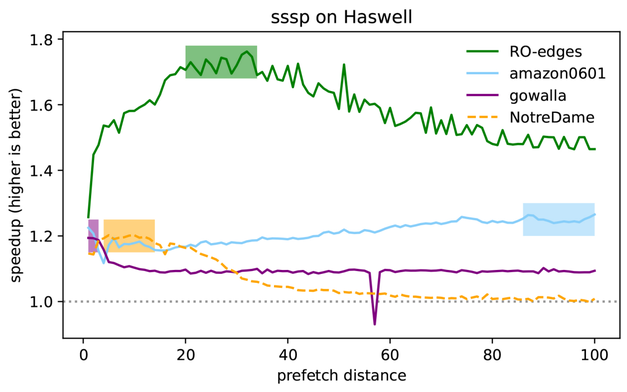
RPG2: Robust Profile-Guided Runtime Prefetch Generation

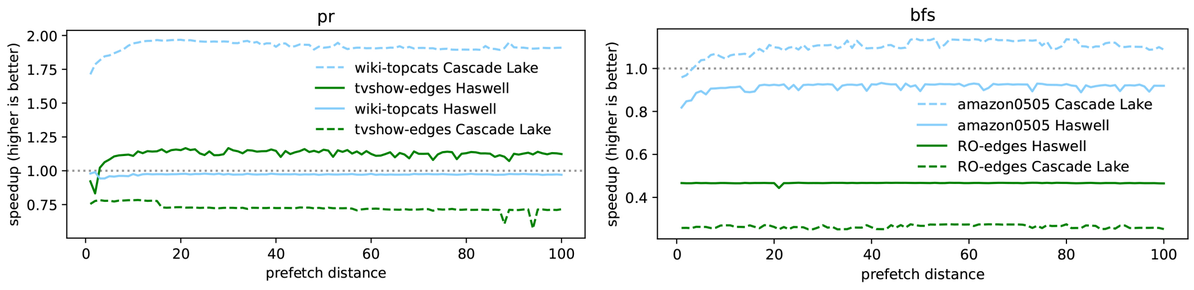
Y. Zhang, N. Sobotka, S. Park, S. Jamilan, T. A. Khan, B. Kasikci, G. Pokam, H. Litz, J. Devietti

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 effectness and timeliness of prefetching

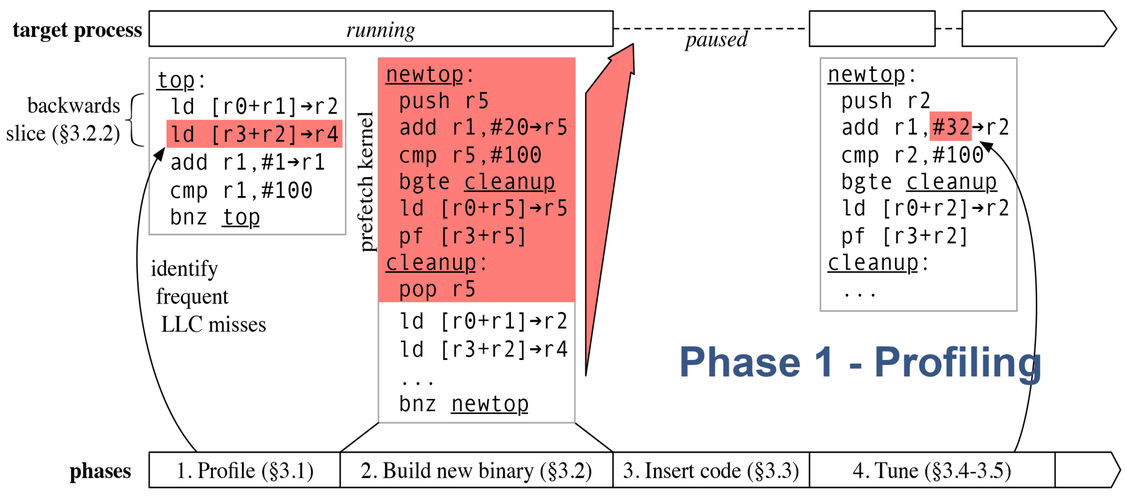




RPG2

- 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



Phase 1 - Profiling

- use perf's Processor Event-Based Sampling (PEBS) event `MEM_LOAD_RETIRED.L3_MISS/ppp` to identify LLC misses

