AD-HOC AGGREGATIONS OF RANKED LISTS IN THE PRESENCE OF HIERARCHIES
SIGMOD 2008, VANCOUVER

Nilesh Bansal, Nick Koudas
University of Toronto
Sudipto Guha
University of Pennsylvania
Ranked Lists are Ubiquitous

- Google Trends
  - List of popular search queries posed at Google
- Flickr
  - List of popular tags sorted by their popularity
- Alexa
  - List of top websites based on traffic ranks
- BlogScope
  - List of top keywords in the blogosphere

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008
www.blogscope.net
Ranked Lists can be Temporal

- Google Trends
  - List of top queries for each day with counts
- Flickr
  - List of top tags for each hour with popularity scores
- Alexa
  - List of top websites for each month with visitor counts
- BlogScope
  - List of top keywords for each day with popularity scores
Top-K is Solved

- Find top-k elements from a collection of lists with highest aggregate score
- Threshold Algorithm
  - Well Studied

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008

www.blogscope.net
Given two lists, one for each day, of top keywords for that day in the blogosphere, find top-3 most talked about keywords.

**Initial lists (with scores)**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercedes, 0.98</td>
<td>Mercedes, 0.94</td>
<td>BMW, 0.91</td>
</tr>
<tr>
<td>Nikon 0.91</td>
<td>Nikon, 0.87</td>
<td></td>
</tr>
<tr>
<td>BMW, 0.87</td>
<td>Minolta, 0.75</td>
<td></td>
</tr>
<tr>
<td>Godfather, 0.73</td>
<td>Godfather, 0.61</td>
<td></td>
</tr>
<tr>
<td>Canon, 0.69</td>
<td>Canon, 0.55</td>
<td></td>
</tr>
<tr>
<td>Minolta, 0.55</td>
<td>Sony, 0.50</td>
<td></td>
</tr>
<tr>
<td>Matrix, 0.42</td>
<td>Matrix, 0.22</td>
<td></td>
</tr>
<tr>
<td>Hyundai, 0.38</td>
<td>Hyundai, 0.12</td>
<td></td>
</tr>
<tr>
<td>Sony, 0.18</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**top-3 answer**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercedes, 1.92</td>
<td>Nikon, 1.78</td>
<td>BMW, 1.78</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008  
www.blogscope.net
Top-K is Simple

Given two lists, one for each day, of top keywords for that day in the blogosphere, **find top-3 most talked about keywords.**

<table>
<thead>
<tr>
<th>Initial lists (with scores)</th>
<th>top-3 answer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mercedes, 0.98</strong>&lt;br&gt;<strong>Nikon 0.91</strong>&lt;br&gt;<strong>BMW, 0.87</strong></td>
<td><strong>Mercedes, 1.92</strong>&lt;br&gt;<strong>Nikon, 1.78</strong>&lt;br&gt;<strong>BMW, 1.78</strong></td>
</tr>
<tr>
<td><strong>Godfather, 0.73</strong>&lt;br&gt;<strong>Canon, 0.69</strong>&lt;br&gt;<strong>Minolta, 0.55</strong>&lt;br&gt;<strong>Matrix, 0.42</strong>&lt;br&gt;<strong>Hyundai, 0.38</strong>&lt;br&gt;<strong>Sony, 0.18</strong></td>
<td><strong>Minolta, 0.75</strong>&lt;br&gt;<strong>Godfather, 0.61</strong>&lt;br&gt;<strong>Canon, 0.55</strong>&lt;br&gt;<strong>Sony, 0.50</strong>&lt;br&gt;<strong>Matrix, 0.22</strong>&lt;br&gt;<strong>Hyundai, 0.12</strong></td>
</tr>
</tbody>
</table>
Hierarchies are Natural

- Dmoz, Yahoo Directory
  - Hierarchical categorization of websites
- Product Stores, EBay
  - Hierarchical categorization of products
- Concept Hierarchy
  - Keywords can be mapped to topics

[Diagram showing a hierarchy of cameras: Camera → Nikon → Coolpix, Camera → Canon → PowerShot]
Top-K with Hierarchies is Interesting

Initial lists (with scores)

- Mercedes, 0.98
- Nikon, 0.91
- BMW, 0.87
- Godfather, 0.73
- Canon, 0.69
- Minolta, 0.55
- Matrix, 0.42
- Hyundai, 0.38
- Sony, 0.18

