MapReduce-based Similarity Join for Metric Spaces

Yasin N. Silva, Jason M. Reed, Lisa M. Tsosie
Arizona State University

Slides prepared for Cloud-I workshop in VLDB 2012.
Overview

• Motivation
• Algorithm
• Implementation
• Performance Evaluation
• Conclusions and Future Work
Introduction

- Similarity Joins used by many companies
- Internet companies have massive amounts of data
- Many non-distributed approaches to Similarity Join problem
- Few cloud based approaches
Our Contribution

- MRSimJoin Algorithm
- General enough for any data in metric space
- Guidelines to implement in Hadoop
- Evaluation of performance and scalability
- Evaluation of pivot numbers, means of choosing a good number of pivots
Overview

- Motivation
- Algorithm
- Implementation
- Performance Evaluation
- Conclusions and Future Work
MRSimJoin Round

- MRSimJoin iteratively partitions the data
  - If partition is small enough, solve in single node SJ routine
- The process is divided into a sequence of rounds
- The initial round partitions the input data
- Any subsequent round repartitions a previously generated partition
Multiple Rounds

- Each round corresponds to a MapReduce job
- The output of a round includes:
  1. Result links for the small partitions that were processed in a single-node
  2. Intermediate data for partitions that require further partitioning
Partitioning Data

Partitioning a Base Partition

Partitioning a Window-Pair Partition

Base Partitions

Window-pair Partition

Q0

Q1

P0

P1

P0_P1

E

F

C

D

Q0_Q1{1}

Q0_Q1{2}

A

B

A

B

A

B

ε

ε
Partition a Base Set

• Choose pivots (randomly chosen subset of input data)
• Create base partitions around closest pivots

- Create window-pair partitions between partitions
- Each partition is sent to a reduce group

<table>
<thead>
<tr>
<th>High order</th>
<th>Base partitions. Ordered by pivot index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window-pair partitions. Ordered by (min pivot index, max pivot index)</td>
<td></td>
</tr>
<tr>
<td>(P₀, P₁), (id₅, elem₅, P₀)</td>
<td>(P₁, P₀), (id₆, elem₆, P₁)</td>
</tr>
<tr>
<td>(P₁, -1), (id₂, elem₂)</td>
<td>(P₁, -1), (id₄, elem₄)</td>
</tr>
<tr>
<td>(P₁, -1), (id₆, elem₆)</td>
<td></td>
</tr>
<tr>
<td>(P₀, -1), (id₃, elem₃)</td>
<td></td>
</tr>
<tr>
<td>(P₀, -1), (id₅, elem₅)</td>
<td></td>
</tr>
</tbody>
</table>

(a) General order of partitions
(b) Order of partitions with 2 pivots
Choose pivots

Partition data around pivots

Create windows space between base partitions

- The window of a window is aware of previous partitioning
Overview

• Motivation
• Algorithm
• Implementation
• Performance Evaluation
• Conclusions and Future Work
Implementation

• Generic enough to implement in any MR framework

• Hadoop implementation:
  – Distribution of Atomic parameters
    • Uses jobConf
  – Distribution of pivots
    • Uses Distributed Cache
  – Renaming Directories
    • Renaming directories does not move data
Overview

• Motivation
• Algorithm
• Implementation
• Performance Evaluation
• Conclusions and Future Work
Testing Platform

- **Amazon EC2 Cloud**
  - 4 virtual cores (2 EC2 Compute Units each)
  - 15 GB memory
  - 1,690 GB local storage
  - 64 bit platform
- **Hadoop 0.20.2**
  - 64 MB block size
  - 10 nodes (1 master, 9 worker nodes)
- **Memory Threshold**
  - 32 MB
Test Data Information

- **SynthData**
  - Synthetic Data set
  - 16D Data
  - Vector components values are [0-1000]
  - Scale Factor 1 = 5 million records

- **ColorData**
  - Corel Color Moments Dataset
  - 9D Data
  - Vector components values are [-4.8 – 4.4]
  - Scale Factor 1 = 5 million records
Increasing Scale Factor

