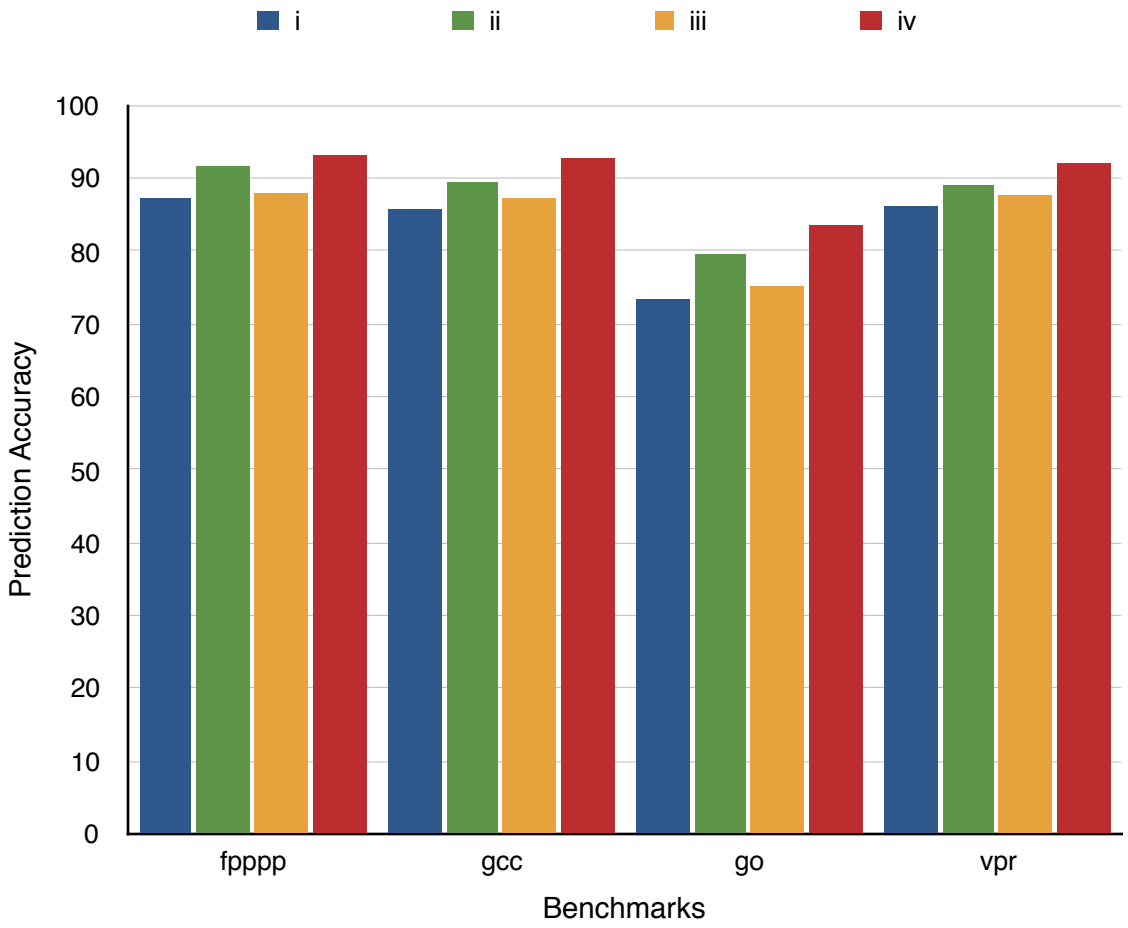
Jay Chang
39787114

CPEN 411 Computer Architecture
W1 2015

Assignment #3: Branch Prediction

Analysis Report

In this assignment, we were given the task of modelling different kind of branch prediction models. The four main models that we covered were: 1-bit predictor(i), 2-bit saturating counter(ii), 1-bit predictor with 1-bit correlation(iii), 2-bit saturating counter with 4-bit correlation(iv).



