Distributed Cube Materialization on Holistic Measures

Arnab Nandi¹, Cong Yu², Philip Bohannon³, Raghu Ramakrishnan³

¹University of Michigan, Ann Arbor
²Google Research, New York
³Yahoo! Research, California
Motivation

• Search engines accumulate a lot of query logs
• Each query event has a lot of information
  – Location (inferred from IP)
  – Demographics (inferred from past behavior)
  – The query itself (mapped to semantic categories)
• Goal: identify interesting facts across all combinations of dimensions
  – e.g. “Women in their 40s from Michigan search a lot about knitting.”
• Typical cube analysis, except …
Challenge of Scale

• Over any extended period of time, there can be a lot of such events (i.e., *tuples*)
  – Traditional cube analysis techniques simply can’t handle this scale.

• Don’t we have MapReduce and problem solved?

• Yes and no:
  – Yes: distributed computation, such as MapReduce, is necessary to handle data at this scale.
  – No: applying MapReduce techniques to cube analysis is not straight-forward.
Outline

• Introduction
• Preliminaries
• Naïve Approach and Challenges
• MR-Cube Approach
• Experimental Evaluation
• Conclusion
Preliminaries: MapReduce

Map: abc, abb, bcd

Shuffle:
- a: 1
- b: 1
- c: 1
- b: 2

Reduce:
- a: 1 → a: {1,1} → a: 2
- b: 1 → b: {1,2,1} → b: 4
- c: 1 → c: {1,1} → c: 2
- d: 1 → d: {1} → d: 1

ICDE, April 2011
Preliminaries: Data Cube

\[
\text{SELECT COUNT(uid)} \\
\text{FROM ‘user.table’ as (uid, color, gender)} \\
\text{CUBE ON color, gender}
\]

<table>
<thead>
<tr>
<th></th>
<th>Men</th>
<th>Women</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>3</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Green</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Total</td>
<td>4</td>
<td>3</td>
<td>7</td>
</tr>
</tbody>
</table>
Preliminaries: Data Cube

<table>
<thead>
<tr>
<th></th>
<th>Men</th>
<th>Women</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>3</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Green</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Total</td>
<td>4</td>
<td>3</td>
<td>7</td>
</tr>
</tbody>
</table>
Preliminaries: Data Cube

<table>
<thead>
<tr>
<th></th>
<th>Men</th>
<th>Women</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Green</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Men</th>
<th>Women</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Green</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

new dimension = fruit

<table>
<thead>
<tr>
<th></th>
<th>Men</th>
<th>Women</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>3</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Green</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Total</td>
<td>4</td>
<td>3</td>
<td>7</td>
</tr>
</tbody>
</table>
Preliminaries: Data Cube
Key to Cube Materialization: Measure

• Algebraic measures
  – Value for parent can be computed easily from values of children.
    – \( \text{COUNT}(A \cup B) = \text{SUM}(\text{COUNT}(A), \text{COUNT}(B)) \)

• Non-Algebraic (i.e, holistic)
  – Value for parent can not be computed from children.
    – \( \text{COUNT}(\text{DISTINCT}(A \cup B)) \)

• Formal definitions in paper.

• Holistic measures pose significant challenges for distribution!
Outline

• Introduction
• Preliminaries
• Naïve Approach and Challenges
• MR-Cube Approach
• Experimental Evaluation
• Conclusion
Example Task

- Given the location and topic hierarchies, compute *volume* and *reach* (number of distinct users) of all cube groups whose reach is at least 5.

<table>
<thead>
<tr>
<th>User</th>
<th>Query (Event)</th>
<th>Time</th>
<th>Topic  …</th>
<th>Location …</th>
</tr>
</thead>
<tbody>
<tr>
<td>u1</td>
<td>Nikon d40</td>
<td>20090601</td>
<td>SLR</td>
<td>Ann Arbor</td>
</tr>
<tr>
<td>u2</td>
<td>Canon sd100</td>
<td>20090601</td>
<td>Point &amp; Shoot</td>
<td>Detroit</td>
</tr>
<tr>
<td>u1</td>
<td>Apple iPhone</td>
<td>20090602</td>
<td>Smartphone</td>
<td>Ann Arbor</td>
</tr>
<tr>
<td>u3</td>
<td>Nikon d40</td>
<td>20090604</td>
<td>SLR</td>
<td>New York</td>
</tr>
<tr>
<td>u4</td>
<td>Nokia 1100</td>
<td>20090605</td>
<td>Basic Phone</td>
<td>Sunnyvale</td>
</tr>
</tbody>
</table>
Cube Lattice (dimensions = location + topic)
Algorithm 1 Naive Algorithm

