Mobile P2P Information Retrieval for Highly-Partitioned Networks

Katrina Hanna
Brian Neil Levine
R. Manmatha

Department of Computer Science
University of Massachusetts, Amherst

Supported in part by National Science Foundation awards ANI-033055, EIA-9983215, and EIA-0080199; and in part by the Center for Intelligent Information Retrieval.
Outline

- Motivation:
  - Requirements of Disaster Response
  - Mobile connections to a database
  - Limitations of Ad hoc routing
- Our solution: Mobile p2p Information Retrieval
- Design
- Evaluation
Types of Disasters

- **PICE stage 0–I disasters:**
  - E.g., chemical disasters or terrorist bombings
  - Intensive emergency response in the first few hours after an incident.
  - Acute onset
  - Limited duration
  - Local scope

- **Pan-American Health Organization:**
  - “Health crisis management cannot be accomplished without access to timely and quality information”.

- **PICE stage II–III disasters**
  - E.g.: mass population displacements; civil war; post-war Iraq.
  - Lasting days to many months
  - Gradual onset
  - International scope
  - Ongoing on-site information needs.
Disaster Response

- UMass Amherst is the disaster coordinator for western Massachusetts.
  - Currently maintains a database of:

  - Floor plans and building images
  - GIS data
  - NIOSH and Hazardous Materials data
  - Tier II data and forms
  - Action guides
  - Demographics
  - Radio frequencies for responding organizations
  - Agricultural and wild life emergency response data
  - Special skills and talents of personnel relevant to incident response (e.g. language skills, certifications, licenses and so forth)
  - Equipment and supply inventories
  - Organizations and business with services deemed useful relevant for incident consideration,
  - Populations and facilities requiring extra considerations
  - Evacuation routes
  - Shelter information
Responding to Disasters

- Stage II-III disasters require a response over a large area.
- You have to assume that critical infrastructure is down or unavailable.
- It’s likely that workers will be spread thinner than coverage of their radios provide.
- Precludes reliable deployment of ad hoc routing solution that covers the entire area.
- Imagine five doctors/medics taking off to a remote part of the disaster.
Fundamental Challenges

- Assumptions:
  - Critical infrastructure (like cell towers) is unavailable.
  - Ad hoc routing isn’t going to provide coverage of a large area.

- Several challenges:
  - Maintaining availability of the IR systems in the presence of attackers.
  - Network partitions can be frequent due to obstacles and sparseness of nodes.
  - Mobility makes it difficult to determine or predict what nodes are going to be neighbors or clustered.
  - Securing the lower layer ad hoc routing
  - How can we provide access to a database that is larger than the devices that workers will carry?

- How do we do this when we aren’t mobile?
  - Information Retrieval: returning relevant documents from a collection of documents based on a user query
  - Google is a centralized example.
Some Definitions

- Searching for a term on google returns a ranked list of results.
- Let $S$ be the set of results that google stores that are relevant to our query.
- Let $P$ be the number of results returned to us per query.
- Let $R$ be the set of results returned to us that are relevant.

$\textbf{Precision}$ is: $\frac{|R|}{|P|}$

$\textbf{Recall}$ is: $\frac{|R|}{|S|}$

As $|P|$ grows, precision drops and recall increases.
Solution

- If we can’t have access to a centralized database, let’s break the database up into pieces.
- Hope that each neighbor has a different piece, together you and your neighbors might have the whole database, or some large portion of it.

Ways to break up the database:
- Not Replicated, sources together
- Not replicated, sources split
- Replicated, sources together
- Replicated, sources-split
Distributing the Database

- Ways to break up the database:
  - Not Replicated, sources together
  - Not replicated, sources split
  - Replicated, sources together
  - Replicated, sources-split

1. A, F, K, P, U
2. B, G, L, Q
3. C, H, M, R
4. D, I, N, S
5. E, J, O, T
Distributing the Database

- Ways to break up the database:
  - Not Replicated, sources together
  - Not replicated, sources split
  - Replicated, sources together
  - Replicated, sources-split