- Mercedes, 0.94
- Nikon, 0.91
- BMW, 0.91
- Minolta, 0.75
- Godfather, 0.61
- Canon, 0.55
- Sony, 0.50
- Matrix, 0.22
- Hyundai, 0.12

Hierarchy Root

- Cars
  - BMW
  - Mercedes
- Movies
- Camera
  - Nikon
  - Minolta
  - Sony
  - Canon

Aggregated lists (with total score)

- Camera, 5.00
- Cars, 4.20
- Movies, 1.98

Lists after using hierarchy

- Cars, 0.98
- Camera, 0.91
- Cars, 0.87
- Movies, 0.73
- Camera, 0.69
- Camera, 0.55
- Movies, 0.42
- Cars, 0.38
- Camera, 0.18

- Cars, 0.94
- Camera, 0.91
- Cars, 0.91
- Camera, 0.87
- Camera, 0.75
- Movies, 0.61
- Camera, 0.55
- Camera, 0.50
- Movies, 0.22
- Cars, 0.12
Top-K with Hierarchies is Important

- Given website traffic rankings from Alexa and Dmoz web directory, find most visited website categories
- Given most used keywords from BlogScope, and a keyword-to-topic hierarchy, find most talked about topics

- Input
  - A collection of ranked lists (with scored items)
    - One list for each time step
  - Hierarchy

- Output: Top-K after ‘mapping’ items in input lists utilizing the hierarchy
Top-K with Hierarchies is Difficult

Initial lists (with scores)

- Mercedes, 0.98
- Nikon, 0.91
- BMW, 0.87
- Godfather, 0.73
- Canon, 0.69
- Minolta, 0.55
- Matrix, 0.42
- Hyundai, 0.38
- Sony, 0.18

- Mercedes, 0.94
- Nikon, 0.87
- BMW, 0.91
- Minolta, 0.75
- Godfather, 0.61
- Canon, 0.55
- Sony, 0.50
- Matrix, 0.22
- Hyundai, 0.12

Aggregated lists (with total score)

- Camera, 5.00
  - Cars, 0.98
  - Camera, 0.91
  - Cars, 0.87
  - Movies, 0.73
  - Camera, 0.69
  - Camera, 0.55
  - Movies, 0.42
  - Cars, 0.38
  - Camera, 0.18

- Cars, 4.20
  - Movies, 1.98

Hierarchy Root

- Cars
  - BMW
  - Mercedes
- Movies
  - Hyundai
  - Godfather
- Camera
  - Nikon
  - Sony
  - Canon

Lists after using hierarchy

- Cars, 0.98
- Camera, 0.91
- Cars, 0.87
- Movies, 0.73
- Camera, 0.69
- Camera, 0.55
- Movies, 0.42
- Cars, 0.38
- Camera, 0.18

- Cars, 0.94
- Camera, 0.91
- Cars, 0.87
- Camera, 0.75
- Movies, 0.61
- Camera, 0.55
- Camera, 0.50
- Movies, 0.22
- Cars, 0.12

www.blogscope.net

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008
Top-K with Hierarchies is Difficult

Initial lists (with scores)

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercedes</td>
<td>0.98</td>
</tr>
<tr>
<td>Nikon</td>
<td>0.91</td>
</tr>
<tr>
<td>BMW</td>
<td>0.87</td>
</tr>
<tr>
<td>Godfather</td>
<td>0.73</td>
</tr>
<tr>
<td>Canon</td>
<td>0.69</td>
</tr>
<tr>
<td>Minolta</td>
<td>0.55</td>
</tr>
<tr>
<td>Matrix</td>
<td>0.42</td>
</tr>
<tr>
<td>Hyundai</td>
<td>0.38</td>
</tr>
<tr>
<td>Sony</td>
<td>0.18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercedes</td>
<td>0.94</td>
</tr>
<tr>
<td>Nikon</td>
<td>0.91</td>
</tr>
<tr>
<td>BMW</td>
<td>0.87</td>
</tr>
<tr>
<td>Godfather</td>
<td>0.75</td>
</tr>
<tr>
<td>Minolta</td>
<td>0.61</td>
</tr>
<tr>
<td>Canon</td>
<td>0.55</td>
</tr>
<tr>
<td>Sony</td>
<td>0.50</td>
</tr>
<tr>
<td>Matrix</td>
<td>0.22</td>
</tr>
<tr>
<td>Hyundai</td>
<td>0.12</td>
</tr>
</tbody>
</table>