MAP(e)
1 # e is a tuple in the data
2 let C be the Cube Lattice
3 for each Region R in C
4 do k = R(e);
5 Emit k ⇒ e

REDUCE/COMBINE(k, \{e_1, e_2, ...\})
1 let M be the measure function
2 Emit k ⇒ M(\{e_1, e_2, ...\})
Naïve Computation Example

<table>
<thead>
<tr>
<th>User</th>
<th>Query (Event)</th>
<th>Time</th>
<th>Topic</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>u1</td>
<td>Nikon d40</td>
<td>20090601</td>
<td>SLR</td>
<td>Ann Arbor</td>
</tr>
<tr>
<td>u2</td>
<td>Canon sd100</td>
<td>20090601</td>
<td>Point &amp; Shoot</td>
<td>Detroit</td>
</tr>
<tr>
<td>u1</td>
<td>Apple iPhone</td>
<td>20090602</td>
<td>Smartphone</td>
<td>Ann Arbor</td>
</tr>
<tr>
<td>u3</td>
<td>Nikon d40</td>
<td>20090604</td>
<td>SLR</td>
<td>New York</td>
</tr>
<tr>
<td>u4</td>
<td>Nokia 1100</td>
<td>20090605</td>
<td>Basic Phone</td>
<td>Sunnyvale</td>
</tr>
</tbody>
</table>

Map phase (16 records!)

Reduce phase

<Ann Arbor, SLR> ➔ u1
...<Ann Arbor, SLR> ➔ {u1}

<Michigan, Camera> ➔ u1
...<Michigan, Camera> ➔ {u1, u2}

<*, *> ➔ u1
...<*, *> ➔ {u1, u2, u3, u4, u5}
Challenges

• **Map phase**: too many intermediate keys
  
  – Number of keys = |C| x |D|, where |C| is the number of regions in the lattice, and |D| is the input size.
  
  – As the number of dimension increases, this can cause problems for the map/shuffle phase.

• **Reduce phase**: extremely large reduce groups

  – Some reduce shards can be very large (due to the a few keys with too many values) and cause the reducer dealing with those groups to run out of memory.

  – In fact, one group is guaranteed to have this problem: 
    
    `<*, *>`
Outline

• Introduction
• Preliminaries
• Naïve Approach and Challenges
• MR-Cube Approach
• Experimental Evaluation
• Conclusion
Handling Reducer Failure

• Inspired by a common task
  – Compute the unique number of users in a query log
  – The usual practice involves two steps: 1) MapReduce to get the set of unique user IDs; 2) calculate the size of result to produce the final answer

• Holistic measure computation usually can not be distributed, but
  – What if the input is split into disjoint sets with regard to the attribute of the measure (in the above example, user ID)?
Partially Algebraic Measure

• For an aggregate measure $M$, if there exists an attribute $a$, such that, given a cube group of tuples $G$, we can divide $G$ into mutually exclusive partitions, ${G_i \mid i = 1...k}$ with the additional conditions that

$$G_i.a \land G_j.a = \text{empty}, \text{ for any (i, j) pair, and}$$

and we can find functions $F$ and $H$, such that

$$M(G) = F(H(G_1), \ldots, H(G_k))$$

where $H$ returns an $n$-tuple and $n$ is constant regardless of the size of the individual partition.

• We call $a$ the algebraic attribute of $M$, and $M$ a partially algebraic measure.
## Example

<table>
<thead>
<tr>
<th>User</th>
<th>Query (Event)</th>
<th>Time</th>
<th>Topic …</th>
<th>Location …</th>
</tr>
</thead>
<tbody>
<tr>
<td>u1</td>
<td>Nikon d40</td>
<td>20090601</td>
<td>SLR</td>
<td>Ann Arbor</td>
</tr>
<tr>
<td>u2</td>
<td>Canon sd100</td>
<td>20090601</td>
<td>Point &amp; Shoot</td>
<td>Detroit</td>
</tr>
<tr>
<td>u1</td>
<td>Apple iPhone</td>
<td>20090602</td>
<td>Smartphone</td>
<td>Ann Arbor</td>
</tr>
<tr>
<td>u3</td>
<td>Nikon d40</td>
<td>20090604</td>
<td>SLR</td>
<td>New York</td>
</tr>
<tr>
<td>u4</td>
<td>Nokia 1100</td>
<td>20090605</td>
<td>Basic Phone</td>
<td>Sunnyvale</td>
</tr>
</tbody>
</table>

\[
\langle *, *, \text{uid} \rangle \rightarrow \{u1\} \\
\langle *, *, \text{uid} \rangle \rightarrow \{u2\} \\
\ldots \\
\langle *, *, \text{uid} \rangle \rightarrow \{u5\} \\
\]

