**An Analysis of Correlation and Predictability: What makes two level branch predictors work**

Marius Evers Sanjay J. Patel Robert S. Chappell Yale N. Patt

# Introduction:

This paper focuses on Analysis of co relation and predictability in branch prediction. The authors claims that there are numerous articles explaining the predictions fails. This paper is all about how to make the branch behavior a more predictable one. They have explained reasons of what makes a global two-level branch predictor works. The authors have demonstrated that it is required to store the information from a few of previous branches would suffice for the co relation based predictor to be accurate. One important thing that is understood from their discussion is that, Gshare two-level predictor will not be able to exploit the correlation completely. In the later sections, the authors have examined per-address predictability. They exploited and stated that repeated branches can be best predicted by “Per-Address History”. In addition to this they have shown experimental results supporting the fact. In one of the sections the authors have stated that PA predictor with separate loop would achieve higher accuracy than PA for gcc. The authors finally concluded that a Gshare two level predictors does not fully exploit the correlation and proposed few methods to improve. In addition to this, they have further concluded that Per address static prediction can be improvised by implementing some dynamic schemes.

# Conclusion:

It is a known fact that a single predictor will not be able to handle each and every scenario. Even if we try to use a single predictor it ends up in diminished performance. The fact can be fast understood as global history scheme will give good performance on correlation between the branches but underperforms in the scenario where we have iterative branches. The Per Address branch predictor seems to be a good solution for handling the Iterative loops but crumbles when it comes to handling correlation. It is quite possible to have both correlated and iterative loops in a piece of code. As stated, using only type of predictor wouldn’t solve the purpose. The solution to this can be to use a tournament branch predictor, which implements the best of both worlds. In terms of implementation the tournament branch predictor is correlated branch predictor based on the last ‘m’ branches, accessed by global history and local per branch prediction, acceded by Program counter (PC) [1]

**References:**

https://courses.cs.washington.edu/courses/csep548/06au/lectures/branchPred.pdf