Beating Hardware Prefetching in Software

A Case for Resource Efficient Prefetching (REP) in Multicores

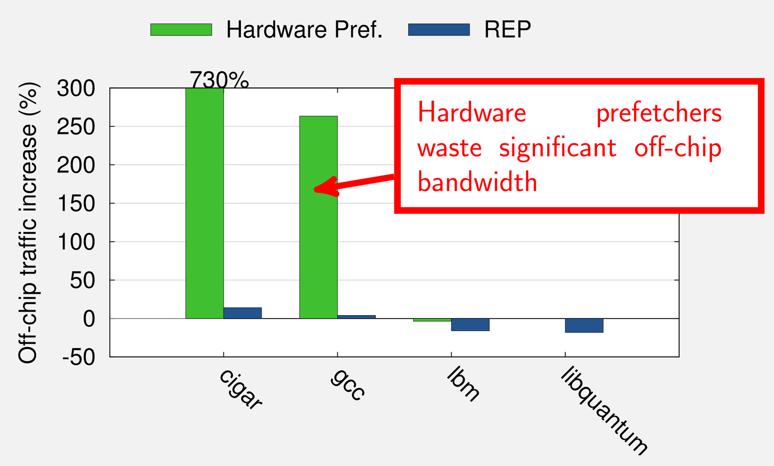
Muneeb Khan, Andreas Sandberg and Erik Hagersten

UPMARC – Uppsala Programming for Multicore Architectures Research Center

UPPSALA UNIVERSITET

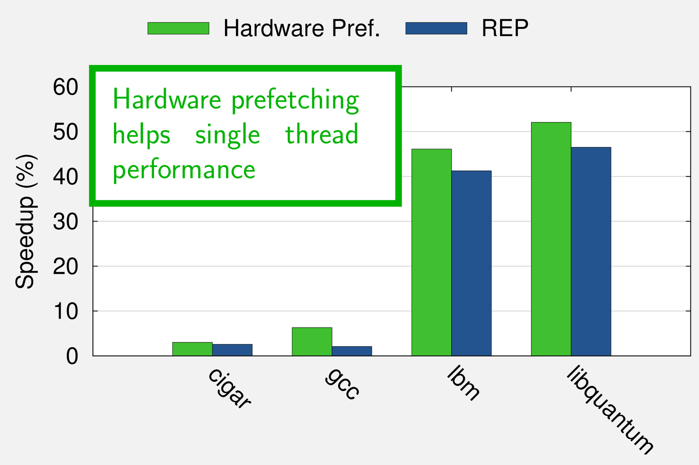
1- Insight: Hardware prefetchers require more shared resources for improving performance

Off-chip Data-Volume Increase



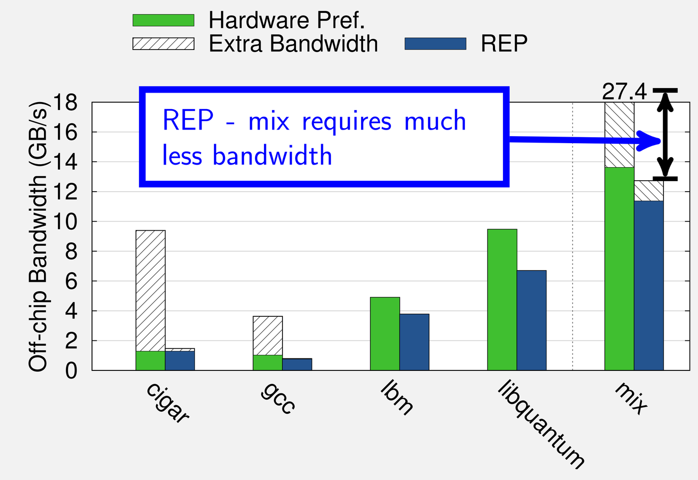
Hardware prefetchers waste considerable off-chip bandwidth by fetching useless data from the DRAM. This happens due to speculative prefetching, which our proposed scheme avoids.

