CMP SCI 635: Modern Computer Architecture

An Analysis of Correlation and Predictability: What Makes Two-Level Branch Predictors Work

Name: Kunjal Panchal Date: 17th Oct, 2019 Student ID: 32126469

Paper: 12 – Instruction Level Parallelism and Branch Prediction [Evers98]

Strengths:

1. The paper clearly described the sufficient history length to make an optimal prediction about the distance to correlated branches. They experimented with 8 to 32 previous instructions and found that at 12, the results in accuracy of predictions start getting saturated. This indicates that the most correlated branches are close together.

2. Quantitively described why a hybrid between correlation and per-address predictions would work best. Also, showed what portion of program would have what kind of branches in general, which would obviously change from program to program, compiler to compiler but it still gives rough estimate on which direction the future research should be conducted on. E.g., how to identify the branches before the beginning of the loop or a subroutine call.

Weaknesses:

1. Can assuming instruction window < 256 be detrimental considering the loop sizes of AI related programs?

2. In most cases, gshare still gives better or equal results, so the comparison of 1/2/3 vs 16 previous instruction window should depend on some other factor like compiler compatibility. Are compilers and branch predictors, in any way, made to work together to provide an optimal prediction? If so, choosing more established gshare still makes sense.

3. Given the big chunk of branches which doesn't belong to any group, one might assume that even a hybrid predictor cannot do anything in the case where a branch has no past pattern footprint and is just occurring randomly for once, it can still disrupt the pipeline, and that portion of instructions seems pretty high, without any solution provided for those.

Questions/Assertions:

1. "*Interference-free gshare*" What does interference mean in branch prediction context?

2. Wouldn't selective history be more time consuming? There is no time vs accuracy trade-off given in the paper, does that mean there is no one method advantageous temporally?

3. What does the sentence "*…since the for-type and while-type branches only differ in the directions predicted…*" mean?

4. Is there any practical/commercial implementation of gshare + PA hybrid?

5. Can this whole process of per-address branch prediction be conducted in compilers?