Branch Predictor Report

Akash (2022CSB1064)
Aniket Kumar Sahil (2022CSB1067)

Implementation Details:

1. Compare the opcodes of the instruction to separate out the branch instructions.

2. Check if branching occurs or not and store the result in a vector for checking :

if SB instruction : extract out the immediate value and add it to the pc to check if the next instruction

if they both match then branching occurs else not

For the rest instructions : We have jalr and jal other than those SB instruction and there is no condition that needs to be met for their branching, hence branching always occur for them then we have made separate functions for each branch predictor and we run them in a loop and they update all the necessary values like predictions and accuracy in separate variables finally we print out all the necessary information

	Bubble_test_lab	Fac_test_lab	sort_test_lab	sort_test_lab	Recursion_test_lab	wikisort_test_lab	recursion_test_lab
Always Taken	43.3526	72.4425	71.0993	74.518	76.56	69.9638	88.1045
Always Not Taken	56.6474	27.5575	28.9007	25.482	23.44	30.0362	11.8955
One Bit	97.1571	90.8259	96.7603	94.5191	96.3638	96.9859	82.5039
Two Bit	98.384	92.067	97.3449	95.2484	96.5174	97.7713	84.1722

Observations:

1. 2-bit predictors are better than 1-bit predictors.
2. Dynamic predicting is better than Static predicting.

Git Hub Link: https://github.com/bInAryY-bArD/CS204_mini_project_1/

Theory

Branch prediction is a technique in computer architecture that guesses the outcome of conditional branch instructions to minimize pipeline stalls.

There are 4 types :

1. Always Taken: Predicts that a branch will always be taken.

2. Always Not Taken: Predicts that a branch will never be taken.

3. One-Bit Predictor: Uses a single bit to remember whether a branch was taken or not taken in the previous execution.

4. Two-Bit Predictor: Uses a two-bit saturating counter to track the history of a branch being taken or not taken, providing better prediction accuracy compared to the one-bit predictor.