Single-Threaded Performance



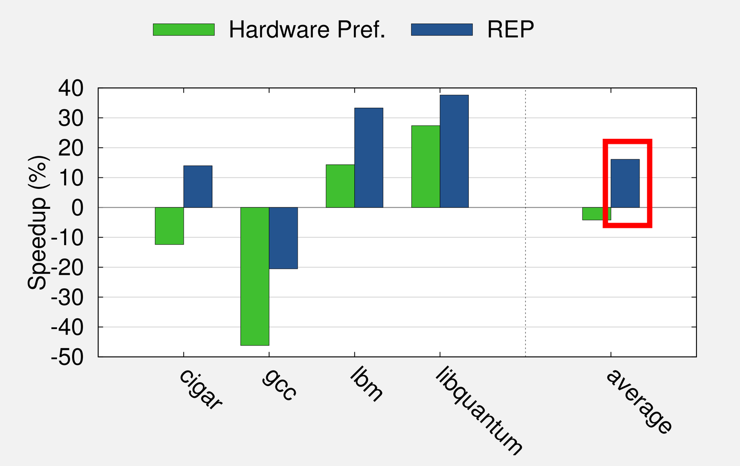
When benchmarks are run in isolation hardware prefetchers perform slightly better than our simple software prefetching method (REP) by aggressively utilizing the off-chip bandwidth (next figure).

Off-chip Bandwidth Requirement



When run together as a mixed workload (*mix*), the hardware prefetchers saturate the off-chip bandwidth much earlier than the ideally required. REP requires and consumes less bandwidth.

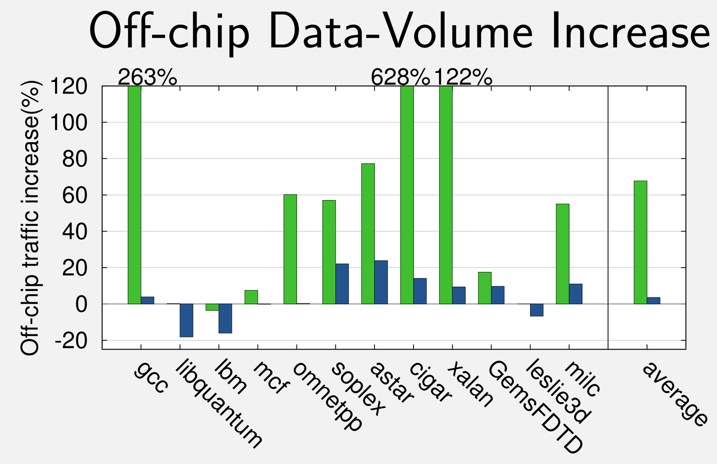
2- Mixed Workload Throughput



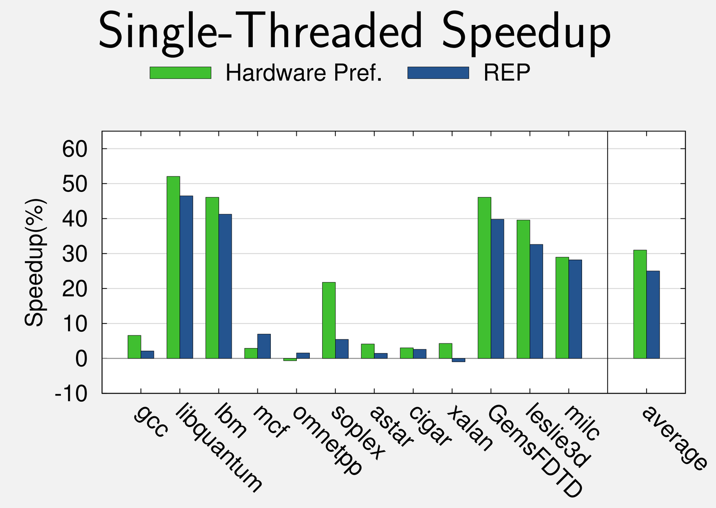
REP's lower bandwidth requirement translates to (23%) performance advantage over hardware prefetching. Hardware prefetching degrades processor throughput by 4% when running the four applications in parallel.

