FloraBench:

An End-to-End Application Benchmark Suite for Datacenter

Liu Xiaoli
sally.lxl@alibaba-inc.com
Alibaba Economy

Abundant business

Sustained growth

Infrastructure of Alibaba Economy

Larger and larger scale datacenter: 14 Regions, 45 zones, more than one million servers
Co-located workloads with different requirements
More Elaborate Design for High Efficient Datacenter

Post-Moore's Law Period

Era of Data Center HW-SW Co-design

Application codesign hardware & runtime

Profiling & benchmarking

Benchmarking (understanding characteristics of workloads and system)

Strengthen High efficiency architecture Design (computing, storage, network, IDC)
FloraBench: Application Benchmark Suite with Alibaba's Characteristics

Aim to evaluate better system design options and system tuning with Alibaba's Characteristics
FloraBench: For Better System Design

E-commerce
- SearchBench
- MyChain
- AliCloud

Alibaba's Characteristics

Application Benchmark Suite

Test Configs

Auto-flow

Collaborative Innovation

Metric Data
- System Metrics
- Application Metrics

Platform
FloraBench: overview

e-commerce search

eCommerce-SearchBench

MyChain-perf
eCommerceSearchBench

Benchmark with TaoBao workloads and typical eCommerce Search system model

https://github.com/alibaba/eCommerceSearchBench
FloraBench Methodology

Data Collection and Cleaning

Real-world Data Source

System trace
User logs
Data Sample

Auto-modeling

Dataset Modeling
Workload Modeling

Benchmark implementation

Dataset Model
Workload Model

Dataset Generator
Workload Generator
Metrics Spec

System Under Test

Database

1. Load Dataset
2. Driver
3. Run Experiment
4. Performance Metrics
**eCommerceSearchBench Architecture**

**eCommerceSearchBench** Mainly consists of:

1. Data Generator
2. Workload Generator
3. eCommerce Search Engine
4. Metrics
eCommerceSearchBench — Data Generator (1/2)

<table>
<thead>
<tr>
<th>Dataset Model</th>
<th>Data Generator</th>
</tr>
</thead>
</table>

Data model

.user model

.data model

.goods model

.excellent goods

.good goods

.bad goods

<table>
<thead>
<tr>
<th>Table 1. 用户信息Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>字段</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>user_id</td>
</tr>
<tr>
<td>sex</td>
</tr>
<tr>
<td>age</td>
</tr>
<tr>
<td>power</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2. 商品信息Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>字段</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>title</td>
</tr>
<tr>
<td>item_id</td>
</tr>
<tr>
<td>shop_title</td>
</tr>
<tr>
<td>shop_id</td>
</tr>
<tr>
<td>price</td>
</tr>
<tr>
<td>category</td>
</tr>
<tr>
<td>others</td>
</tr>
</tbody>
</table>
**eCommerceSearchBench — Data Generator (2/2)**

Dataset Model

Data Generator

Data generator

- **data gen**
  - user data
  - forward index
  - Inverted Index
  - summary index

index built

- data load

- excellent/good/bad goods

- slicing, filtering the stop-word

result txt

Char-RNN

goods doc

Data Generator is provided by docker image: **aliesearch-benchmark-cli**
eCommerceSearchBench — workload Generator (1/5)
A session is defined in the context of a user and its continuous queries.

1. Timing Model
   - session-based

- 1. Time Series Model (R, forecast)
  - sessions per hour
  - periodic patterns for each day
  - sessions per second
  - uniform distribution

periodic patterns for user inter-session
eCommerceSearchBench — workload Generator (3/5)

1. Timing Model

- **session-based**

A session is defined in the context of a user and its continuous queries.

- **Inter-session Interval**
- **Intra-session Interval**

2. session length distribution

- Server side aggregate

80% of the session length is less than 15

3. query interval model (python Scipy)

- **Independent Identically Distribution**

According to MLE and K-S test, the model is more likely obeys:

- burr distribution
- exponential distribution
A session is defined in the context of a user and its continuous queries.

1. Timing Model

   ![Diagram: Intra-session Interval vs. Inter-session Interval over time]
   - Intra-session Interval
   - Inter-session Interval
   - Time

2. Semantic Model

   ➢ request type

   - Search for first page (80%)
   - Search for second page (20%)

   ➢ query words distribution

   ![Zipf distribution chart]

   Frequency vs. Ranking
The workload is generated hour granularity and described based on sessions and queries:

- total sessions for specified hour: decide by the time of day
- sessions per second: \( \text{total sessions} / 3600 \)
- session length: generated by the CDF of session length
- query interval: generated by the query interval model
- request type: 80% for page 1, 20% for other page
- query words: generated from 1 million sample words

