Interesting-Phrase Mining for Ad-Hoc Text Analytics

Klaus Berberich
(kberberi@mpi-inf.mpg.de)

Joint work with:
Srikanta Bedathur, Jens Dittrich, Nikos Mamoulis, and Gerhard Weikum
(originally presented at VLDB 2010)
Motivation

- Continuously growing wealth of unstructured text data, e.g.:
  - e-mail and instant messaging
  - social media content (e.g., twitter and facebook)
  - customer reports and product reviews
  - general Web (e.g., news portals)

- Search on this data is understood and under control (e.g., find all news articles that mention barack obama)

- Gaining insights is tedious and a task mostly left to users (e.g., identify characteristic quotations by barack obama)
Motivation

○ **Tools** available today include
  - Tag clouds
  - Search-result snippets

○ **Limitations:**
  - focus on single terms – no entity names, quotations, etc.
  - frequency-based – not necessarily interesting (e.g., stopwords)
  - global analysis – no ad-hoc document sets (e.g., determined by query)

○ **Our idea:** Identify interesting phrases that distinguish an ad-hoc document set from the document collection as a whole
Motivation

- **Tools** available today include
  - Tag clouds
  - Search-result snippets

- **Limitations:**
  - focus on single terms – no entity names, quotations, etc.
  - frequency-based – not necessarily interesting (e.g., stopwords)
  - global analysis – no ad-hoc document sets (e.g., determined by query)

- **Our idea:** Identify interesting phrases that distinguish an ad-hoc document set from the document collection as a whole
Motivation

- **Tools** available today include
  - Tag clouds
  - Search-result snippets

- **Limitations:**
  - focus on single terms – no entity names, quotations, etc.
  - frequency-based – not necessarily interesting (e.g., stopwords)
  - global analysis – no ad-hoc document sets (e.g., determined by query)

- **Our idea:** Identify **interesting phrases** that distinguish an ad-hoc document set from the document collection as a whole
Outline

- Motivation
- Problem Statement
- Our Approach
- Prior Art
- Experimental Evaluation
- Conclusion & Ongoing Research
Model & Problem Statement

- Document collection $D$

- Documents are sequences of terms (e.g., $d_{12} = \langle a \ x \ z \ b \ l \ k \ a \ q \ x \rangle$)

- Phrases are sequences of terms (e.g., $\langle a \ x \ z \rangle$, $\langle a \ x \ z \ b \rangle$, $\langle z \ b \ l \ k \rangle$, ...)

- Input: Ad-hoc document set $D' \subseteq D$ (e.g., all documents that contain barack obama)

- Output: $k$ most interesting phrases in $D'$
Model & Problem Statement

- **Document collection** $D$

- **Documents** are sequences of terms (e.g., $d_{12} = \langle a \ x \ z \ b \ l \ k \ a \ q \ x \rangle$)

- **Phrases** are sequences of terms (e.g., $\langle a \ x \ z \rangle$, $\langle a \ x \ z \ b \rangle$, $\langle z \ b \ l \ k \rangle$, ...)

- **Input:** Ad-hoc document set $D' \subseteq D$ (e.g., all documents that contain **barack obama**)

- **Output:** $k$ most interesting phrases in $D'$
Model & Problem Statement

- Document collection $D$

- Documents are sequences of terms (e.g., $d_{12} = <a x z b l k a q x>$)

- Phrases are sequences of terms (e.g., $<a x z>$, $<a x z b>$, $<z b l k>$, ...)

- Input: Ad-hoc document set $D' \subseteq D$ (e.g., all documents that contain barack obama)

- Output: $k$ most interesting phrases in $D'$
When Do We Consider a Phrase Interesting?