3- Single-Threaded Performance

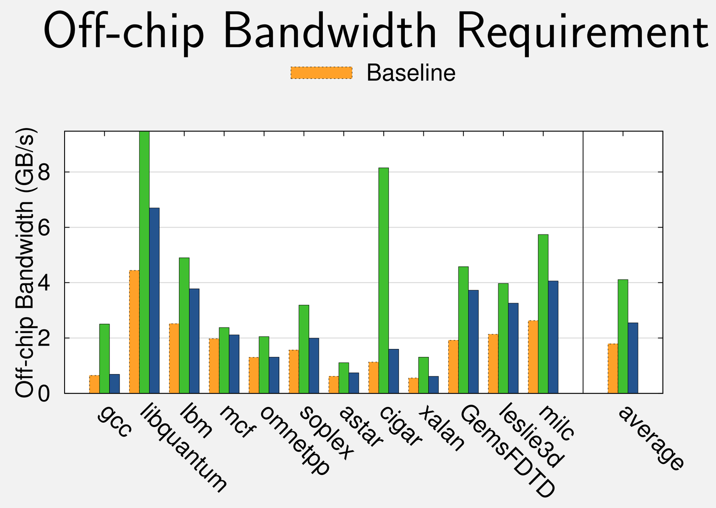
Off-chip Data-Volume Increase



Single-Threaded Speedup



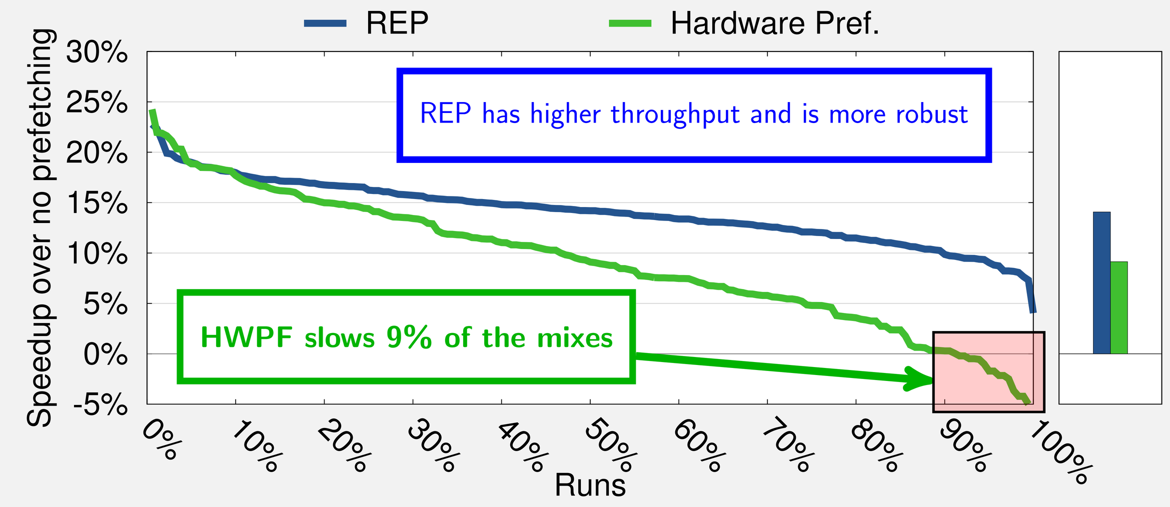
Off-chip Bandwidth Requirement



- REP performance trails slightly behind hardware prefetching (within 5%)
- However, REP lowers off-chip traffic significantly, >60%on average
- Great potential for mixed workloads

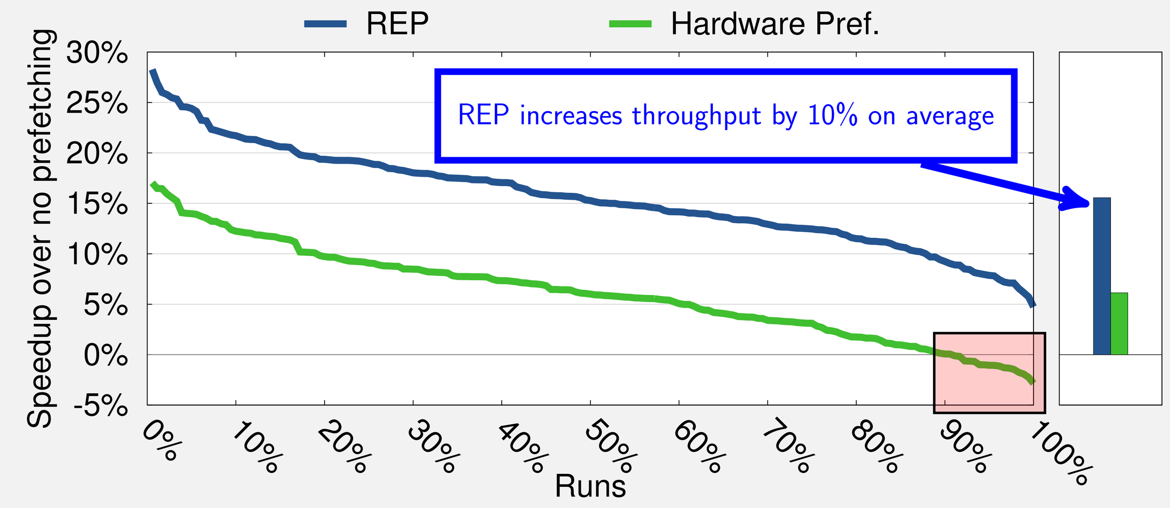
4- Mixed Workloads: Robust Prefetching Method

