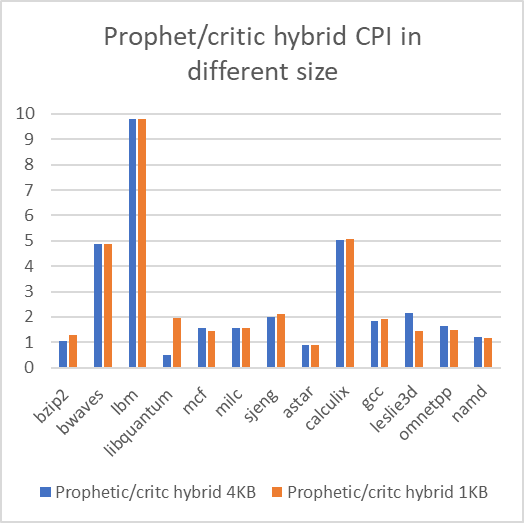
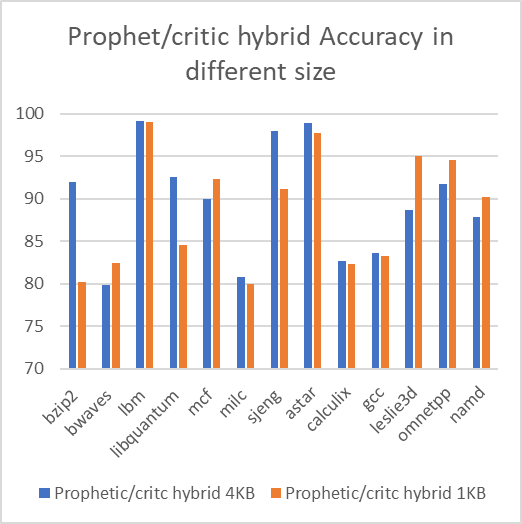


|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Prophet/critic hybrid 4 future bits | Prophet/critic hybrid 2 future bits | Prophet/critic hybrid 0 future bits | Fewer in mispredictions |
| Misprediction% | 10.33% | 21.59% | 22.15% | 53.4% |
| CPI | 2.625 | 2.691 | 2.9978 | 12.4% |

In the paper, there is a 24% fewer in misprediction, but I have 53.4% fewer in misprediction. The reason is that in the paper they compare prophet/critic hybrid with gskew branch predictor, which has much higher prediction rate than what I implement for Prophet itself. Therefore, I will observe a better misprediction decrease. And the reason why adding two future bits is not enough to reduce the misprediction is that two future bits pattern is simple. There are only four combinations, so there might be many different many-to-few mapping branches to the critic table with same entry and same pattern of future bits. This will pollute the table to reduce the prediction accuracy by a large amount.

Then I make a comparison with different sizes of the Prophet/critic branch predictors.

chart 2 Prophet/critic hybrid in different table size



|  |  |  |  |
| --- | --- | --- | --- |
|  | Prophet/critic hybrid 4KB | Prophet/critic hybrid 1KB | Fewer in mispredictions |
| Misprediction% | 10.33% | 12.34% | 9.78% |
| CPI | 2.625 | 2.691 | 2.5% |

In paper, they do not discuss the effect of size of the table of prophet/critic hybrid. From the above statistics, the bigger the size of the table, the lower misprediction it has, but the improvement is not a big factor.

In conclusion, the number of future bits will improve prediction accuracy starts from 4 bits, but the improvement after specific number of future bits will be gradually small. In addition, the more future bits, the more many-few-mapping it has, since the total index for critic is fixed. There will be many branch share with same entry in critic, resulting lower prediction accuracy. The size of the look-up table can also improve the prediction accuracy, but not too much.