- We distinguish for a phrase $p$ its
  - local frequency $\text{freq}(p, D')$ in the ad-hoc document set $D'$
  - global frequency $\text{freq}(p, D)$ in the document collection $D$

- The interestingness of phrase $p$ is defined as
  $$ I(p, D') = \frac{\text{freq}(p, D')}{\text{freq}(p, D)} $$

- We consider only phrases as relevant that
  - are globally frequent, i.e., occur in at least $\tau$ documents from $D$
  - have at most length $m$ (e.g., 5 in our experiments)

**Note:** Our methods adapt to other definitions of interestingness (e.g., based on log-likelihood ratios or PMI)
Outline

- Motivation
- Problem Statement
- Our Approach
- Prior Art
- Experimental Evaluation
- Conclusion & Ongoing Research
Overview

- We precompute the global frequency $\text{freq}(p, D)$ for any relevant phrase $p$ using an apriori-style algorithm and keep a phrase dictionary that maps $p \rightarrow \text{freq}(p, D)$.
- Our methods rely on forward indexes that map each document to a representation of its content.

For a given ad-hoc document set $D'$ our methods:
- access the forward index for each $d \in D'$
- merge the $|D'|$ content representations
- output the $k$ most interesting phrases
Idea: Keep document content explicitly as a sequence of contained terms (or, term identifiers)

Benefit:
- Space-efficient

Drawbacks:
- Enumeration of phrases needed (including globally-infrequent ones)
- Requires phrase dictionary with global frequencies $freq(p, D)$
Phrases

- **Idea:** Keep globally-frequent phrases contained in document in a consistent (e.g., lexicographical) order

- **Benefits:**
  - Considers only globally-frequent phrases
  - Consistent sort order facilitates merging

- **Drawbacks:**
  - Space-inefficient
  - Requires phrase dictionary with global frequencies \( \text{freq}(p, D) \)
Frequency-Ordered Phrases

- **Idea:** Keep contained globally-frequent phrases in ascending order of their embedded global frequencies $\text{freq}(p, D)$

- **Interestingness** of any unseen phrase is upper-bounded by

$$\min \left\{ 1, \frac{|D'|}{\text{freq}(p, D)} \right\}$$

where $p$ is the last phrase encountered
Frequency-Ordered Phrases

- **Idea:** Keep contained globally-frequent phrases in ascending order of their embedded global frequencies $\text{freq}(p, D)$

- **Interestingness** of any unseen phrase is upper-bounded by

$$\min \left\{ 1, \frac{|D'|}{\text{freq}(p, D)} \right\}$$

where $p$ is the last phrase encountered.
Frequency-Ordered Phrases

- **Idea:** Keep contained globally-frequent phrases in ascending order of their embedded global frequencies \( \text{freq}(p, D) \)

- **Interestingness** of any unseen phrase is upper-bounded by

\[
\min \left\{ 1, \frac{|D'|}{\text{freq}(p, D)} \right\}
\]

where \( p \) is the last phrase encountered.
Frequency-Ordered Phrases

- **Idea:** Keep contained globally-frequent phrases in ascending order of their embedded global frequencies $\text{freq}(p, D)$

- **Interestingness** of any unseen phrase is upper-bounded by

$$\min \left\{ 1, \frac{|D'|}{\text{freq}(p, D)} \right\} = \frac{|D'|}{\text{freq}(<s>, D)} = \frac{3}{6}$$

where $p$ is the last phrase encountered
Frequency-Ordered Phrases

- **Idea:** Keep contained globally-frequent phrases in ascending order of their embedded global frequencies \( \text{freq}(p, D) \)

- **Benefits:**
  - Early termination possible when no unseen phrase can make it into the top-\( k \) of most interesting phrases
  - Self-contained (i.e., no phrase dictionary needed)

- **Drawback:**
  - Space-inefficient
Prefix-Maximal Phrases

- **Observation:** Globally-frequent phrases are often redundant and therefore do not need to be kept explicitly.

