EE 7700-4 **Homework 1** **Due: 14 September 1998**

Problem 0 below has been pre-solved, the solution and other files needed can be found in /data4/koppel/tca/hw02 on the ECE Solaris systems. The techniques used in its solution can be applied to the other problem.

Problem 0: Using Shade, write a program to simulate a gshare branch predictor. After a simulation run the program should report the prediction accuracy, the BHT hit ratio, and fraction of the BHT used. Define the hit ratio as the number of times that a read of the BHT by a branch finds data that was last written by the branch divided by the number of times read. Note that the predictor being simulated, since it does not store tags, cannot detect misses. The misses are detected by the simulator. For that reason the predictor cannot take special action (such as predicting a backward branch as taken and a forward branch as not taken) on a miss. Lower BHT hit ratio numbers indicate more interference between branches. Program output may look like the following:

```
[sol] % echo xeyes | pred
X connection to omega:0.0 broken (explicit kill or server shutdown).
Number of BHT entries, 2^10; pattern history size, 4 outcomes.
Of 1162540 branches, 85.245% predicted correctly, hit ratio 0.935
Table entries used: 100.000%
[sol] % echo hello | pred
Hello, world!
Number of BHT entries, 2^10; pattern history size, 4 outcomes.
Of 137685 branches, 88.927% predicted correctly, hit ratio 0.947
Table entries used: 78.711%
```

Problem 1: Develop a better branch prediction scheme. Describe the scheme and determine how much memory it will take to implement it. Explain how the new scheme is supposed to improve performance over existing schemes and using Shade determine the prediction accuracy of the new scheme on some benchmark programs for a small and large (amount of memory) sizes.

For those who prefer a less open-ended problem develop the branch predictor described below:

Initially, as the length of the outcome history used in a gshare predictor is increased (up to a point) prediction accuracy increases. If the length of the outcome history is made too long then performance suffers, one reason is aliasing: several branches can be mapped to the same entry because the outcome history spreads an individual branch over many entries.

Develop a branch predictor based on gshare that can make use of a longer outcome pattern but that avoids the aliasing problems by hashing the longer pattern down to fewer bits or by only using a portion of the pattern (other than the most recent outcomes).