# Tournament Branch Predictor

*CS6600: Computer Architecture*

Arjun Menon V, Akilesh Kannan

EE18B104, EE18B122

The code developed for this assignment can be found here - link

## 1 Introduction

Architecture techniques that exploit Instruction-Level Parallelism, such as Pipelining and Superscalar execution, are often adversely affected by stalls in the pipeline due to branch instructions. With branch instructions forming a significant portion of a typical program, it is imperative to use effective dynamic (run-time) branch prediction mechanisms to make the most out of ILP.

In this assignment, we implement the Tournament Branch Predictor [1] using the framework provided by JILP for the 2016 version of the Branch Prediction Championship (CBP2016). A quick summary of some popular dynamic branch prediction ideas is provided leading up to the Tournament Predictor.

### 1.1 4-state FSM based Predictor

A four-state Finite State Machine is used to predict the branch direction based on the past history of branch directions that were taken. The states 00 and 11 correspond to a strong prediction of `NOT-TAKEN` and `TAKEN` respectively- the predicted direction is changed only after at least two consecutive incorrect decisions. The local and global predictors in the Tournament Predictor use $n$-bit ($2^n$ state) FSM based predictors (in our implementation, we use $n = 2$).

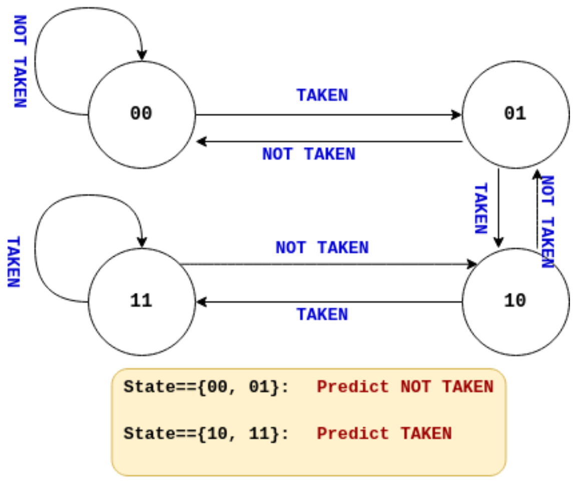
NOT TAKEN
00
TAKEN
01
NOT TAKEN
TAKEN
NOT TAKEN
TAKEN
NOT TAKEN
11
10
TAKEN
State=={00, 01}: Predict NOT TAKEN
State=={10, 11}: Predict TAKEN

Fig. 1: 4-state FSM based Predictor. Text in blue indicates the true branch direction as determined in the execution stage of the pipeline

## 1.2 Correlating Branch Predictor

Often, the branch results of instructions with similar branch history are correlated. To account this behaviour in the prediction mechanism, a correlating branch predictor extends the FSM-based predictor by maintaining multiple FSM counters for different branch histories. Branch instructions with the same branch history share the FSM state. To facilitate this, a table of branch histories, indexed by the branch instruction address, is maintained with each entry containing the branch history of the instructions mapped to it. Since the History Table size is limited, only a few bits from the PC is required to index the table.

When there is only one entry in the history table, the correlating predictor essentially tracks global prediction history. With sufficiently large history tables, each FSM in the correlating predictor uses local branch history information to make the decision.

## 1.3 The Tournament Branch Predictor [1]

Different programs exhibit varying trends in the correlation between branch instructions- some functions may have a high correlation in the global branch history while some others may exhibit correlation locally. This trend may even vary in the same program. A tournament branch predictor integrates a global correlating predictor with a local correlating predictor. This lets the prediction mechanism adapt to the program's correlation trend dynamically.

When both the local and global predictors agree on a prediction, the prediction is used, else the tournament predictor uses yet another FSM (choice predictor) to choose between the local predictor and the global predictor. The choice predictor uses a 4-state FSM similar to the one discussed. When the choice predictor's state is equal to 00 or 01, the local predictor is chosen, else the global predictor is chosen. Upon resolution of the branch direction, the state of the FSM is updated in the direction favouring the correct predictor.