- **Definition:** A phrase $p$ is prefix-maximal in document $d$ if
  - $p$ is globally-frequent
  - $d$ does not contain a globally-frequent phrase $p'$ of which $p$ is a prefix
  - A prefix-maximal phrase (e.g., $<a x z>$) can represent all its prefixes (i.e., $<a>$ and $<a x>$) and it’s guaranteed that they’re globally-frequent and contained in $d$. 

```plaintext
\[
\begin{align*}
d_{12} &\rightarrow <a><ax><axz><b><bl> \\
d_{37} &\rightarrow <d><dl><dle><e><es> \\
d_{42} &\rightarrow <a><ak><ay><d><da>
\end{align*}
\]
Prefix-Maximal Phrases

**Observation:** Globally-frequent phrases are often redundant and therefore do not need to be kept explicitly

- \(d_{12}\) → \(<a><ax><axz><b><bl>\)
- \(d_{37}\) → \(<d><dl><dle><e><es>\)
- \(d_{42}\) → \(<a><ak><ay><d><da>\)

**Definition:** A phrase \(p\) is prefix-maximal in document \(d\) if
- \(p\) is globally-frequent
- \(d\) does not contain a globally-frequent phrase \(p'\) of which \(p\) is a prefix

A prefix-maximal phrase (e.g., \(<a><ax><axz>\)) can represent all its prefixes (i.e., \(<a>\) and \(<a><ax>\)) and it’s guaranteed that they’re globally-frequent and contained in \(d\)
Prefix-Maximal Phrases

- **Idea:** Keep contained prefix-maximal phrases in lexicographical order and extract prefixes on-the-fly

  - ![Diagram](image)

  - $\text{d}_{12} \rightarrow \langle axz\rangle\langle bl\rangle \ldots$
  - $\text{d}_{37} \rightarrow \langle dle\rangle\langle es\rangle \ldots$
  - $\text{d}_{42} \rightarrow \langle ak\rangle\langle ay\rangle\langle da\rangle \ldots$

- **Benefit:**
  - Space-efficient

- **Drawbacks:**
  - Extraction of prefixes entails additional bookkeeping
  - Requires phrase dictionary with global frequencies $\text{freq}(p, D)$
Prefix-Maximal Phrases

- **Idea:** Keep contained prefix-maximal phrases in lexicographical order and extract prefixes on-the-fly

- **Benefit:**
  - Space-efficient

- **Drawbacks:**
  - Extraction of prefixes entails additional bookkeeping
  - Requires phrase dictionary with global frequencies \( \text{freq}(p, D) \)
Prefix-Maximal Phrases

- **Idea:** Keep contained prefix-maximal phrases in lexicographical order and extract prefixes on-the-fly

- **Benefit:**
  - Space-efficient

- **Drawbacks:**
  - Extraction of prefixes entails additional bookkeeping
  - Requires phrase dictionary with global frequencies \( \text{freq}(p, D) \)
Prefix-Maximal Phrases

- **Idea:** Keep contained prefix-maximal phrases in lexicographical order and extract prefixes on-the-fly.

- **Benefit:**
  - Space-efficient

- **Drawbacks:**
  - Extraction of prefixes entails additional bookkeeping
  - Requires phrase dictionary with global frequencies $\text{freq}(p, D)$
Outline

- Motivation
- Problem Statement
- Our Approach
- Prior Art
- Experimental Evaluation
- Conclusion & Ongoing Research
MCX: Multidimensional Content eXploration

**Objective:** Identify k most frequent relevant phrases in a given ad-hoc document set D’

**Idea:** Build inverted index that maps \( p \rightarrow \{ d \mid d \text{ contains } p \} \) and sorts phrases in descending order of \( \text{freq}(p, D) \)

\[
\begin{align*}
< a > & \rightarrow d_{12}, d_{2}, d_{89}, d_{122}, d_{7}, d_{128}, d_{876}, d_{4} \\
< b > & \rightarrow d_{22}, d_{5}, d_{19}, d_{272}, d_{8}, d_{218}, d_{926} \\
< c > & \rightarrow d_{6}, d_{51}, d_{29}, d_{92}, d_{41}, d_{928} \\
< a i > & \rightarrow d_{12}, d_{52}, d_{7}, d_{218} \\
