

## **CS 745, Milestone Report** **Nathan Snyder and Q Hong**

### **Major Changes**

There have not been any major changes to what we're doing in our project, but we have been able to refine our initially somewhat vague plans of "applying prefetching in the LLVM compiler". As we investigated the specific organization and capabilities of the LLVM compiler, we gained a better perspective on how the abstract prefetching techniques discussed in class map onto the concrete implementation of LLVM. For prefetching accesses to recursive data structures, the insertion of greedy prefetches works nicely but LLVM offers no way (that we have found after a great deal of searching documentation) to modify struct types in the way that history pointers require. Also, rather than having the RDS prefetch optimization be implemented as a single pass as initially expected, we found it was simpler to use two separate LLVM passes to catch different types of traversals (loop based and recursive function based).

### **What You Have Accomplished So Far**

So far we have written code that automatically identifies recursive data structure traversals in recursive functions and loops and inserts prefetch instructions for the addresses those traversals will use in the future. These prefetches take the form of calls to the LLVM prefetch intrinsic function, which is eventually compiled to an actual x86 prefetch instruction rather than a function call (intrinsic functions were designed as a way to extend the LLVM language without needing to update all of the LLVM passes). We have also established the constraints on where these prefetches can be placed such that the address computation cannot crash the program. In other prefetching optimizations, we have written code which identifies array traversal loops and gathers all of the information we need to make informed prefetching decisions. Lastly, we have created test code which presents opportunities for our optimizations, both to check that they are doing what they're supposed to and eventually to quantify the improvement achieved.

### **Meeting Your Milestone**

We are roughly where we wanted to be at this point, and do not anticipate needing to scale back our project goals.

### **Surprises**

Figuring out how to use LLVM's prefetching intrinsic function was much more difficult than expected, due to almost non-existent documentation. We got around this by finding posts about related issues on the LLVM developers boards and by simply experimenting until we got them working.

### **Revised Schedule**

Week 5 (April 14 - April 21)

Implement prefetching of conditionally needed data. Insert prefetches for array traversal loops.

Week 6 (April 22 - April 27)

Try heuristics for not inserting low-value prefetches. Gather performance data. Prepare final report and poster.

### **Resources Needed**

We still have all of the resources we will need to complete this project.