**Reduce phase**

**no more large groups!**
Intuitively, the notion of partially algebraicness is the same as the old notion, with the additional constraint that the partition of the data must be done in a certain way (i.e., along the algebraic attribute).

Some intuitive guidelines for detecting whether a holistic measure is partially algebraic:

- An aggregation $A$ on the algebraic attribute
- A fully algebraic measure, $M$, such that
- The original measure can be computed as $M(A(D))$
- E.g., reach = count(groupby-uid(D))
Detecting Large Cube Groups based on Sampling

• We run a small MapReduce job over a sample of the data using the naïve algorithm
  – Goal is to detect any cube group that might be reducer-unfriendly

• With a sample size of 2M tuples, we can obtain accurate enough estimation for 20B tuples
  – Proof based on Chernoff Bounds, see paper for details
Leverage Partially Algebraic Measures, Naïvely

producing even more intermediate keys!

reducer-friendly

reducer-unfriendly
Value Partition (with partition factors)

- Reducer-friendly:
  - <city, subcat>
  - <state, subcat>
  - <city, cat>
  - <country, subcat>
  - <state, cat>
  - <city, topic>
  - <*, subcat>
  - <country, cat>
  - <state, topic>
  - <city, *>
  - <*, cat, uid%2>
  - <country, topic>
  - <state, *>
  - <*, topic, uid%2>
  - <country, *, uid%10>
  - <*, *, uid%10>

- Reducer-unfriendly:
  - <*, *, uid%10>
Handling Too Many Intermediate Keys

batch areas (b1…5)

reducer-friendly

< *, * >
< *,uid%10 >
< country, * >
< state, * >
< *, topic, uid%2 >
< *,cat,uid%2 >
< *,subcat >
< *,cat >
< country, *,uid%10 >
< country,topic >
< country,cat >
< state,topic >
< state,cat >
< country,subcat >
< state,subcat >
< city,subcat >
< city,cat >
< city,topic >
< city,* >
< state,subcat >
< state,cat >
< state,topic >
< state,* >
< country,*,uid%10 >

reducer-unfriendly

b1
b2
b3
b4
b5
Batch Area: A Constraint Satisfaction Problem

• We want to put parent/child groups together as much as possible to facilitate pruning based on monotonic measures

• Group cube regions into areas such that
  – A region that is reducer-friendly must belong to a batch area that contains at least one of its parents
  – No two regions whose parents are reducer-unfriendly can belong to the same area
  – The maximum difference in the number of regions between any pair of areas is 2, a heuristic to balance the load

• If there are more than one batch area configurations, we have a cost model to select the one with the lowest cost (see paper for details).
MR-Cube Algorithm

MR-CUBE(Cube Lattice C, Dataset D, Measure M)
1   \( D_{sample} = \text{SAMPLE}(D) \)
2   RegionSizes R = \text{ESTIMATE-MAPREDUCE}(D_{sample}, C)
3   \( C_{\alpha} = \text{ANNOTATE}(R, C) \) # value part. & batching
4   \textbf{while} (D)
5   \textbf{do} R ← R ∪ MR-CUBE-MAPREDUCE\( (C_{\alpha}, M, D) \)
6   \( D \leftarrow D' \) # retry failed groups \( D' \) from MR-Cube-Reduce
7   \( C_{\alpha} \leftarrow \text{INCREASE-PARTITIONING}(C_{\alpha}) \)
8   Result ← MERGE(R) # post-aggregate value partitions
9   \textbf{return} Result
Outline

• Introduction
• Preliminaries
• Naïve Approach and Challenges
• MR-Cube Approach
• Experimental Evaluation
• Conclusion
Experimental Setup

