Results Report

Thomas Cumming (Matric No: s1230426)

Static

The following results were obtained by doing

```
$ python BPSim.py s [gcc_trace_file_loc / mcf_trace_file_loc]
```

at a DICE terminal and choosing each scheme the simulator provides.

Scheme(misprediction rate) Trace File	always taken	always not taken	profile guided
gcc trace	23.61%	76.39%	6.63%
mcf trace	32.24%	67.76%	10.94%

We need to make one key assumption when we use a profile guided approach; that the relatively frequency estimate is indicative of branch behaviour at a general level in a trace file. The results imply this is the case, though, since using the profile guided scheme yielded the lowest misprediction rate for both files. It incorporates more information since it is based on relative frequencies rather than just flat counts. There are a few issues with this approach though - for example, how do we decide on a threshold (in this coursework it was chosen to be 50%)? Also, there may be other methods of estimation that might produce better results. Further experimentation would be required to determine the solutions to these issues.

We can also make some interesting inferences about the code of gcc and mcf; for example: we were wrong 76.39% of the time (according to the simulation) when we predicted every branch as not taken. This shows most branches *are* taken suggesting that gcc must have a very varied execution path, perhaps a reflection on it's complexity; this would make sense considering gcc is a compiler. We could interpret the results for the mcf trace in the same way. Comparison would imply mcf is not as complex as gcc.

Dynamic

The following results were obtained by doing

```
        $ python BPSim.py d [gcc_trace_file_loc / mcf_trace_file_loc]
```

at a DICE terminal and choosing each history length available.

History length(bit number) Trace File	4	8	12	16
gcc trace	51.05%	51.81%	58.48%	65.88%
mcf trace	50.21%	54.36%	56.72%	61.56%