CS-622A
Project
Proposal

Sumit Lahiri	Group No	Radhit Dedania
19111274	13	150544

PART-I : Problem Statement

We have chosen to work on "Data Prefetching". We will focus only on hardware data prefetching techniques. We will study different standard prefetching techniques listed in the reference section. At first, we will consider the effect of prefetching on single-core machines only. Gradually, we will move to prefetching on multi-core machine(if time permits) as it is a very complicated problem.

PART-II : Expected Outcomes/Results

We will analyse and compare some standard prefetching algorithms under different combinations of external parameters like branch predictor, cache levels, cache replacement policy along with explanations of observed behaviour . Using the analysis, we would try to figure out some of their shortfalls. We will also tweak the prefetcher(one of them) to address it's shortcoming and explain the observed behaviour.

PART-III : Tools, Benchmarks and Procedure

For this project, we will use **ChampSim** Simulator for analysing the prefetchers in different external scenarios and drawing a comparison between them. As far as benchmarks are concerned, we will use **SPEC CPU 2017** traces which are generally used in a standard **Data Prefetching Championship(DPC)**. There are two types of traces in that benchmark. One starting with **4XX** and another starting with **6XX**. We will only test the prefetcher on traces starting with **6XX** so as to be consistent with the DPC testing criteria. The tentative procedure of our project is outlined below:

- Studying certain state-of-the-art prefetching techniques
- Analysing these prefetchers(using existing implementation) on the above benchmarks and explaining the observations
- Considering a set of different scenarios(as mentioned above), finding out the best prefetcher for each case along with explanation of observations in terms of comparison between different prefetchers
- If time permits, we will also implement one of these prefetchers from scratch and compare it's execution on simulator with the original available implementation by giving reasons for observed behaviour.
- If time permits, we will make changes at fine granularity in the one of the standard prefetcher to overcome some shortfalls along with explanation of the effect(good or bad) of that change.

References

Papers

- Best-Offset Hardware Prefetching(Pierre Michaud)
  https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7446087
- Combining Local and Global History for High Performance Data Prefetching(Martin Dimitrov and Huiyang Zhou)
  https://pdfs.semanticscholar.org/2957/200e36b7cf809553c706e17ffadd722bed2a.pdf
- A Hybrid Adaptive Feedback Based Prefetcher(Santhosh Verma, David M. Koppelman and Lu Peng)
  https://www.jilp.org/dpc/online/papers/07verma.pdf
- Efficiently Prefetching Complex Address Patterns(Manjunath Shevgoor, Sahil Koladiya and Rajeev Balasubramonian) https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7856594&tag=1

Simulator

- ChampSim Simulator
  https://github.com/ChampSim/ChampSim

Benchmark Traces

- SPEC CPU 2017 traces(starting with 6XX)
  http://hpca23.cse.tamu.edu/champsim-traces/speccpu/

Data Prefetching Championship(DPC)

https://dpc3.compas.cs.stonybrook.edu/?SW_IS