• Yahoo! Hadoop Cluster
  – Hadoop 0.20
  – Each node has 2x4core 2.5GHz Xeon/8GB RAM/4x1TB
  – 4GB per mapper and reducer
  – Python + Hadoop Streaming

• Datasets:
  – A sample of real clicklog: 516M tuples, 3 dims, 6 levels
  – Synthetic clicklog: 1B tuples, 2 dims, 6 levels

• Compare against
  – Naïve
  – MR-BPP and MR-PT, adapted from Ng et. al., You et. al., Sergey et. al.
Increasing Data Size (real, reach)

Dataset: Real
Measure: Reach

Time (s)

Data Size (Number of Events, in Millions)

MR-BPP
MR-PT
MR-Cube
Naïve
Increasing Data Size (synthetic, reach)

Dataset: Example
Measure: Reach

(Dashed lines imply graceful degradation (see text))
Increasing Parallelism

Dataset: Example-50M
Measure: Reach

(Parallelism: 50, 100, 128, 150, 200, 500, 1000 and 1500)
Effects of Hierarchies

- Time (s)
  - Depth of Hierarchies (Hierarchies = 2)
    - 2
    - 3
    - 4
  - Number of Hierarchies (Depth = 2)
    - 2
    - 3
    - 4
  - Hierarchy Depth & # of Flat attrs
    - 8.0
    - 6.2
    - 4.4
    - 2.6
    - 0.8

Measure: Reach
Data: Example-10M
Anecdotal Results

(a) Top 4 frequent days for female users clicking on IMDB URLs, by state

<table>
<thead>
<tr>
<th>NY</th>
<th>RO</th>
<th>ME</th>
<th>WI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday</td>
<td>Sunday</td>
<td>Sunday</td>
<td>Monday</td>
</tr>
<tr>
<td>Tuesday</td>
<td>Friday</td>
<td>Monday</td>
<td>Sunday</td>
</tr>
<tr>
<td>Friday</td>
<td>Monday</td>
<td>Tuesday</td>
<td>Tuesday</td>
</tr>
<tr>
<td>Thursday</td>
<td>Thursday</td>
<td>Friday</td>
<td>Friday</td>
</tr>
</tbody>
</table>

(b) Queries with highest reach on Wikipedia URLs, by Month

<table>
<thead>
<tr>
<th>Jan</th>
<th>Feb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joanna Pacitti</td>
<td>Joanna Pacitti</td>
</tr>
<tr>
<td>Lisa Bonet</td>
<td>Tonya Harding Today</td>
</tr>
<tr>
<td>Martin Luther King Jr</td>
<td>Eliza Dushku</td>
</tr>
<tr>
<td>Kim Kardashian</td>
<td>Rihanna</td>
</tr>
<tr>
<td>Malia Obama</td>
<td>Chris Brown</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>March</th>
<th>April</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crystal Mckellar</td>
<td>Swine Flu</td>
</tr>
<tr>
<td>Natasha Richardson</td>
<td>Twitter</td>
</tr>
<tr>
<td>Watchmen</td>
<td>Keshia Night Pullam</td>
</tr>
<tr>
<td>XBox 360 Ring</td>
<td>Sabrina Lebeauf</td>
</tr>
<tr>
<td>Marcus Jordan</td>
<td>Lady Gaga</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>May</th>
<th>June</th>
</tr>
</thead>
<tbody>
<tr>
<td>Montauk Monster</td>
<td>Hyalinobatrachium pellucidum</td>
</tr>
<tr>
<td>Kris Allen</td>
<td>David Carradine</td>
</tr>
<tr>
<td>Adam Lambert</td>
<td>Dream Interpretations</td>
</tr>
<tr>
<td>Kate Gosselin</td>
<td>Frank Lloyd Wright Houses</td>
</tr>
<tr>
<td>Derecho Storms</td>
<td>Air France Flight 447</td>
</tr>
</tbody>
</table>
Related Works

• Lots of works on single machine cube materialization
  – BUC, etc.

• Parallel cube materialization over small clusters
  – Ng et al, *SIGMOD* 2001
  – None of the above handle holistic measures
Conclusion

• Large scale cubing is important

• MR-Cube brings cubing to MapReduce
  – Solve scale challenges by introducing
    • Value Partitioning for Partially Algebraic Measures
    • Batch Areas Identification
  – For very large datasets, MR-Cube is the only option

• Future works
  – Approximate computation using sketches
  – Mining “interesting” cube groups at large scale