Appendex(please delete in formal report)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Prophetic | | Piecewise | |
|  | Accuracy | CPI | Accuracy | CPI |
| bzip2 | 92.00 | 1.05 | 82.98 | 1.34 |
| bwaves | 79.85 | 4.88 | 85.17 | 4.82 |
| lbm | 99.12 | 9.80 | 99.04 | 9.7 |
| libquantum | 92.55 | 0.49 | 94.21 | 0.403 |
| mcf | 90.01 | 1.55 | 88.4 | 1.62 |
| milc | 80.75 | 1.55 | 87.2 | 1.56 |
| sjeng | 98.00 | 2.00 | 97.6 | 2 |
| astar | 98.87 | 0.90 | 98.84 | 0.895 |
| calculix | 82.71 | 5.04 | 83.46 | 5.04 |
| gcc | 83.65 | 1.85 | 83.8 | 1.7 |
| leslie3d | 88.70 | 2.17 | 97.116 | 1.37 |
| omnetpp | 91.70 | 1.63 | 90.3 | 1.7 |
| namd | 87.90 | 1.22 | 85.4 | 1.3 |
|  | Prophetic | | Piecewise | |
|  | Accuracy | CPI | Accuracy | CPI |
| bzip2 | 92.00 | 1.05 | 82.98 | 1.34 |
| bwaves | 79.85 | 4.88 | 85.17 | 4.82 |
| lbm | 99.12 | 9.80 | 99.04 | 9.7 |
| libquantum | 92.55 | 0.49 | 94.21 | 0.403 |
| mcf | 90.01 | 1.55 | 88.4 | 1.62 |
| milc | 80.75 | 1.55 | 87.2 | 1.56 |
| sjeng | 98.00 | 2.00 | 97.6 | 2 |
| astar | 98.87 | 0.90 | 98.84 | 0.895 |
| calculix | 82.71 | 5.04 | 83.46 | 5.04 |
| gcc | 83.65 | 1.85 | 83.8 | 1.7 |
| leslie3d | 88.70 | 2.17 | 97.116 | 1.37 |
| omnetpp | 91.70 | 1.63 | 90.3 | 1.7 |
| namd | 87.90 | 1.22 | 85.4 | 1.3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Prophetic 4KB | | Prophet 1KB | |
|  | Accuracy | CPI | Accuracy | CPI |
| bzip2 | 92.00 | 1.05 | 80.232 | 1.277889 |
| bwaves | 79.85 | 4.88 | 82.467 | 4.873011 |
| lbm | 99.12 | 9.80 | 99.009 | 9.799628 |
| libquantum | 92.55 | 0.49 | 84.532 | 1.952374 |
| mcf | 90.01 | 1.55 | 92.268 | 1.433664 |
| milc | 80.75 | 1.55 | 79.996 | 1.551371 |
| sjeng | 98.00 | 2.00 | 91.145 | 2.102214 |
| astar | 98.87 | 0.90 | 97.689 | 0.903001 |
| calculix | 82.71 | 5.04 | 82.328 | 5.056859 |
| gcc | 83.65 | 1.85 | 83.233 | 1.910776 |
| leslie3d | 88.70 | 2.17 | 94.996 | 1.450929 |
| omnetpp | 91.70 | 1.63 | 94.596 | 1.494883 |
| namd | 87.90 | 1.22 | 90.201 | 1.177948 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Prophetic 4KB | | Old Prophet 4KB | |
|  | Accuracy | CPI | Accuracy | CPI |
| bzip2 | 92.00 | 1.05 | 86.923 | 0.871400 |
| bwaves | 79.85 | 4.88 | 82.009 | 5.007452 |
| lbm | 99.12 | 9.80 | 67.981 | 9.985810 |
| libquantum | 92.55 | 0.49 | 77.610 | 2.036946 |
| mcf | 90.01 | 1.55 | 84.645 | 1.708197 |
| milc | 80.75 | 1.55 | 86.682 | 1.565876 |
| sjeng | 98.00 | 2.00 | 23.590 | 2.240085 |
| astar | 98.87 | 0.90 | 83.519 | 2.022027 |
| calculix | 82.71 | 5.04 | 79.521 | 5.246940 |
| gcc | 83.65 | 1.85 | 82.838 | 1.973385 |
| leslie3d | 88.70 | 2.17 | 87.748 | 2.615595 |
| omnetpp | 91.70 | 1.63 | 84.782 | 2.108989 |
| namd | 87.90 | 1.22 | 84.251 | 1.588223 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Prophetic 4 future bit | | Prophet 2 future bit | |
|  | Accuracy | CPI | Accuracy | CPI |
| bzip2 | 92.00 | 1.05 | 87.744 | 1.127527 |
| bwaves | 79.85 | 4.88 | 84.684 | 4.842879 |
| lbm | 99.12 | 9.80 | 68.000 | 9.985810 |
| libquantum | 92.55 | 0.49 | 77.518 | 2.039603 |
| mcf | 90.01 | 1.55 | 84.428 | 1.737105 |
| milc | 80.75 | 1.55 | 87.473 | 1.584638 |
| sjeng | 98.00 | 2.00 | 23.084 | 2.233215 |
| astar | 98.87 | 0.90 | 86.614 | 2.004843 |
| calculix | 82.71 | 5.04 | 79.811 | 5.246260 |
| gcc | 83.65 | 1.85 | 82.440 | 2.023480 |
| leslie3d | 88.70 | 2.17 | 87.733 | 2.617508 |
| omnetpp | 91.70 | 1.63 | 85.134 | 2.002444 |
| namd | 87.90 | 1.22 | 84.705 | 1.440401 |