- **ColorData**
  - SF1-SF5
  - 9D Color Vectors
  - Epsilon 1.5%
  - 1.6x (SF1) – 13.3x (SF4)

- **SynthData**
  - SF1-SF5
  - 8D Vectors
  - Epsilon 1.5%
  - 2.4x (SF1) – 11.4x (SF3)
Increasing Epsilon

- **SynthData**
  - 0.5% - 4.0% epsilon
  - 8D Vectors
  - SF1
  - 1.4x – 3x

- **ColorData**
  - 0.5% - 2.5% Epsilon
  - 9D Color Vectors
  - SF1
  - ~60% of MRThetaJoin
Increasing Dimension

- SynthData
  - 4D-10D
  - Epsilon 1.5%
  - SF1
  - MRThetaJoin is 20 - 200% higher

![Graph showing execution time vs number of dimensions for SynthData, Eps:1.5%, SF:1 with MRSimJoin and MRThetaJoin]
Increasing Pivot Number

- **SynthData**
  - 25 - 300 pivots
  - 8D vector
  - Epsilon 1.5%

![Graph showing execution time and number of rounds for SynthData, 8D, SF:1, Eps:1.5%](image-url)
Increasing Node Number & SF

- ColorData
  - (SF1, nodes) – (SF5,10 nodes)
  - 9D Color Vector
  - Epsilon 1.5%
  - MRTetaJoin
    - 9.8x between (SF1, 2) & (SF5, 10)
- MRSimJoin
  - 2.8x between (SF1, 2) & (SF5, 10)
Overview

• Motivation
• Algorithm
• Implementation
• Performance Evaluation
• Conclusions and Future Work
Conclusions

• MRSimJoin efficiently solves the distributed Similarity Join problem
• Significantly better than state-of-the-art MapReduce arbitrary join algorithm
• Partitions data till data can be joined in single node
• Any data set that lies in a metric space
• Scalable
• Highly parallel
Future Work

• Other similarity-aware operators
  – kNN Join, kDistance Join, etc
• Indexing techniques for implementing Similarity Join operations
• Cloud queries with multiple similarity-based operators
Questions?

![MRSimJoin Interface](image-url)

**Query Parameters**
- **Epsilon**: 0.9
- **Input Directory**: /corel/colorMomentsData/
- **Number of Pivots**: 200
- **Memory Threshold**: 32MB
- **Output Directory**: /dataOut/corel/colorMoments/
- **Number ofReducers**: 25

**Query Output**

<table>
<thead>
<tr>
<th>Figure 1</th>
<th>Figure 2</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMG0010692</td>
<td>IMG0045893</td>
<td>0.06996983</td>
</tr>
<tr>
<td>IMG0013483</td>
<td>IMG0045838</td>
<td>0.06997345</td>
</tr>
<tr>
<td>IMG0013538</td>
<td>IMG0045838</td>
<td>0.07121230</td>
</tr>
<tr>
<td>IMG0015994</td>
<td>IMG0045887</td>
<td>0.07322230</td>
</tr>
<tr>
<td>IMG0017033</td>
<td>IMG0045850</td>
<td>0.07508767</td>
</tr>
<tr>
<td>IMG0017044</td>
<td>IMG0045851</td>
<td>0.08664375</td>
</tr>
<tr>
<td>IMG0018757</td>
<td>IMG0045844</td>
<td>0.08763093</td>
</tr>
<tr>
<td>IMG0019481</td>
<td>IMG0045844</td>
<td>0.08873102</td>
</tr>
<tr>
<td>IMG0019481</td>
<td>IMG0045889</td>
<td>0.08909231</td>
</tr>
</tbody>
</table>

**Job Statistics**
- **Matches Found**: 14,832
- **Total Run Time**: 00:01:33 (HH:MM:SS)
- **Total Number of Rounds**: 1
- **Total Mappers**: 6
- **Average Mappers/ Round**: 6
- **Total Reducer Groups**: 459
- **Average Reducer/ Round**: 459