180 Mixed Workloads Throughput Performance (Intel)



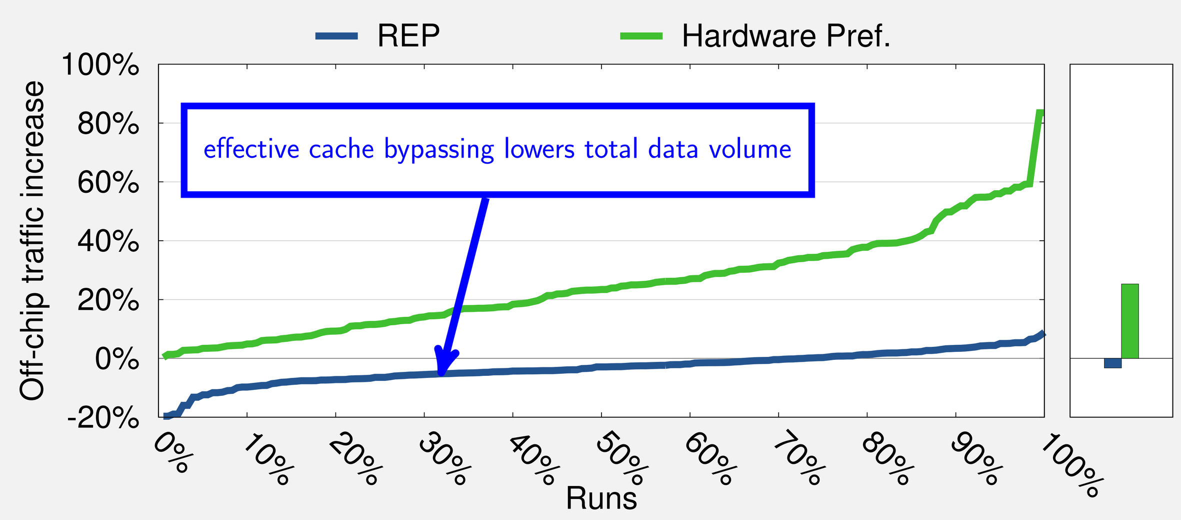
Our scheme's performance is more robust than hardware prefetching across the 180 mixed workloads. REP performs 5% better on average. Cache bypassing helps lower data volume on average by 3% over the baseline.

180 Mixed Workloads Throughput Performance (AMD)



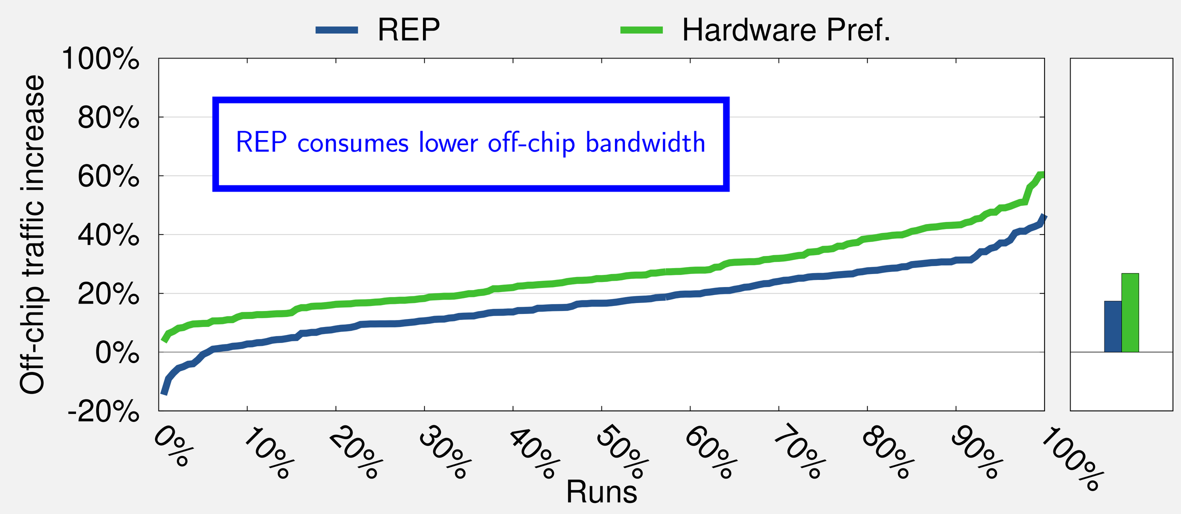
REP performs 10% better on average. On AMD, REP improved throughput performance across all mixed workloads, performing strictly better than hardware prefetching. REP consistently maintains less DRAM traffic.