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>11</td>
<td>16</td>
<td>21</td>
<td>26</td>
<td>31</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>41</td>
<td>46</td>
<td>51</td>
<td>56</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>7</td>
<td>12</td>
<td>17</td>
<td>22</td>
<td>27</td>
<td>32</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>42</td>
<td>47</td>
<td>52</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>8</td>
<td>13</td>
<td>18</td>
<td>23</td>
<td>28</td>
<td>33</td>
<td>38</td>
</tr>
<tr>
<td></td>
<td>43</td>
<td>48</td>
<td>53</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>9</td>
<td>14</td>
<td>19</td>
<td>24</td>
<td>29</td>
<td>34</td>
<td>39</td>
</tr>
<tr>
<td></td>
<td>44</td>
<td>49</td>
<td>54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>30</td>
<td>35</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>45</td>
<td>50</td>
<td>55</td>
<td>56</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Distributing the Database

- Ways to break up the database:
  - Not Replicated, sources together
  - Not replicated, sources split
  - Replicated, sources together
  - Replicated, sources-split

At each node:
- For each collection:
  - Store with prob $p$

1. A, B, M, K, N, U
2. G, S, T, U
4. Q, D, E, J, F
5. H, I, P,
### Distributing the Database

- Ways to break up the database:
  - Not Replicated, sources together
  - Not replicated, sources split
  - Replicated, sources together
  - Replicated, sources-split

- At each node:
  - For each collection:
    - Store with prob $p$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1,7,13,20,28,34,36,41,55</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>8,9,11,12,15,17,25,36,44,46,47,49,51,56</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>10,12,19,20,29,39,43,47,52,54,55</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td>4</td>
<td>9,11,15,24,27,30,32,38,40,56</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
</tr>
<tr>
<td>5</td>
<td>3,11,14,16,17,20,35,37,42,43,44</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
</tr>
<tr>
<td></td>
<td>41,42,43,44,45,46,47,48</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
</tr>
</tbody>
</table>
Basic Algorithm

- Peers search by asking direct neighbors only; no ad hoc routing.
- How do you evaluate success of the scheme?
  - Resulting precision of returned results.
  - The IR field has a standard set of collections, associated queries, and results judged relevant by humans.
    - This is called TREC and is set up by NIST.
    - We took the TREC 1-2-3 collection and queries 1-50.
- First, evaluate accuracy of different distributions.
- Second, evaluate in a mobile context when the number of neighbors changes dynamically.
Sample Comparison (10 documents)

Comparison: at 10 docs

- **Replicated, split (p = .05):** 7Gb; 133-144 per node
- **No-replication, split:** 3Gb; 60-66 per node
- **Replicated, no split:** 3Gb; 33-345 per node
- **No-replication, no split:** 3Gb; 15-451 per node
Comparison against Ad hoc routing

- Remaining questions:
  - What performance can we see as the density of nodes changes?
    - How does ad hoc routing to a central server compare?
  - How much work is it per node?
    - How does ad hoc routing to a central server compare?

- Scenario:
  - 50 nodes
  - random waypoint model: 2 m/s
    - hardest model; other models allow grouping of nodes.
  - Geographies: 500sq; 1000sq; 1500sq; 2000sq.
500m-by-500m comparison

Ad hoc routing to a central server

P2P IR

Precision vs. number of documents

Initiating node number

Precision

50% 50% 50%

10 docs 100 docs

10 docs 100 docs

30% 30% 30%

10 docs 100 docs

10 docs 100 docs

10 docs 100 docs

10 docs 100 docs
1500m-by-1500m comparison

Ad hoc routing to a central server
(No route to central server 40% of the time on average.)

P2P IR

Precision

10 docs
100 docs

Number of docs
Average node

50%
40%
30%
20%
10%
Additionally, each peer must route queries and replies to the central server.
Related Work

- Ad hoc routing:
  - AODV; DSR; DSDV; and so on.
- Routing in highly-partitioned environments (message ferrying):
  - Davis, Fagg, and Levine;
  - Zhao and Ammar
- Mobile web caches:
  - 7DS (Papadopouli & Schulzerine)
- Consistent File systems and databases across partitioned networks:
  - Davidson, Garcia-Molina, and Skeen;
  - Wang, Reiher, Bagrodia, and Popek
- Distributed Information Retrieval:
  - Callan, Croft, and so on.
Summary

- Ad hoc routing does not work well in sparse deployments.
- Maintaining connectivity may not be as viable as maintaining access to data.
- Our scheme:
  - Simple to implement.
  - No overhead for routing.
  - Even requirements for nodes:
    - 133Mb storage for access to a 3100Mb database.
    - No central point of attack or failure.
    - Any random collection of neighbors is sufficient.