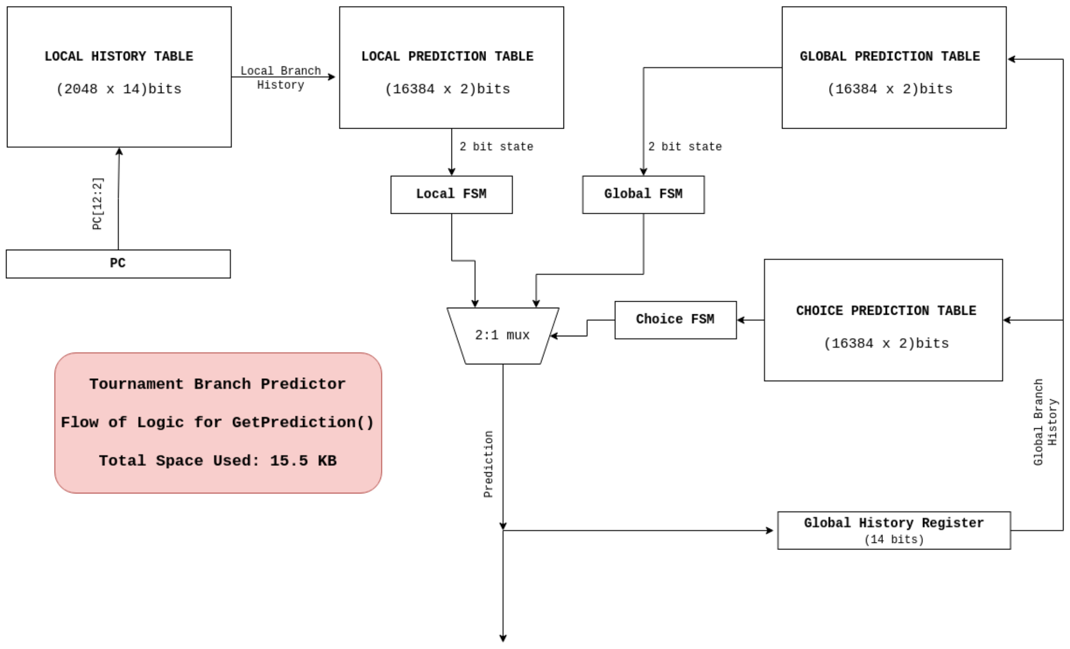

Fig. 2: Tournament Branch Predictor

### 1.3.1 Storage Budget Justification

| Structure | Number | Size/structure (in bits) | Total Size (in bits) |
|---|---|---|---|
| Global Predictor | | | |
| GHR | 1 | 14 | 14 |
| GPT counters | 16384 | 2 | 32678 (32K) |
| Local Predictor | | | |
| LHT entry | 2048 | 14 | 28672 (28K) |
| LPT counters | 16384 | 2 | 32768 (32K) |
| Choice Predictor | | | |
| CPT counters | 16384 | 2 | 32768 (32K) |
| Total | | | 124K + 14 $\approx$ 15.5KB |

Tab. 1: Storage allocation

% Used out of maximum allowed budget (16KB) $= 96.88\%$

## 2 Code Structure - `class PREDICTOR`

A data structure given by the framework, with the following fields and methods to help in designing the branch predictor.

## 2.1 Methods

- `PREDICTOR()`: Constructor to initialise the tables, `GHR` and counters.
- `~PREDICTOR()`: Destructor to delete all dynamically allocated pointers.
- `GetPrediction()`: Takes as input the `PC` value corresponding to the branch instruction, and returns `TAKEN` (`True`) or `NOT-TAKEN` (`False`).
- `UpdatePrediction()`: Takes as input the `PC`, predicted direction of branch (`predDir`) and true direction of the branch (`resolveDir`) and updates the `GHR`, local history, local predictor counter, global predictor counter and the choice predictor counter that were used to predict the direction.

## 2.2 Fields

### 2.2.1 Global Predictor Related

- `ghr`: Global History Register
- `gpt`: Global Predictor Table
- `numGptEntries`: Number of entries in the GPT

### 2.2.2 Local Predictor Related

- `lht`: Local History Table
- `numLhtEntries`: Number of entries in the LHT
- `lpt`: Local Predictor Table
- `numLptEntries`: Number of entries in the LPT

### 2.2.3 Choice Predictor Related

- `cpt`: Choice Predictor Table
- `numCptEntries`: Number of entries in the CPT

## 3 Results

We had experimented with the initial choice in the choice predictor to observe how the initial bias of the choice predictor affected the overall outcome of the BPU. In one configuration, we started all the counters with a *Weakly Local Bias* (`01`) and in the other, a `Weakly Global Bias` (`10`).

