Towards using Cached Data Mining for Large Scale Recommender Systems

Swapneel Sheth, Gail Kaiser
Department of Computer Science, Columbia University
New York, NY 10027
{swapneel, kaiser}@cs.columbia.edu

Introduction

- Recommender systems have become increasingly commonplace - Pandora, Amazon, Facebook
- Most of the research has focused on aspects such as algorithms [10, 11] and social network implications [12, 13]
- Very little research that has explored the use of caches and cached data mining to improve the performance of recommender systems

Introduction (2)

- As recommender systems become popular, its user base will grow
- Two important issues will need to be dealt with
  - How to generate recommendations efficiently from a large set of data
  - How to provide these recommendations efficiently to a diverse set of users

Introduction (3)

- We describe how we use cached data mining to answer users' queries and provide recommendations in an efficient way
- We describe an empirical study highlighting their benefits and improvements to response time and throughput for recommendations

Related Work

- There is very little in the published literature discussing caches for recommendation systems
- We found exactly one paper - Qasim et al. [21]
- They propose a general solutions using Active Caches
- Active Caches can answer neighborhood queries to a given query

Related Work (2)

- However, this may not work well in general with a diverse user base that requires different kinds of recommendations
- Due to overheads of caching, Active Caches might perform worse than having no cache
- Unlike Active Caches, genSpace uses a Prefetch Cache so all recommendations (and not just neighborhood ones) can be answered by the cache

Background & Motivation

- We are exploring new ways for researchers in computational biology and bioinformatics to collaborate by sharing data and knowledge
- Our approach is based on social networking metaphors for collaborative work
- Our implementation is a system called genSpace [14]
- Plugin for geWorkbench [15], an open-source Java-based system for integrated genomics targeted toward biomedical researchers

Background & Motivation (2)

- geWorkbench includes more than 50 tools for genomics data analysis and visualizations
- Can be very daunting for users who don't know which tools to use, the order of using the tools, etc.
- genSpace provides recommendations such as the most frequently occurring workflows including a given tool or starting with the sequence of tools the user has already executed

Background & Motivation (3)

- We log users' activities as they use geWorkbench
- These logs are periodically sent to our central server where data mining and collaborative filtering techniques are used to generate recommendations
- Currently we have about 150 distinct users and 10000 rows of data

Recommendations in genSpace

- Static Recommendations
  - Do not depend on the current activity of the user
  - Typically follows a “pull” model
  - Examples - Top Tools, Top Workflows
- Dynamic Recommendations
  - Does depend on the current activity of the user
  - Typically follows a “push” model
  - Examples - Best Analysis Tool to run next based on the what the user has done so far

genSpace Caching

- Server-Side Cache that supports Static and Dynamic Recommendations
- Prefetch Cache that prefetches all types of recommendations supported
- Not a traditional cache - every recommendation needed will be present in the cache
- We do not need to worry about cache misses as, by definition, hit rate and recall is 100%

genSpace Caching (2)

- Cache generated when the server starts up using SQL queries and stored procedures
- Periodically re-generated as needed - currently, every day
- If we did not have the cache, we would have to re-run the query every time on demand as requests come in for recommendations from users

genSpace Caching (3)

- We use an exponential time-decay formula [19] to address the problem of concept drift [18] to weigh recent user data more heavily
- First, static recommendations are computed and stored
- For tool specific information, we build a hash-based index to represent information such as: workflows including this tool, number of times this tool has been used, etc.
- Finally, a tree-based index of popular workflows is built
- These three parts comprise the genSpace Caching system and are used to provide recommendations

genSpace Cache Limitations

- Due to structure of the cache, it can only support the currently existing types of recommendations in genSpace
- If we want to support additional types of recommendations, the cache would have to be augmented with the appropriate information