Phase 2 - Code Analysis & Generation

- **prefetch catagories**

  support loads that fall into these three categories, but these patterns general enough to match many code patterns

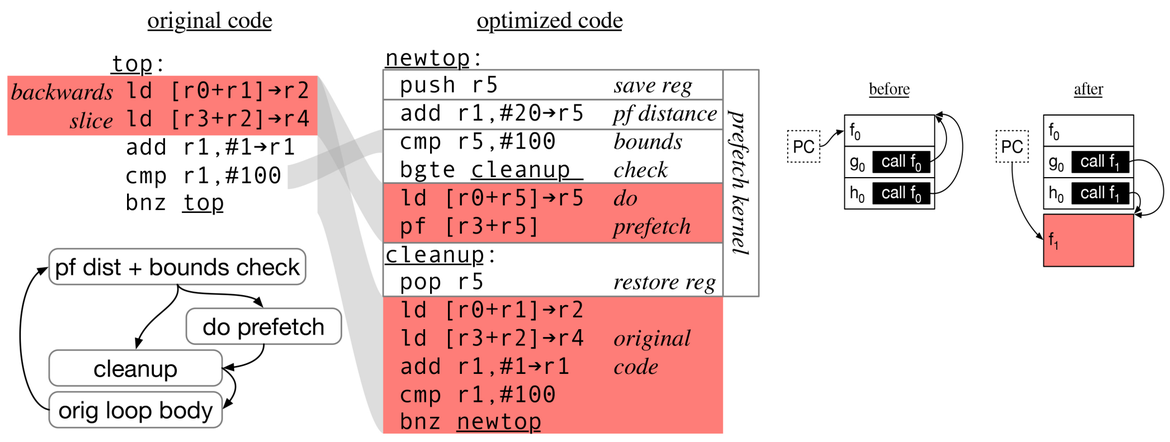
demend access	prefetch	description
`a[j]`	`a[j+d]`	direct access using inner loop induction var
`a[f(b[j])]`	`a[f(b[j+d])]`	indirect access using inner loop induction var
`a[f(b[i])+j]`	`a[f(b[i+d])+j]`	indirect access using inner and outer loop induction var

d: prefetch distance, $f()$:data dependency chain from the load of b[ ] to the demand load that needs to be prefetched

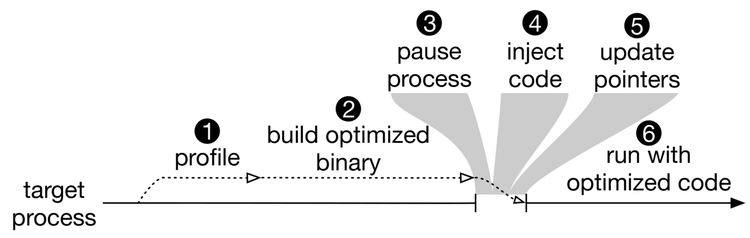
- **backwards slicing**

  to prefetch for indirect memory accesses, RPG2 needs to compute the address for prefetching. The detection algorithm computes the backward slice starting at the demand load that causes the most LLC misses in the function.

- **code generation**



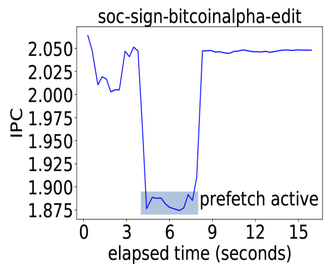
Phase 3 - Runtime Code Insertion



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.