<table>
<thead>
<tr>
<th>time of day</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018031200</td>
<td>0.74992588</td>
</tr>
<tr>
<td>2018031201</td>
<td>0.39099045</td>
</tr>
<tr>
<td>2018031202</td>
<td>0.21682172</td>
</tr>
<tr>
<td>2018031203</td>
<td>0.13952101</td>
</tr>
<tr>
<td>2018031204</td>
<td>0.10864263</td>
</tr>
<tr>
<td>2018031205</td>
<td>0.1101313</td>
</tr>
<tr>
<td>2018031206</td>
<td>0.20561628</td>
</tr>
<tr>
<td>2018031207</td>
<td>0.37391364</td>
</tr>
<tr>
<td>2018031208</td>
<td>0.57469569</td>
</tr>
<tr>
<td>2018031209</td>
<td>0.78527737</td>
</tr>
<tr>
<td>2018031210</td>
<td>0.96596103</td>
</tr>
<tr>
<td>2018031211</td>
<td>0.98083724</td>
</tr>
<tr>
<td><strong>2018031212</strong></td>
<td><strong>1</strong></td>
</tr>
<tr>
<td>2018031213</td>
<td>1.12280701</td>
</tr>
<tr>
<td>2018031214</td>
<td>1.13986167</td>
</tr>
<tr>
<td>2018031215</td>
<td>1.16860428</td>
</tr>
<tr>
<td>2018031216</td>
<td>1.12206576</td>
</tr>
<tr>
<td>2018031217</td>
<td>0.99190554</td>
</tr>
<tr>
<td>2018031218</td>
<td>0.92822212</td>
</tr>
<tr>
<td>2018031219</td>
<td>1.09155613</td>
</tr>
<tr>
<td>2018031220</td>
<td>1.36240543</td>
</tr>
<tr>
<td>2018031221</td>
<td>1.64210058</td>
</tr>
<tr>
<td>2018031222</td>
<td>1.6951944</td>
</tr>
<tr>
<td>2018031223</td>
<td>1.28393165</td>
</tr>
</tbody>
</table>

Workload is driven by jmeter and is provided by docker image: \text{--aliesearch-jmeter-image}
Typical eCommerce Search Engine consists of:

1. SP: “entrance” of search service
2. QP: “Brain” of search service
   - query intention analysis
   - personalized recommendation
3. HA3 searcher: “executor” of the search query
   - searcher: reverted index, three types of goods
   - summary: detail of goods
   - QPS: scheduler
4. Ranker: precise ranking for goods
   - forward index
   - ranking algorithm models
A reference eCommerce Search Engine Model is implemented by 5 docker images:

1. **SP**: “entrance” of search service
   - **aliesearch-search-planner**: based on Java web framework

2. **QP**: “Brain” of search service
   - **aliesearch-query-planner**: based on Java web framework
   - **aliesearch-tf-serving**: for personalized recommendation

3. **HA3 searcher**: “executor” of the search query
   - **aliesearch-ha3**: based on ElasticSearch
     - excellent-goods
     - good-goods
     - bad-goods
     - goods-summary

4. **Ranker**: precise ranking for goods
   - **aliesearch-ranking-service**: based on ElasticSearch
eCommerceSearchBench — Example Run

Run example experiment:
1. Download the repo, build and start all docker containers
2. Login Jmeter docker and Run example by:
   ```
   cd apache-jmeter-5.1.1
   bin/jmeter -n -t search.jmx -l result -e -o report
   ```

Performance Metrics:
1. End-to-end performance metrics (provided by Jmeter)

2. Breakdown of Response Time

---

## Summary Table

<table>
<thead>
<tr>
<th>Requests</th>
<th>#Samples</th>
<th>KO</th>
<th>Error %</th>
<th>Average</th>
<th>Min</th>
<th>Max</th>
<th>99th pct</th>
<th>99th pct</th>
<th>99th pct</th>
<th>Throughput</th>
<th>Network (KB/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>32048</td>
<td>0</td>
<td>0.00%</td>
<td>147.85</td>
<td>15</td>
<td>804</td>
<td>178.00</td>
<td>284.00</td>
<td>426.00</td>
<td>270.33</td>
<td>99920.27</td>
</tr>
<tr>
<td>Request 1</td>
<td>32048</td>
<td>0</td>
<td>0.00%</td>
<td>147.85</td>
<td>15</td>
<td>804</td>
<td>178.00</td>
<td>284.00</td>
<td>426.00</td>
<td>270.33</td>
<td>99920.27</td>
</tr>
</tbody>
</table>

```java

-------- statistics--------

total requests: 51463

latency statistics(ms):

    TotalTimeCost : min= 13, max= 462, avg= 122.97, P90= 161, P95= 166, P99= 218
    SearchPlaner  : min= 30, max= 158, avg= 33.40, P90= 138, P95= 142, P99= 150
    HA3Searcher   : min= 0, max= 85, avg= 7.32, P90= 9, P95= 10, P99= 15
    Ranking       : min= 1, max= 176, avg= 98.29, P90= 138, P95= 142, P99= 150
```
eCommerceSearchBench — Future Improvements

1. Enrich the User schema to enhance the personalized recommendation for search

2. Optimize the eCommerceSearchModel to enable:
   - cluster deployment
   - its performance to be compared with the on-line system

3. Update the workload model with time
FloraBench  |  eCommerceSearchBench

https://github.com/alibaba/eCommerceSearchBench

will make sustained efforts to open to work together with researchers, help people in related field get a better understanding of modern datacenter, enable researchers & practitioners doing research with more realistic data, and thus making better design options for industry datacenter.
Alibaba Legal Notices and Disclaimer

Copyright © 2019 by Alibaba Group Holding Limited and its affiliates (“Alibaba”)

All trademarks, logos and brands are property of Alibaba or respective owners.
THANKS