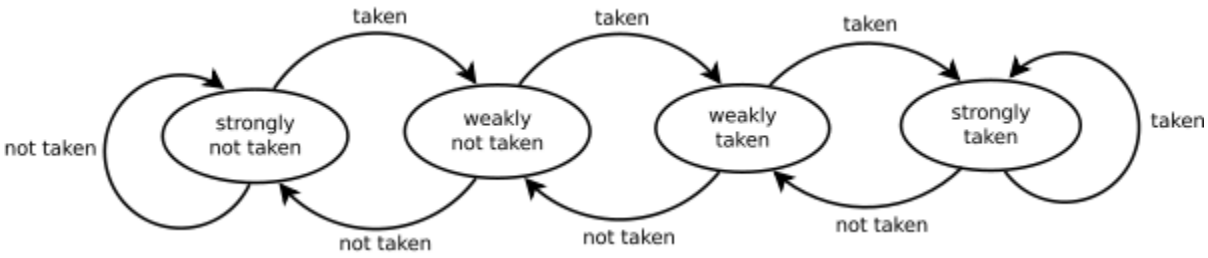
The figure above is the prediction accuracy for the 5 different branch prediction models using the four different benchmarks.

Interpreting the data from the figure above, out of the four default models (i to iv), we can notice that the least accurate model is the 1-bit predictor and the most accurate one is the 2 bit saturating counter with 4-bit correlation.

It is no surprise that the most basic model, 1-bit predictor(i), offers the lowest accuracy. This model works by checking the last branch condition and decides whether or not to jump for

the next branch condition based on this last branch condition. If there is minimal branch changes (eg. T T T T T), this model offers somewhat of a decent prediction as it will always predict correctly. However whenever there is a random changes (eg. NT NT NT T NT NT), there will be 2 miss predictions because this model only accounts for the *last branch condition* only. This method is even worst if there are constant changes (eg. T NT T NT T NT) because the previous branch condition will always be different from the next condition. If we can take into account more than just the last branch condition, it can help improve the accuracy of our model. The next models are an extension of the basic 1-bit predictor.

The second model that we will cover is the 2-bit saturating counter(ii). This model offers more accuracy over the 1-bit predictor because it accounts for much more than just the last branch condition. This model works by using a saturated counter and will predict the result that happens the most often.



This mage is how a 2-bit saturating counter works. The prediction is evaluated based on the previous branches

While this does not seem like much, this can improve some of the miss predictions of part i. Going back to the simple example before (NT NT T NT NT), this model will only have 1 miss prediction because of the saturating counter. It will see that the branch has changed, however will decide only to change the predicted outcome if the same result happens again as the prediction will go from strongly not taken to weakly not taken.

The third model that we will cover is the 1-bit predictor with 1-bit correlation. This model works by keeping a shared history of the previous branch condition (I used the global branch condition for this assignment). There are two branches, one for when the prediction is taken and one when the prediction is not taken. And depending on the outcome, we will determine which table to pull our prediction from on the next condition. One of the reason that this method (global correlation) doesn't offer significant changes compared to the basic 1-bit predictor is because different branches can have no correlation with each other, thus making the history "useless". This is the reason why this model is worst than the 2-bit saturating counter. However if there are correlation between the different branches, then it will offer an improvement over the basic 1-bit predictor.

The final default model that we replicated is a 2-bit saturating counter with 4-bit correlation. This model is a combination of (ii) and (iii) and it is the best predictor model out of the 4 default ones. One reason for the high accuracy is because this hybrid model offers a lot of benefits. It combines power of checking more than just a single branch condition and uses the history to determine how to check for the prediction. This model works by checking the history of conditions to determine which table of saturated counters to use. In addition, this model also checks the last 4 history by shifting in the last history into a 4-bits variable which we use to determine the table, unlike (iii) where it checks only the last history.