Aggregated lists (with total score)

<table>
<thead>
<tr>
<th>Category</th>
<th>Total Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera</td>
<td>0.00</td>
</tr>
<tr>
<td>Cars</td>
<td>1.92</td>
</tr>
<tr>
<td>Movies</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Hierarchy Root

Cars
- BMW
- Mercedes

Movies
- Hyundai
- Godfather

Camera
- Nikon
- Minolta

Lists after using hierarchy

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cars</td>
<td>0.98</td>
</tr>
<tr>
<td>Camera</td>
<td>0.91</td>
</tr>
<tr>
<td>Cars</td>
<td>0.87</td>
</tr>
<tr>
<td>Movies</td>
<td>0.73</td>
</tr>
<tr>
<td>Camera</td>
<td>0.69</td>
</tr>
<tr>
<td>Camera</td>
<td>0.55</td>
</tr>
<tr>
<td>Movies</td>
<td>0.42</td>
</tr>
<tr>
<td>Cars</td>
<td>0.38</td>
</tr>
<tr>
<td>Camera</td>
<td>0.18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cars</td>
<td>0.94</td>
</tr>
<tr>
<td>Camera</td>
<td>0.91</td>
</tr>
<tr>
<td>Cars</td>
<td>0.87</td>
</tr>
<tr>
<td>Movies</td>
<td>0.75</td>
</tr>
<tr>
<td>Camera</td>
<td>0.55</td>
</tr>
<tr>
<td>Movies</td>
<td>0.22</td>
</tr>
<tr>
<td>Cars</td>
<td>0.12</td>
</tr>
</tbody>
</table>

Top-K with Hierarchies is Difficult

Initial lists (with scores)

<table>
<thead>
<tr>
<th>Cars</th>
<th>Movies</th>
<th>Camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercedes, 0.98</td>
<td>BMW, 0.91</td>
<td>Nikon, 0.87</td>
</tr>
<tr>
<td>Nikon, 0.91</td>
<td>BMW, 0.91</td>
<td>Nikon, 0.87</td>
</tr>
<tr>
<td>Godfather, 0.73</td>
<td>Canon, 0.69</td>
<td>Minolta, 0.75</td>
</tr>
<tr>
<td>Minolta, 0.55</td>
<td>Matrix, 0.42</td>
<td>Godfather, 0.61</td>
</tr>
<tr>
<td>Hyundai, 0.38</td>
<td>Sony, 0.18</td>
<td>Canon, 0.55</td>
</tr>
</tbody>
</table>

Hierarchy Root

<table>
<thead>
<tr>
<th>Cars</th>
<th>Movies</th>
<th>Camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMW</td>
<td>Hyundai</td>
<td>Nikon</td>
</tr>
<tr>
<td>Mercedes</td>
<td>Minolta</td>
<td>Sony</td>
</tr>
<tr>
<td>Canon</td>
<td>Matrix</td>
<td>Canon</td>
</tr>
</tbody>
</table>

Aggregated lists (with total score)

<table>
<thead>
<tr>
<th>Cars</th>
<th>Movies</th>
<th>Camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera, 1.78</td>
<td>Cars, 3.70</td>
<td>Movies, 0.00</td>
</tr>
</tbody>
</table>

Lists after using hierarchy

<table>
<thead>
<tr>
<th>Cars</th>
<th>Movies</th>
<th>Camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera, 0.75</td>
<td>Movies, 0.61</td>
<td>Camera, 0.55</td>
</tr>
<tr>
<td>Camera, 0.55</td>
<td>Movies, 0.42</td>
<td>Camera, 0.50</td>
</tr>
<tr>
<td>Cars, 0.22</td>
<td>Movies, 0.18</td>
<td>Camera, 0.12</td>
</tr>
</tbody>
</table>

Top-K with Hierarchies is Difficult

Initial lists (with scores)