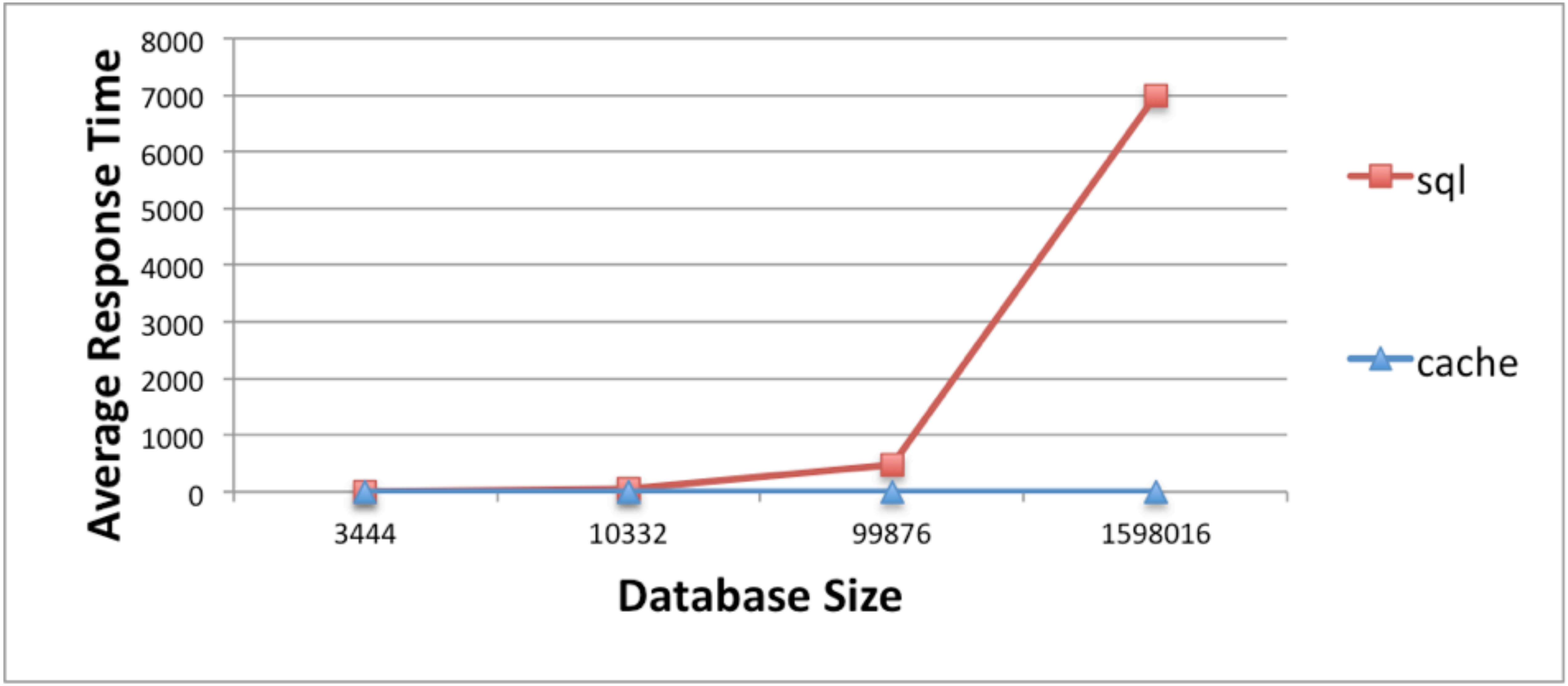
Empirical Study

- We varied the size of the database - 3500, 10000, 100000, 1 million
- We simulated 1000 concurrent users requesting recommendations
- We compared these results to the results obtained if we did not have a cache and used SQL queries every time for generating recommendations

Empirical Study (2)

