INTRODUCTION

Data prefetching is a hard problem.
- Hardware Prefetching
  - can only prefetch for simple memory access pattern
- Software Prefetching
  - extracting prefetch kernel is hard
  - prefetch has to be done at the right time to guarantee timeliness
    - not only the program, but the input & hardware affect the effectiveness and timeliness of prefetching

RPG2
- a pure-software system that operates on running C/C++ programs, profiling them, injecting prefetch instructions, and then tuning those prefetches to maximize performance
- can provide a comparable speedup to the best profile-guided prefetching insertion compilers
- can also respond when prefetching ends up being harmful and roll back to the original code

THE DESIGN OF RPG2

- code generation
  - original code
  - optimized code

Phase 3 - Runtime Code Insertion

Phase 4 - Monitoring and Tuning
RPG2 repeatedly tunes the prefetch distance based on the IPC collected at runtime until it finds the optimal prefetch distance.
- stage 1: decide the searching direction
- stage 2: double the jump size to compute the new prefetch distance in the chosen direction
- stage 3: binary search within this interval to identify the local optimum

Also support rolling back to the original code if the performance measurement result shows that prefetch is harmful to performance.

EVALUATION

Performance

Other measurements

Overhead

benchmark | pr | sssp | bfs | bc | is | randacc | cg
---|---|---|---|---|---|---|---
RPG2 exec (s) | 7.6 | 7.9 | 7.8 | 8.5 | 8.6 | 7.9 | 8.8
BOLT (ms) | 30.3 | 28.4 | 28.6 | 29.1 | 27.2 | 32.0 | 28.4
code insert (ms) | 3.9 | 3.3 | 3.1 | 3.4 | 3.3 | 3.0 | 2.9
1x pd edit (ms) | 1.2 | 1.1 | 1.4 | 1.2 | 1.4 | 1.1 | 1.1
# pd edit | 11.1 | 11.3 | 11.5 | 12.7 | 11.4 | 8.8 | 12.0