<table>
<thead>
<tr>
<th></th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercedes</td>
<td>0.98</td>
</tr>
<tr>
<td>Nikon</td>
<td>0.91</td>
</tr>
<tr>
<td>BMW</td>
<td>0.87</td>
</tr>
<tr>
<td>Godfather</td>
<td>0.73</td>
</tr>
<tr>
<td>Canon</td>
<td>0.69</td>
</tr>
<tr>
<td>Minolta</td>
<td>0.55</td>
</tr>
<tr>
<td>Matrix</td>
<td>0.42</td>
</tr>
<tr>
<td>Hyundai</td>
<td>0.38</td>
</tr>
<tr>
<td>Sony</td>
<td>0.18</td>
</tr>
</tbody>
</table>

Hierarchy Root

- Cars
  - BMW
  - Mercedes
- Movies
- Camera
  - Nikon
  - Minolta
  - Sony
  - Canon
- Godfather
  - Matrix

Aggregated lists (with total score)

<table>
<thead>
<tr>
<th></th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera</td>
<td>2.53</td>
</tr>
<tr>
<td>Cars</td>
<td>3.70</td>
</tr>
<tr>
<td>Movies</td>
<td>1.34</td>
</tr>
</tbody>
</table>

Lists after using hierarchy

<table>
<thead>
<tr>
<th></th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cars</td>
<td>0.98</td>
</tr>
<tr>
<td>Camera</td>
<td>0.91</td>
</tr>
<tr>
<td>Cars</td>
<td>0.87</td>
</tr>
<tr>
<td>Movies</td>
<td>0.73</td>
</tr>
<tr>
<td>Camera</td>
<td>0.69</td>
</tr>
<tr>
<td>Camera</td>
<td>0.55</td>
</tr>
<tr>
<td>Cars</td>
<td>0.42</td>
</tr>
<tr>
<td>Cars</td>
<td>0.38</td>
</tr>
<tr>
<td>Camera</td>
<td>0.18</td>
</tr>
<tr>
<td>Camera</td>
<td>0.55</td>
</tr>
<tr>
<td>Cars</td>
<td>0.50</td>
</tr>
<tr>
<td>Cameras</td>
<td>0.22</td>
</tr>
<tr>
<td>Cars</td>
<td>0.12</td>
</tr>
</tbody>
</table>
Outline

- Motivation
- Formal Problem Setting
- Stopping Conditions
  - Probabilistic Guarantees
  - Approximate
- Pre-computation
- Experiments
Given $N$ lists, $X_1, X_2, \ldots, X_N$; and hierarchy $H$

Let $x_{ij} = j^{th}$ element in the $i^{th}$ list

Hierarchy maps $x_{ij}$ to $y_{ij}$

Hierarchy is used as a many-to-one mapping

Find top-$k$ mapped elements $\{y\}$ with highest aggregate scores
Threshold Algorithm - NRA

3 initial lists

\[ x_{i1} \quad x_{12} \quad x_{13} \quad x_{14} \quad x_{15} \]

\[ x_{21} \quad x_{22} \quad x_{23} \quad x_{24} \quad x_{25} \]

\[ x_{31} \quad x_{32} \quad x_{33} \quad x_{34} \quad x_{35} \]

Temporary Buffer

\((\text{element, seen score, unseen score})\)

\[ x_{ij} \rightarrow^H y \]

\[ \text{seen\_score}(y) = \text{aggregate score of all seen instances of } y \]

\[ \text{unseen\_score}(y) = \text{aggregate score of all unseen instances of } y \]

\[ \text{total\_score}(y) = \text{seen\_score}(y) + \text{unseen\_score}(y) \]

\[ \text{min}-k = \text{seen score of } k^{th} \text{ highest element in the buffer} \]

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008

www.blogscope.net
Threshold Algorithm - NRA

\[
x_{ij} \rightarrow^H y
\]

\[\text{seen\_score}(y) = \text{aggregate score of all seen instances of } y\]
\[\text{unseen\_score}(y) = \text{aggregate score of all unseen instances of } y\]
\[\text{total\_score}(y) = \text{seen\_score}(y) + \text{unseen\_score}(y)\]
\[\text{min-}k = \text{seen score of } k^{th} \text{ highest element in the buffer}\]
Threshold Algorithm - NRA