< x z k > & \rightarrow d_{2737}, d_{57} \\
< x z z > & \rightarrow d_{12}
\end{align*}
\]
**Objective:** Identify \( k \) most frequent relevant phrases in a given ad-hoc document set \( D' \)

**Idea:** Build \textit{inverted index} that maps \( p \rightarrow \{ d \mid d \text{ contains } p \} \) and sorts phrases in descending order of \( \text{freq}(p, D) \)

\[
\begin{align*}
< a > & \rightarrow d_{12}, d_2, d_{89}, d_{122}, d_7, d_{128}, d_{876}, d_4 \ldots \\
< b > & \rightarrow d_{22}, d_5, d_{19}, d_{272}, d_8, d_{218}, d_{926} \ldots \\
< c > & \rightarrow d_6, d_{51}, d_{29}, d_{92}, d_{41}, d_{928} \ldots \\
< a i > & \rightarrow d_{12}, d_{52}, d_7, d_{218} \\
< x z k > & \rightarrow d_{2737}, d_{57} \\
< x z z > & \rightarrow d_{12}
\end{align*}
\]

\( D' \cap \cap_{2} \)
MCX: Multidimensional Content eXploration

- **Objective:** Identify the most frequent relevant phrases in a given ad-hoc document set $D'$
- **Idea:** Build an inverted index that maps $p \rightarrow \{ d \mid d \text{ contains } p \}$ and sorts phrases in descending order of $\text{freq}(p, D)$
Objective: Identify \( k \) most frequent relevant phrases in a given ad-hoc document set \( D' \)

Idea: Build inverted index that maps \( p \rightarrow \{ d \mid d \text{ contains } p \} \) and sorts phrases in descending order of \( \text{freq}(p, D) \)
**MCX: Multidimensional Content eXploration**

- **Objective:** Identify **k** most frequent relevant phrases in a given ad-hoc document set **D’**
- **Idea:** Build *inverted index* that maps \( p \rightarrow \{ d \mid d \text{ contains } p \} \) and sorts phrases in *descending order of* \( \text{freq}(p, D) \)

\[
\begin{align*}
&\langle a \rangle \quad \rightarrow \quad d_{12}, d_2, d_{89}, d_{122}, d_7, d_{128}, d_{876}, d_4 \ldots \\
&\langle b \rangle \quad \rightarrow \quad d_{22}, d_5, d_{19}, d_{272}, d_8, d_{218}, d_{926} \ldots \\
&\langle c \rangle \quad \rightarrow \quad d_6, d_{51}, d_{29}, d_{92}, d_{41}, d_{928} \ldots \\
&\langle a \ i \rangle \quad \rightarrow \quad d_{12}, d_{52}, d_7, d_{218} \\
&\langle x \ z \ k \rangle \quad \rightarrow \quad d_{2737}, d_{57} \\
&\langle x \ z \ z \rangle \quad \rightarrow \quad d_{12} \\
\end{align*}
\]

\( \bigcap \quad \text{D’} \quad d_{12}, d_{52}, d_7, d_{218} \quad 4 \)
**Objective:** Identify $k$ most frequent relevant phrases in a given ad-hoc document set $D'$

**Idea:** Build inverted index that maps $p \rightarrow \{ d \mid d \text{ contains } p \}$ and sorts phrases in descending order of $\text{freq}(p, D)$
MCX: Multidimensional Content eXploration

- **Objective:** Identify \( k \) most frequent relevant phrases in a given ad-hoc document set \( D' \)
- **Idea:** Build **inverted index** that maps \( p \rightarrow \{ d | d \text{ contains } p \} \) and sorts phrases in descending order of \( \text{freq}(p, D) \)

\[
\begin{align*}
\langle a \rangle & \rightarrow d_{12}, d_2, d_{89}, d_{122}, d_7, d_{128}, d_{876}, d_4, \ldots \\
\langle b \rangle & \rightarrow d_{22}, d_5, d_{19}, d_{272}, d_8, d_{218}, d_{926}, \ldots \\
\langle c \rangle & \rightarrow d_6, d_{51}, d_{29}, d_{92}, d_{41}, d_{928}, \ldots \\
\langle a \ i \rangle & \rightarrow d_{12}, d_{52}, d_7, d_{218} \\
\langle x \ z \ k \rangle & \rightarrow d_{2737}, d_{57} \\
\langle x \ z \ z \rangle & \rightarrow d_{12} \\
\end{align*}
\]

\( \bigcap \ D' = d_{12}, d_{52}, d_7, d_{218} \/ 1 \)
MCX: Multidimensional Content eXploration