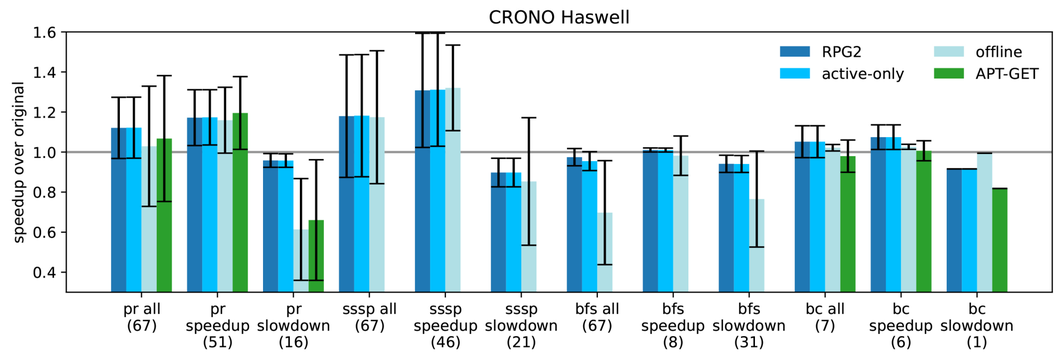
- **stage 1**: decide the searching direction
- **stage 2**: double the jump size to compute the new prefetch distance in the chosen direction. decide the upper and lower bound of the interval for binary search
- **stage 3**: binary search within this interval to identify a local optimum

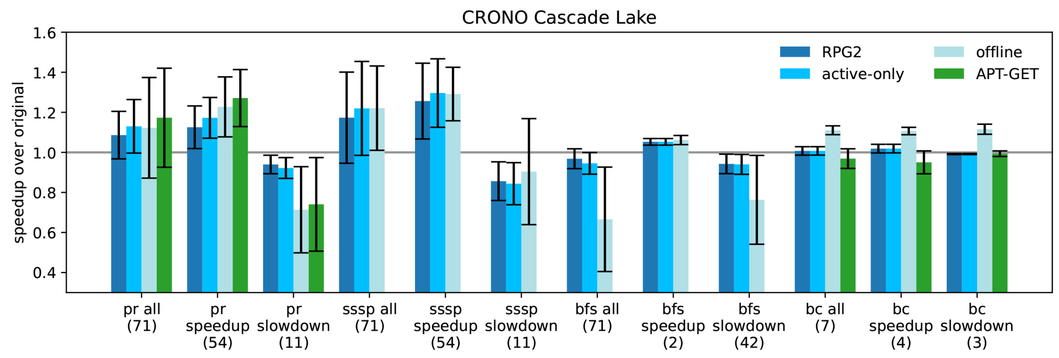


Also support rolling back to the original code if the performance measurement result shows that prefetch is harmful to performance.

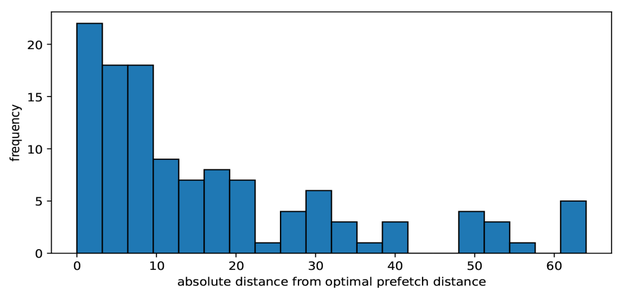
EVALUATION

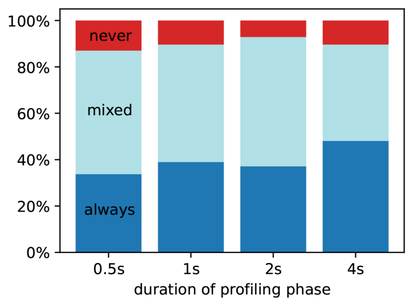
Performance





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	26.4
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	1.4	1.2	1.1	1.1
# pd edit	11.1	11.3	11.5	12.7	11.4	8.8	12.0