3 initial lists

| $x_{11}$ | $x_{21}$ | $x_{31}$ |
| $x_{12}$ | $x_{22}$ | $x_{32}$ |
| $x_{13}$ | $x_{23}$ | $x_{33}$ |
| $x_{14}$ | $x_{24}$ | $x_{34}$ |
| $x_{15}$ | $x_{25}$ | $x_{35}$ |

seen

unseen

$seen\_score(y) = \text{aggregate score of all seen instances of } y$

$unseen\_score(y) = \text{aggregate score of all unseen instances of } y$

$total\_score(y) = seen\_score(y) + unseen\_score(y)$

$min-k = \text{seen score of } k^{th} \text{ highest element in the buffer}$

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008

www.blogscope.net
Threshold Algorithm - NRA

seen_score(y) = aggregate score of all seen instances of y
unseen_score(y) = aggregate score of all unseen instances of y
total_score(y) = seen_score(y) + unseen_score(y)
min-k = seen score of k^{th} highest element in the buffer
Outline

- Motivation
- Formal Problem Setting
- **Stopping Conditions**
  - Probabilistic Guarantees
  - Approximate
- Pre-computation
- Experiments
Stopping Condition

- Exact top-k: Terminate when \( \text{min-k} > \text{total}_{\text{score}(y)} \)
  for all \( y \) not in candidate top-k
  - Very conservative, costly

- 1. Probabilistic Guarantee
  - With some confidence, say 95%
  - [TWS04] Theobald et al, Top-k query evaluation with probabilistic guarantees, VLDB 2004

- 2. Approximate top-k
  - Stopping condition with relaxed precision guarantees
Stopping Condition - Probabilistic

- Exact top-$k$ – Stopping Condition
  - $\Pr[y \text{ belongs to exact top}-k] = 0$ for all $y$ not in candidate top-$k$
Stopping Condition - Probabilistic

- Exact top-k – Stopping Condition
  \[ \Pr[y \text{ belongs to exact top-k}] = 0 \text{ for all } y \text{ not in candidate top-k} \]

- Probabilistic Guarantee – Stopping Condition
  \[ \Pr[y \text{ belongs to exact top-k}] < e \]
  - For constant \( e \); confidence \( = 1 - e = 0.95 \)
Sto
g
Condition - Probabilistic

- Exact top-k – Stopping Condition
  \[\Pr[y \text{ belongs to exact top-k}] = 0 \text{ for all } y \text{ not in candidate top-k}\]

- Probabilistic Guarantee – Stopping Condition
  \[\Pr[y \text{ belongs to exact top-k}] < e\]
  - For constant \(e\); confidence = 1 – \(e\) = 0.95
  \[\Pr[\text{unseen_score}(y) < \text{min-k} – \text{seen_score}(y)] < e\]

- min-k and seen_score(y) are known
  - Need to estimate (probability distribution) of unseen_score(y)

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008
www.blogscape.net
Probabilistic TA with Hierarchies

Terminate when $\Pr[unseen\_score(y) < \min-k - seen\_score(y)] < \epsilon$ for all $y$ not in candidate top-$k$

<table>
<thead>
<tr>
<th>3 mapped lists</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_{11}$</td>
</tr>
<tr>
<td>$y_{12}$</td>
</tr>
<tr>
<td>$y_{13}$</td>
</tr>
<tr>
<td>$y_{14}$</td>
</tr>
<tr>
<td>$y_{15}$</td>
</tr>
</tbody>
</table>

- Estimate $unseen\_score(y)$
- Knowledge of max number of times $\{y\}$ can appear in a single list
  - Possible by single scan of the hierarchy
- Knowledge of score distribution in the lists required

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008  
www.blogscope.net
Parametric Distribution

- Assume parametric distribution of scores of elements in initial lists
  - Geometric distribution

- Compute distribution of unseen_score(y)
  - Possible using numerical tools like Matlab
    - High computational overhead, practically not viable
  - Details in the paper
Pre-computed Histogram

- Maintain a histogram for each initial list
  - Distribution of scores

- Compute distribution of $\text{unseen\_score}(y)$
  - Possible, but needs convolutions of multiple distributions
    - High computational overhead, practically not viable
  - Details in the paper

- [TWS2004] did not have hierarchies
  - Number of convolutions required is much less without hierarchies
  - Top-K with hierarchies is difficult
Outline

- Motivation
- Formal Problem Setting
- Stopping Conditions
  - Probabilistic Guarantees
  - Approximate
- Pre-computation
- Experiments

