LSU EE 7700-1 **Homework 3** **Due: 30 April 2007**

The dataset files for this assignment, the same as those used for Homework 1, are located in directory /home/faculty/koppel/pub/ds/2007/batch.sm.0098 which should be accessible from ECE Linux machines. The dataset files are named using *run ids*, view the web page `index.html` to see the benchmark and machine configuration. If your system is set up properly you can view the dataset by clicking the run id.

The simulated machines have the following characteristics:

- Dynamically scheduled. (Out of order execution.)
- Superscalar. Issue (fetch/decode/commit) width (IW) is either 2, 4, 8, or 16 instructions per cycle (depending on configuration).
- Reorder buffer (ROB) size is either 64, 128, 256, or 512 entries.
- Can predict either 1 or 3 blocks per cycle.
- Uses YAGS predictor with 8-branch GHR and 2^{16}-entry PHTs.

For the following questions refer to the rePlay paper: Brian Fahs, Satarupa Bose, Matthew Crum, Brian Slechta, Francesco Spadini, Tony Tung, Sanjay J. Patel and Steven S. Lumetta, "Performance Characterization of a Hardware Mechanism for Dynamic Optimization," in the proceedings of the International Symposium on Microarchitecture, December 2001, pp. 16-27. It is available on the class references Web page under critical path compression or directly at `http://www.ece.lsu.edu/tca/papers/fahs01performance.pdf`.

For the following assignment it will be necessary to find sections of code containing strongly biased branches. In class we saw that PSE shows branch information (such as prediction ratio) in the annotation pane and this information can be used to find strongly biased branches. There is also information in the disassembly pane that can be used to more quickly find easy to predict (and perhaps strongly biased) branches. Some instructions are preceded by a gray *W* and a digit. The *W* stands for weakness (referring to the prediction) if the digit is zero the prediction is accurate, higher numbers indicate lower accuracy (weaker) predictions.

In a recent change to PSE (so far just for this class) the disassembly pane shows instructions that write unused registers or memory locations in a strikethrough font (crossed out). These instructions might be eliminated by rePlay's dead-code removal optimization. So far PSE does not show all dead instructions in strike through, but that may be available before the assignment is due. If so, something will be posted.

Problem 1: Show how rePlay can improve an execute-limited (using Fields' definition) region. Try to find a region in which the improvement will be large, both on the region itself and on the benchmark as a whole. A region should cover thousands of instructions, an entire segment or a large fraction of one. Several such regions have been discussed in class, so this part should be easy.

Note that when looking for runs with lots of execute-limited regions one should look at IW (fetch/decode width) and ROB size.

- Give the run id, benchmark, and a segment number.
- Show the original code and the resulting frame.
- Estimate the performance improvement of the region containing the code.
- Estimate the performance improvement of the benchmark as a whole. Note that in PSE's overview plot, when the mouse is over a segment the address of the first instruction is shown. It is okay to use this to estimate how many segments are similar to the one you've identified. One might also use instruction counts for branches and loads to solve this part.

Problem 2: Show how rePlay can improve a fetch-limited region. Try to find a region in which the improvement will be large, both on the region itself and on the benchmark as a whole. A region should cover thousands of instructions, an entire segment or a large fraction of one.

- Give the run id, benchmark, and a segment number.
- Show the original code and the resulting frame.
- Estimate the performance improvement of the region containing the code.
- Estimate the performance improvement of the benchmark as a whole.

Problem 3: Show how rePlay can improve a commit-limited (using Fields' definition) region. Recall that a region is commit-limited if performance would improve with a larger ROB.

Note that this is the most difficult problem of this assignment.

- Give the run id, benchmark, and a segment number.
- Show the original code and the resulting frame.
- **Provide a sketch (or printout) showing how replay will improve performance. The sketch should show the instruction suffering due to the commit-limiting and should show the impact that rePlay would have.**
- Estimate the performance improvement of the region containing the code.
- Estimate the performance improvement of the benchmark as a whole.