Approximate Object Location and Spam Filtering on Tapestry

Feng Zhou (zf@cs.berkeley.edu)
Li Zhuang (zl@cs.berkeley.edu)
Ben Y. Zhao (ravenben@cs.berkeley.edu)
Ling Huang (hling@cs.berkeley.edu)
Motivation

- **Objective:** Search for similar content published on Tapestry
  - Text documents: same news article altered due to formatting
  - General content with descriptive fields: MP3 file with textual tags

- **Content-hashed GUID**
  - Hashing object content to get GUID
  - Good for locating exact copies

- **Our Way**
  - Describe each object using a set of feature values
  - Build an index of these feature values on top of Tapestry
  - Search this index to find GUID of matching objects
Approximation Extension to Tapestry

Publication using features

- Objects are described using a set of features:
  \[ AO \equiv \text{Feature Vector (FV)} = \{f_1, f_2, f_3, \ldots, f_n\} \]

- Location \equiv \text{find all objects in the network with}
  \[ |FV^* \cap FV| \geq \text{THRES}, \ 0 < \text{THRES} \leq |FV| \]

Primitives

- PublishApproxObject(Object ID, FV)
- UnpublishApproxObject (Object ID, FV)
- RouteToApproxObject (FV, THRES)
Potential Applications

- **Approximate Text Addressing**
  - Problem: find similar text document copies
  - Feature Vector: *a text fingerprint vector*
  - Application: P2P spam filter, P2P content based “e-pinions”, etc.

- **Database Queries on P2P**
  - Hash values of a tuple into a feature vector
  - Feature Vector: *hashes of values to query*
  - Approximate query: \( \text{THRES} < |FV| \)

- **Media Retrieving on P2P**
  - Most of pattern recognition results of image or video are represented as a feature vector.
  - Discretize feature values
Prototype on Tapestry

- **A substrate on top of Tapestry**
  - The set of IDs of all objects matching a feature value.

- **PublishApproxObject**
  - Add Object ID to all involved Feature Objects
  - Publish new Feature Objects if needed

- **RouteToApproxObject**
  - Lookup all involved Feature Objects
  - Count occurrence of each ID and compare with THRES
  - RouteToObject
Approximate Text Addressing

- **Fingerprint Vector** [manber94finding]
  - Divide document into \((n-L+1)\) overlapping substrings (length \(L\))
  - Calculate checksums of substrings
  - The largest \(N\) checksums \(\Rightarrow\) FV

- **Parameters**
  - Length of substrings: \(L\)
  - Length of checksums: \(L_{ck}\)
  - FV Size: \(|FV| = N\)

- **Two sets of experiments**
  - “Similarity”: digest slightly changed documents into the same or similar FV? – **The higher the better!** \((1 - \text{False-negative})\)
    - We developed an **analytical model** for this
  - **False-positive**: digest totally different documents into the same or similar FV? – **The lower the better!**
P2P Spam Filtering

- Collaborative Spam Filtering

- **Observation:** *Human recognition is the only fool-proof spam identification tool.*

- **Basic idea:** connecting university or company-wide servers into a Tapestry network.

- **Why P2P?**
  - Compared with existing university/company-wide systems: The effectiveness of collaborative spam filtering systems grows with the number of users
  - Compared with existing periodically updated systems: Timeliness of information is vital in spam filtering systems
Fingerprint Vectors for Spam Filtering
- Length of substring (L): one to several phases
- |FV|: large enough to avoid collisions
- THRES is decided by doing “Similarity” Test and “False Positive” Test

Locating Spam Using Extended API
- Vote: RouteToApproxObject() to vote or PublishApproxObject() new one
- Check: RouteToApproxObject() to get current votes
- Performance Considerations
  - \(N \uparrow \rightarrow \text{Accuracy} \uparrow, \text{Network bandwidth consumption} \uparrow\)
  - Non-Spam: never published in the network \(\rightarrow \) need to route to ROOT before getting negative result
  - Solution: TTL (tradeoff between accuracy and bandwidth)
Evaluation of FV on Random Text

**Similarity Test 1**

- Probability vs. Number of Matching Fingerprints
- Task: modify 10 consecutive characters
- Graphs show comparison of 1K-sized, analytical vs. simulation and 5K-sized, analytical vs. simulation

**Similarity Test 2**

- Probability vs. Number of Matching Fingerprints
- Task: modify 50 consecutive characters
- Graphs show comparison of 1K-sized, analytical vs. simulation and 5K-sized, analytical vs. simulation

**Similarity Test 3**

- Probability vs. Number of Matching Fingerprints
- Task: modify 25 characters with 5 a group
- Graphs show comparison of 1K-sized, analytical vs. simulation and 5K-sized, analytical vs. simulation

**False Positive Test**

- Probability vs. Document Size (Byte)
- Graphs show the probability of matching fingerprint between a pair of documents with 1 or 2 matching fingerprints
- Note: Probability of 2 matching fingerprints is very low
Evaluation of FV on Real Emails

- Spam (29631 Junk Emails from [www.spamarchive.org](http://www.spamarchive.org))
  - 14925 (unique)
  - 5630 (exact copies)
  - 9076 (modified copies of 4585 unique ones)
  - 86% of spam ≤ 5K

- Normal Emails
  - 9589 (total)=50% newsgroup posts + 50% personal emails

**“Similarity” Test**
3440 modified copies of 39 emails, 5~629 copies each

<table>
<thead>
<tr>
<th>THRES</th>
<th>Detected</th>
<th>Fail</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/10</td>
<td>3356</td>
<td>84</td>
<td>97.56</td>
</tr>
<tr>
<td>4/10</td>
<td>3172</td>
<td>268</td>
<td>92.21</td>
</tr>
<tr>
<td>5/10</td>
<td>2967</td>
<td>473</td>
<td>86.25</td>
</tr>
</tbody>
</table>

**“False Positive” Test**
9589(normal)×14925(spam) pairs

<table>
<thead>
<tr>
<th>Match FP</th>
<th># pair</th>
<th>probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/10</td>
<td>270</td>
<td>1.89e-6</td>
</tr>
<tr>
<td>2/10</td>
<td>4</td>
<td>2.79e-8</td>
</tr>
<tr>
<td>&gt;2/10</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Evaluation and Status

- **Effective Fingerprint Routing w/ TTL**
  - Network of 5000 nodes
  - Diameter latency = 400ms
  - 4096 Tapestry nodes

- **Status**
  - Approximate Text Addressing prototype implemented on Tapestry.
  - **SpamWatch** – P2P spam filtering system prototype implemented
  - Outlook add-in usable!
  - Website: [http://www.cs.berkeley.edu/~zf/spamwatch/](http://www.cs.berkeley.edu/~zf/spamwatch/)