- **Objective:** Identify \( k \) most frequent relevant phrases in a given ad-hoc document set \( D' \)
- **Idea:** Build inverted index that maps \( p \rightarrow \{ d \mid d \text{ contains } p \} \) and sorts phrases in descending order of \( \text{freq}(p, D) \)

\[
\begin{align*}
< a > & \rightarrow d_{12}, d_2, d_{89}, d_{122}, d_7, d_{128}, d_{876}, d_4, \ldots \\
< b > & \rightarrow d_{22}, d_5, d_{19}, d_{272}, d_8, d_{218}, d_{926}, \ldots \\
< c > & \rightarrow d_6, d_{51}, d_{29}, d_{92}, d_{41}, d_{928}, \ldots \\
< a i > & \ldots \rightarrow d_{12}, d_{52}, d_7, d_{218} \\
< x z k > & \ldots \rightarrow d_{2737}, d_{57} \\
< x z z > & \rightarrow d_{12}
\end{align*}
\]
MCX: Multidimensional Content eXploration

- **Optimization:**
  - Approximate fast set intersections through order randomization

- **Benefits:**
  - Early termination possible when no unseen phrase can make it into the top-k of most frequent phrases
  - Performs well for very large ad-hoc document sets $D'$

- **Drawback:**
  - Considers only frequency as a coarse-grained pre-filter; re-ranks identified frequent phrases based on interestingness

- **Reference:** A. Simitsis, A. Baid, Y. Sismanis, and B. Reinwald. *Multidimensional Content eXploration*. In VLDB ’08
Outline

- Motivation
- Problem Statement
- Our Approach
- Prior Art
- Experimental Evaluation
- Conclusion & Ongoing Research
Experimental Evaluation

- **Dataset:** *The New York Times* Annotated Corpus consisting of 1.8M newspaper articles published between 1987 and 2007

- **Queries** to determine ad-hoc document sets based on
  - person-related (e.g., steve jobs, hillary clinton,...)
  - news-related (e.g., world trade center, world cup,...)
  - based on metadata (e.g., /travel/destinations/europe,...)

- **Implementation** in Java represents data compactly (e.g., variable-length encoding)

- **System:** SUN server-class machine
  (4 CPUs / 16Gb RAM / RAID-5 / Windows 2003 Server)
Examples of Interesting Phrases

- **Query: john lennon**
  1) ...since john lennon was assassinated...
  2) ...lennon’s childhood...
  3) ...post beatles work...

- **Query: bob marley**
  1) ...music of bob marley...
  2) ...marley the jamaican musician...
  3) ...i shot the sheriff...

- **Query: john mccain**
  1) ...to beat al gore like...
  2) ...2000 campaign in arizona...
  3) ...the senior senator from virginia...
Index Sizes for Different Values of $\tau$

![Graph showing index sizes for different values of $\tau$.]
Index Sizes for Different Values of $\tau$

![Graph showing index sizes for different values of $\tau$]

- **Document Content**
- **Frequency-Ordered Phrases**
- **Prefix-Maximal Phrases**
- **MCX**

Index Size (bytes)

Minimum Support Threshold $\tau$

- 0
- 10
- 20
- 30
- 40
- 50

1.80 Gb

**Table**

- Minimum Support Threshold $\tau$
  - 0
  - 10
  - 20
  - 30
  - 40
  - 50

- Index Size (bytes)
  - 0
  - 1e+09
  - 2e+09
  - 3e+09
  - 4e+09
  - 5e+09
  - 6e+09
  - 7e+09
  - 8e+09
  - 9e+09
  - 1e+10
  - 1.2e+10
  - 1.4e+10

**Note**

- Minimum Support Threshold $\tau$ affects the index size for different types of phrases.
Index Sizes for Different Values of $\tau$

Index Size (bytes)

Minimum Support Threshold $\tau$

- Document Content
- Phrases
- Frequency-Ordered Phrases
- Prefix-Maximal Phrases
- MCX

4.41 Gb

1.80 Gb
Index Sizes for Different Values of $\tau$

