# Branches

Computer architecture work on the base of instructions. Instruction set work one by after one. One after one is the normal flow without any branches and interrupt. For the most advanced level we use the branches in our program. Branches basically terminate the normal flow (which is one after one instruction execution) and jump to another instruction and execute it. For the assembly language we can use the branch as loop if we need it work fine.

## Classification of Branch

Branches has divided into sub category as

* Direct branches
* Indirect branches

Direct branches are branches in which we pass the direct value and execute it. In the case of indirect we get the data from the register.

# Branches Predication

Branches have important role on the speed of program. If we want to design the architecture like computer architecture which are used ARM and MIPS type then branches are more important because if we predicate the true branches then we can design the faster architecture.

## Techniques

* Predicting branches with long period
* Predicting based on address correlation
* Multi hybrid branch predication

## Predicting branches with long period

Predication can be determining with loop and nested loop. In loop-based predication we use the history. Kampe discover the “prediction with long period”. He observed a lot of branches execute due to the long period. After a lot of experiment, he observes that for the period 213 bits required for per branch period. As compare to history-based branches, we require only 52 bits for the branch predication. Based on the last experiment he takes the DFT of the branches period and the period are not corelate to each other. For the long period we use the hybrid scheme for the branch prediction. In the hybrid case some are static BP, some dynamic BP, PAp and GAp.

We can represent the taking or non-taking branches as 1 and 0. The best for the analysis is that we use the frequency domain analysis. Take the FT of the taking branches and its magnitude give the peaks and normal value. We can subtract the normal value and get only branches peak. In the time domain we can’t fine the peak as we did in frequency domain.

## Predicting based on address correlation

Gao work on the prediction on address correlation. He observes that branches depend on the long-latency caches misses. It only occurs when we reload the irregular data every time. Branch predication using the leveraging history give the poor performance. He designs the leverage address-branch for the improvement in performance. He observes address in some data structure remain same for a long time. it occurs in case if link list most of the time, because last node holds the same address for the long time until another node not extend it. In this way we can determine the output by using the address instead using the actual data. Because if address same in next iteration it must be data same. For the same address we don’t need to check the actual value again and again in this way performance increase with using the address correlation concept.

## Multi hybrid branch predication

Evers observed that if we combine two and more than two branch predication technique, we get the better result. In this method, he uses counter of 2 bits in every branch predication method which are combined and make it hybrid. In the start initialize every counter with 3 value. If any one branch predication method finds the value 3 in counter than it means it predicate the result otherwise decrease by 1 in remaining all. During this need to sure that one of them is must with 3 counter value.