Increase in Total Data Volume



Intel i7-2600K (Sandybridge)

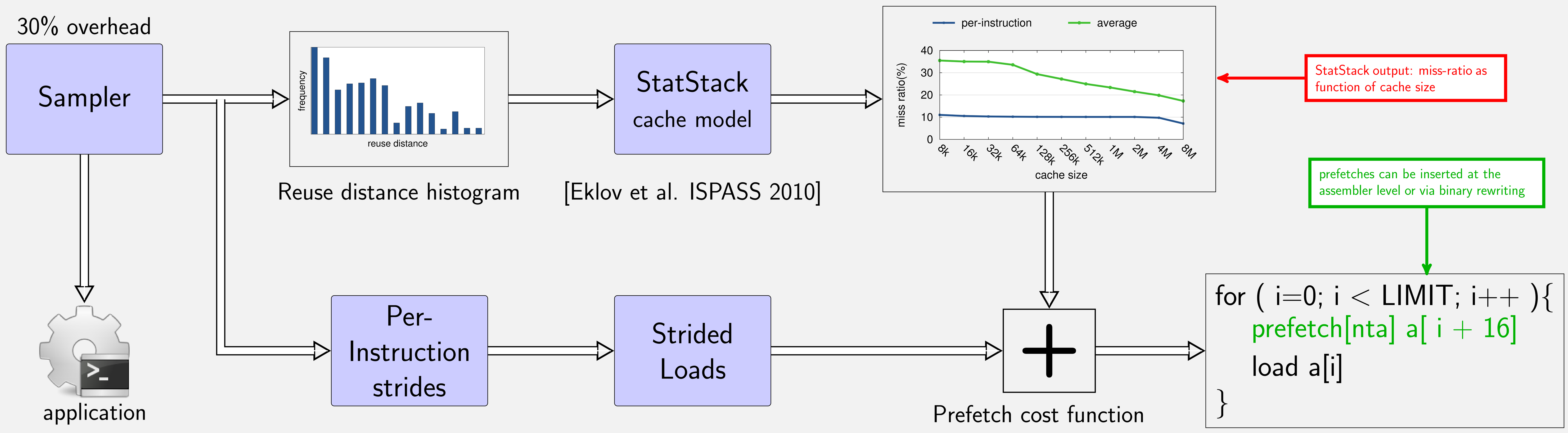
Increase in Total Data Volume



AMD Phenom II X4 920

The benchmarks in Section 4 were selected to create 180 workload mixes. Each mix contains 4 different randomly chosen benchmarks that were run in parallel on the 4 cores of a Intel Sandybridge and AMD Phenom II processors. Different mixes stress shared resources differently, and help us explore how REP benefits performance under varying conditions. The graphs above compare REP's performance against hardware prefetching.

5- REP Framework: A fast software prefetching framework



6- Conclusions

This work investigates how a resource-efficient prefetching (REP) method can help improve throughput performance in multicores when shared resources are constrained. We propose an efficient method that 1) accurately prefetches the required data, 2) avoids (useless) speculative prefetching, and 3) employs cache bypassing to retain useful data in the higher level caches. In contrast to hardware prefetchers, REP is designed to maintain minimal off-chip traffic, and as a result avoids LLC pollution and lowers off-chip bandwidth demand.

This benefits throughput performance in multicores when several applications co-execute and share resources. Compared to state-of-the-art hardware prefetching on two high-performance commodity processors, REP performs up to 10% better on average. Our work highlights the importance of shared-resource friendly prefetching for optimizing multicore performance.