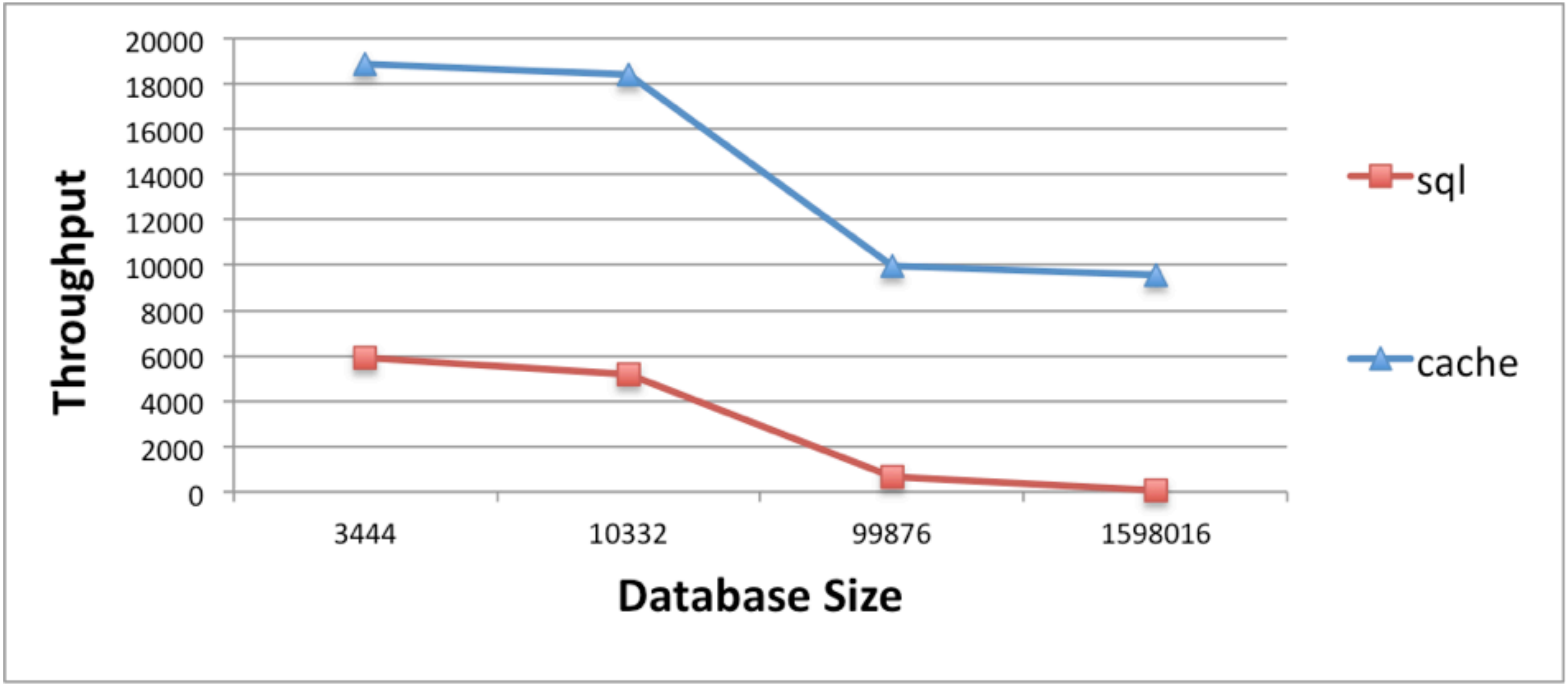
- We used Apache JMeter [20] for load testing our server and measuring performance
- genSpace server and cache is implemented in Java
- Our server and client machines are common Windows XP machines (no non-essential system processes running; >2GB of surplus RAM)

Empirical Study (3)



"Get Most Popular Workflow Heads"

Empirical Study (4)



“Get Most Popular Tools”

Conclusion

- We have described how we use Prefetch Caching in our genSpace recommender system
- We have described the structure of our cache
- Our empirical study shows the advantages of using our cache, which results in improvements to throughput and response time
- We believe such caches will prove very beneficial to recommender systems particularly as the system needs to support a diverse and large user base

Acknowledgments

- Aris Floratos, Kiran Keshav, Zhou Ji
- Cheng Niu, Joshua Nankin, Eric Schmidt, Yuan Wang
- The authors are members of the Programming Systems Lab, funded in part by NSF CNS-0905246, CNS-0717544, CNS-0627473 and CNS-0426623, and NIH 1 U54 CA121852-01A1

Towards using Cached Data Mining for Large Scale Recommender Systems

Swapneel Sheth, Gail Kaiser
Department of Computer Science, Columbia University
New York, NY 10027
{swapneel, kaiser}@cs.columbia.edu