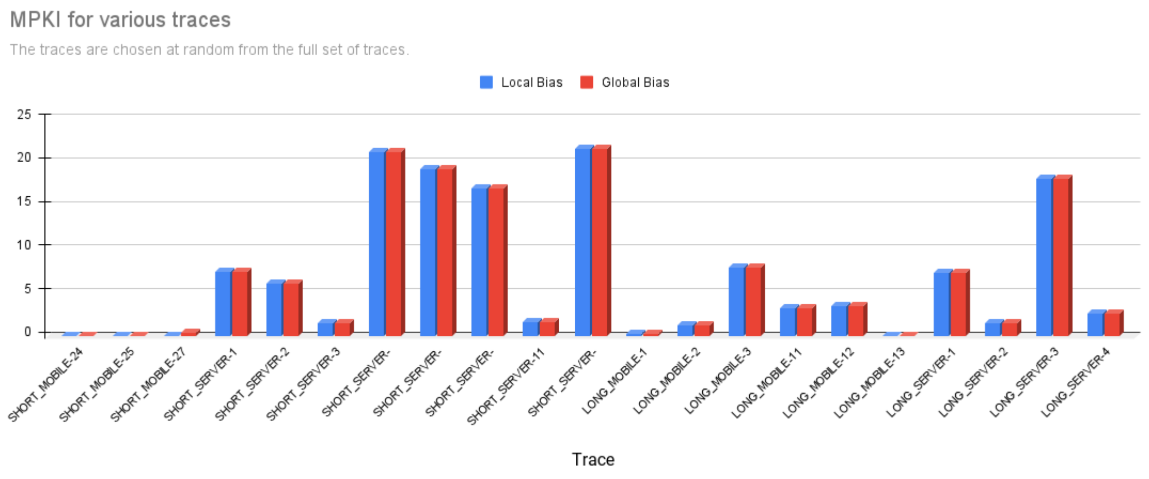

Fig. 3: MisPrediction per 1K Instructions

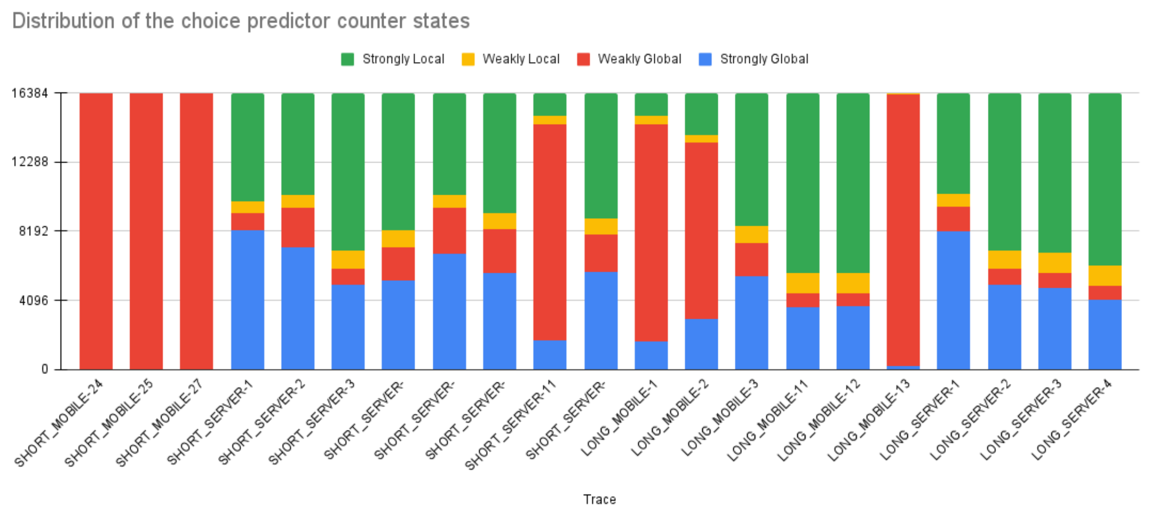

Fig. 4: Choice Distribution with Global Bias

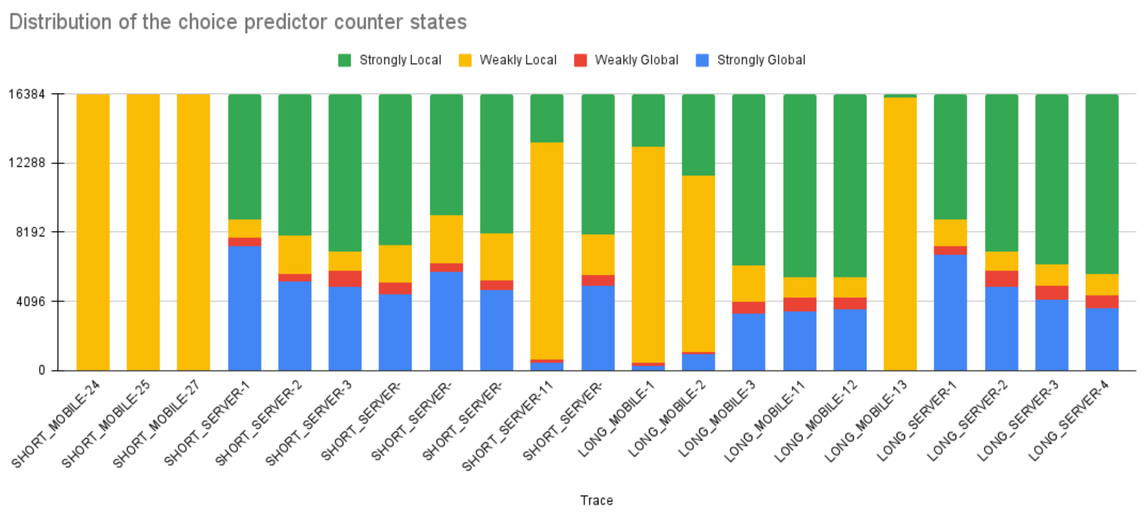

Fig. 5: Choice Distribution with Local Bias

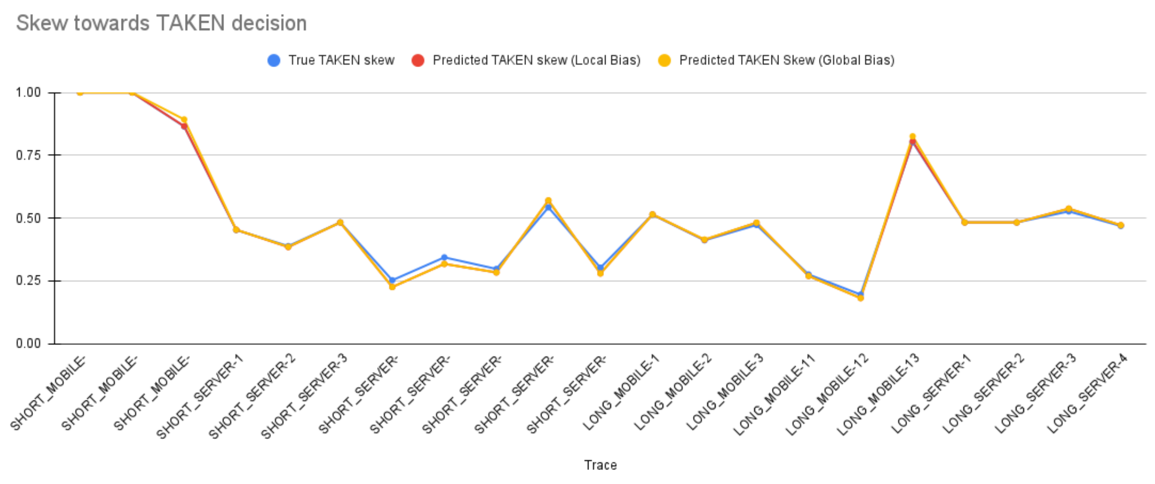

Fig. 6: Skew towards `TAKEN` branches

We can see that the MPKI doesn't change much between the two configurations. This is expected because we don't expect the initial bias of the choice predictor to affect the steady state misprediction rate.

However, we can clearly see that the bias does affect the choice predictor's decisions. A global bias would make the predictor choose the global predictor more often and vice-versa.

## References

[1] McFarling, S. Combining branch predictors.