GL-Cache: Group-level Learning for Efficient and High-performance Caching

Juncheng Yang (Carnegie Mellon), Ziming Mao (Yale University), Yao Yue (Pelikan Foundation), K. V. Rashmi (Carnegie Mellon)

Introduction

• Cache is widely deployed to support the modern Internet
• Two metrics are important for a cache: efficiency (measured by hit ratio) and throughput performance
• Many recent works improve the efficiency of caches using machine learning

Group-level Learning

• Amortizes the cost of learning across multiple objects
• Can accumulate more information for learning since most objects have very few requests

Background: Learned Caches

• We categorize the existing learned caches into 3 types:
  1. Learning from simple experts, e.g., LeCaR and-CACHEUS
  2. Learning from probability distribution, e.g., LHD
  3. Object-level learning

Design of GL-Cache

• Many challenges:
  › How does GL-Cache group objects?
  › What and How does GL-Cache learn?
  › How does GL-Cache evict?

INSERTION-TIME-BASED GROUPING

A NEW UTILITY FUNCTION

• Properties desired:
  › Larger object → smaller utility
  › Sooner-to-be-accessed → larger utility
  › Group size one → Belady’s MIN (weighted by size)
• Model: gradient-boosting trees
• Objective: regression
• Eviction in GL-Cache
  1. Let’s make this the same format as “learning in GL-Cache”
  2. remove this “a utility…”

A UTILITY FUNCTION TO MEASURE

Evaluation

• Efficiency
  › GL-Cache-E is slightly better than state-of-the-art algorithms
  › GL-Cache-T is close to LRB

• Throughput
  › GL-Cache-E is faster than all state-of-the-art algorithms
  › GL-Cache-T is significantly faster

Summary

• Group-level learning
  1. Amortizes the overhead of learning, and
  2. Accumulates more information for learning