![Graph showing index sizes for different minimum support thresholds $\tau$. The graph compares Document Content, Phrases, Frequency-Ordered Phrases, Prefix-Maximal Phrases, and MCX. The x-axis represents the minimum support threshold $\tau$, while the y-axis shows the index size (bytes).]

- Document Content: 5.64 Gb
- Phrases: 4.41 Gb
- Frequency-Ordered Phrases: 1.80 Gb
- Prefix-Maximal Phrases: 5.64 Gb
- MCX: 4.41 Gb
Index Sizes for Different Values of $\tau$

![Index Size Graph]

- **Document Content**: 10.12 Gb
- **Frequency-Ordered Phrases**: 5.64 Gb
- **Prefix-Maximal Phrases**: 4.41 Gb
- **MCX**: 1.80 Gb

**X-Axis**: Minimum Support Threshold $\tau$

**Y-Axis**: Index Size (bytes)

**Legend**:
- Document Content
- Frequency-Ordered Phrases
- Prefix-Maximal Phrases
- MCX
Index Sizes for Different Values of $\tau$

![Index Sizes Diagram](image)

- **Document Content**
- **Phrases**
- **Frequency-Ordered Phrases**
- **Prefix-Maximal Phrases**
- **MCX**

Index Size (bytes) vs. Minimum Support Threshold $\tau$

- 10.12 Gb
- 5.64 Gb
- 4.41 Gb
- 1.80 Gb
Wall-Clock Times for Different Values of $|D'|$

- **Document Content**
- **Phrases**
- **Frequency-Ordered Phrases**
- **Prefix-Maximal Phrases**
- **MCX**

<table>
<thead>
<tr>
<th>Input Cardinality</th>
<th>Wallclock Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$100$</td>
<td>$100$</td>
</tr>
<tr>
<td>$1000$</td>
<td>$1000$</td>
</tr>
<tr>
<td>$10000$</td>
<td>$10000$</td>
</tr>
<tr>
<td>$100000$</td>
<td>$100000$</td>
</tr>
<tr>
<td>$1e+06$</td>
<td>$1e+06$</td>
</tr>
</tbody>
</table>

$k = 100$
$\tau = 10$
Wall-Clock Times for Different Values of $|D'|$

- **Document Content**
- **Phrases**
- **Frequency-Ordered Phrases**
- **Prefix-Maximal Phrases**
- **MCX**

$k = 100$
$\tau = 10$

1,030 ms
Wall-Clock Times for Different Values of $|D'|$

$k = 100$
$\tau = 10$

$3,500 \text{ ms}$
$1,030 \text{ ms}$
Wall-Clock Times for Different Values of $|D'|$

**Input Cardinality vs. Wallclock Time (ms)**

- **Document Content**
- **Phrases**
- **Frequency-Ordered Phrases**
- **Prefix-Maximal Phrases**
- **MCX**

**Table**

<table>
<thead>
<tr>
<th>Input Cardinality</th>
<th>Wallclock Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1,030 ms</td>
</tr>
<tr>
<td>1000</td>
<td>3,500 ms</td>
</tr>
<tr>
<td>10000</td>
<td>14,779 ms</td>
</tr>
<tr>
<td>$10^5$</td>
<td>1,030 ms</td>
</tr>
<tr>
<td>$10^6$</td>
<td>3,500 ms</td>
</tr>
<tr>
<td>$10^7$</td>
<td>14,779 ms</td>
</tr>
</tbody>
</table>

**Parameters**

- $k = 100$
- $\tau = 10$
Wall-Clock Times for Different Values of $|D'|$

- **Document Content**: $1,030 \text{ ms}$
- **Frequency-Ordered Phrases**: $3,500 \text{ ms}$
- **Prefix-Maximal Phrases**: $14,779 \text{ ms}$
- **MCX**: $85,575 \text{ ms}$

Input Cardinality vs. Wallclock Time (ms)

$k = 100$

$\tau = 10$
Wall-Clock Times for Different Values of $|D'|$

$k = 100$
$\tau = 10$

| Input Cardinality $|D'|$ | Wallclock Time (ms) |
|-----------------|-------------------|
| 100             | 1,030 ms          |
| 1000            | 3,500 ms          |
| 10000           | 14,779 ms         |
| 100000          | 85,575 ms         |

Document Content
Frequency-Ordered Phrases
Prefix-Maximal Phrases
MCX
Wall-Clock Times for Different Values of $k$

- Frequency-Ordered Phrases
- Frequency-Ordered Phrases w/ EarlyTermination

\[ \tau = 10 \]
\[ |D'| = 500 \]
Wall-Clock Times for Different Values of $k$

$\tau = 10$

$|D'| = 500$

357 ms
Wall-Clock Times for Different Values of $k$