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008

www.blogscope.net
Approximate top-k with Hierarchies

- Relaxed precision guarantees
  - But with 100% confidence, no probabilities

- Guarantee: \( \text{precision} > \pi \)
  - For target precision \( \pi < 1 \)
  - Out of \( k \) elements in result set, at least \( k' \) will be in true top-k, where \( k'/k > \pi \)
    - Precision = \( k'/k = \frac{\text{correct results}}{\text{total results}} \)
  - As \( \pi \) is increased, accuracy increases at cost of performance
Approximate top-k Algorithm

Terminate when less than $k-k'$ elements not in candidate top-k have their $\text{total\_score}(y) > \text{min-k}$

For exact top-k, $k-k'$ is equal to 0

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008
Outline

- Motivation
- Formal Problem Setting
- Stopping Conditions
  - Probabilistic Guarantees
  - Approximate
- Pre-computation
- Experiments
Pre-computation

- N total input lists, one for each time step
- Query on s consecutive lists
- Example: One list of popular keywords for each day of year 2007, query for popular keywords during first week of April 2007
  - N = 365; s=7
- Only consecutive lists are aggregated
  - Can not compute top-k over one list from 12th April and other from 4th May
Precomputation – Sparse System

- Since lists are available in advance, pre-computation is possible
  - Hierarchy is supplied at query time
    - Different analysts have different hierarchies

- Sparse Interval Set System
Sparse Set Construction

Given $N$ lists, each one corresponding to a point on number line. Define level $j$ points as $0, l^j, 2l^j, \ldots$. Any pair of level $j$ points between adjacent level $j + 1$ points defines an interval.

- For each interval, maintain pre-computed list:
  - $[1,2], [2,3], [1,3], [3,6], [4,6], [3,9], \ldots$
**Sparse Set Example**

- **Interval [4,6]:**
  - Aggregate lists 4, 5, and 6 in advance (without hierarchy)

<table>
<thead>
<tr>
<th>BMW, 0.9</th>
<th>Canon, 0.8</th>
<th>Canon, 0.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canon, 0.7</td>
<td>Hyundai, 0.7</td>
<td>Nikon, 0.5</td>
</tr>
<tr>
<td>Nikon, 0.6</td>
<td>Canon, 0.4</td>
<td>BMW, 0.4</td>
</tr>
</tbody>
</table>

- BMW, 2.1
- Canon, 2.0
- Nikon, 1.8
- Hyundai, 1.3

---

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008

www.blogscope.net
Querying with Sparse Set

- Assume query for time interval [1,8]
  - Requires aggregation over three pre-computed lists
  - [1,2]
  - [3,6]
  - [7,8]
- As opposed to over 8 lists without sparse set
Sparse Set Guarantees

- Performance guarantee, tunable parameter $l$
  - Speed vs. pre-computation tradeoff
- For query over $s$ consecutive lists
  - Aggregation of less than $2\log_2 s + 1$ lists required
- Worstcase pre-computation overhead guarantee
  - No list will participate in more than $O(l^2 \log l n)$ intervals
- Amenable to streaming maintenance
  - Sparse set is easily updatable when a new list is added to the system

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008

www.blogscepe.net
Outline

- Motivation
- Formal Problem Setting
- Stopping Conditions
  - Probabilistic Guarantees
  - Approximate
- Pre-computation
- Experiments
Experimental Setup

- Probabilistic top-k is not practical in our setting
- Approximate top-k
  - Performance and accuracy tradeoffs
- Sparse Set - huge performance improvements
- Real data from BlogScope www.blogslope.net
  - Live blog search and analysis engine
  - Tracking over 30M blogs, 350M posts, 3TB database
  - Produces list of top-k keywords from blogosphere, one for each day
Approximate top-k Performance

Aggregating 9 lists from BlogScope, each consisting of 50K keywords, with synthetic hierarchy having different fanout.

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008

www.blogscope.net
Sparse Set - Improvement

Conclusions

- Introduced the problem of Top-K with Hierarchies
- Difficult problem to solve with 100% accuracy
- Probabilistic approach is not practical with hierarchies
- Approximate top-k leads to performance gains
- Pre-computation can help a lot

Future work

- Aggregation over multiple levels of hierarchies
- Maintenance of continuous top-k in a streaming setting

Ad-Hoc Aggregations of Ranked Lists in the Presence of Hierarchies, SIGMOD 2008
Thanks!

Nilesh Bansal, Sudipto Guha, Nick Koudas