```
\[ \tau = 10 \]
\[ |D'| = 500 \]

\[
\begin{array}{c|c}
\text{Result Size } k & \text{Wallclock Time (ms)} \\
\hline
10 & 357 ms \\
100 & 590 ms \\
1000 & \text{---} \\
\end{array}
\]

---

Interesting-Phrase Mining for Ad-Hoc Text Analytics (Klaus Berberich)
Wall-Clock Times for Different Values of $k$

![Graph showing wall-clock times for different values of $k$.

- Frequency-Ordered Phrases:
  - $\tau = 10$
  - $|D'| = 500$
  - Graph points:
    - $k = 10$: 357 ms
    - $k = 100$: 357 ms
    - $k = 1000$: 590 ms

- Frequency-Ordered Phrases with EarlyTermination:
  - Graph points:
    - $k = 10$: 357 ms
    - $k = 100$: 357 ms
    - $k = 1000$: 590 ms

The graph illustrates the wall-clock time for different result sizes $k$, showing how the time increases with larger $k$ values. The results indicate that early termination is effective in reducing processing time, especially for larger $k$ values.
Outline

- Motivation
- Problem Statement
- Our Approach
- Prior Art
- Experimental Evaluation
- Conclusion & Ongoing Research
Conclusion

- Efficient identification of interesting phrases on ad-hoc document sets is a challenging research problem.

- Our methods to tackle this problem:
  - are based on forward indexes
  - differ in how they represent document contents
  - Frequency-Ordered Phrases allow for early termination
  - Prefix-Maximal Phrases result in a very compact index

- Experiments on real-world dataset show that our methods work in practice and outperform the state-of-the-art method substantially on realistic inputs.
Ongoing Research

- **Alternative scenario:** Identify phrases that best distinguish two ad-hoc document sets (e.g., *barack obama* vs. *george bush* or recently published vs. all documents on *UEFA*)

- Implement our index structures and pre-computations using Hadoop (or another MapReduce implementation)

- Support grouping of almost-identical phrases (e.g., *george bush said* vs. *george w. bush said*)

- Make use of NLP techniques such as part-of-speech tagging (e.g., to abstract from concrete pronouns)
Thank you!

Questions?
Wall-Clock Times for Different Values of $\tau$

- Document Content
- Phrases
- Frequency-Ordered Phrases
- Prefix-Maximal Phrases

$k = 100$
$|D'| = 500$
Wall-Clock Times for Different Values of $\tau$

$k = 100$
$|D'| = 500$

360 ms
Wall-Clock Times for Different Values of $\tau$

$k = 100$
$|D'| = 500$

Minimum Support Threshold $\tau$

Wallclock Time (ms)

784 ms
360 ms
Wall-Clock Times for Different Values of $\tau$

- Document Content
- Phrases
- Frequency-Ordered Phrases
- Prefix-Maximal Phrases

$k = 100$
$|D'| = 500$

Wallclock Time (ms)

Minimum Support Threshold $\tau$

- 1,125 ms
- 784 ms
- 360 ms
Wall-Clock Times for Different Values of $\tau$

$k = 100$
$|D'| = 500$

<table>
<thead>
<tr>
<th>Minimum Support Threshold $\tau$</th>
<th>Wallclock Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1,125 ms</td>
</tr>
<tr>
<td>10</td>
<td>784 ms</td>
</tr>
<tr>
<td>15</td>
<td>360 ms</td>
</tr>
<tr>
<td>20</td>
<td>4,075 ms</td>
</tr>
</tbody>
</table>
Wall-Clock Times for Different Values of $\tau$

- **Document Content**: 4,075 ms
- **Phrases**: 1,125 ms
- **Frequency-Ordered Phrases**: 784 ms
- **Prefix-Maximal Phrases**: 360 ms

**Parameters**
- $k = 100$
- $|D'| = 500$