Optimizing near duplicate detection for peer-to-peer networks

Odysseas Papapetrou*
Stefan Siersdorfer
Sukriti Ramesh
Wolfgang Nejdl
Introduction

• Near duplicate on the **content level**:
  – near duplicates: resources with minor differences
  – videos with different advertisements, text with last-update-time
  – audio/video of different quality
  – different performance of the same song

• Why near duplicate detection for P2P?
  – Multimedia
    • finding alternative sources to parallelize the download
    • finding media of different resolutions/qualities
    • detecting copies of copyrighted multimedia
    • ignore minor differences, e.g., advertisements
  – Text
    • different versions of the same text
    • ignore insignificant changes, e.g., last-update-time
    • detect copyrighted text

• **Common property:**
  – *One can decide a priori on the minimum similarity for considering two files as near duplicates*
  – *Desired detection probability*
Locality Sensitive Hashing for NDD

- Use Locality Sensitive Hashing (LSH) for building an inverted index of files/resources
  - Resources $R_1, R_2, R_3, \ldots$
  - $R_i \approx R_j$ when $\text{sim}(R_i, R_j) > \text{minSim}$
  - $\text{LSH}(R_i) \rightarrow \text{Labels} \{ \text{label}_1, \text{label}_2, \ldots, \text{label}_l \}$
  - For example, $\text{LSH}(R_i) \rightarrow \{10010, 01011, 11011\}$
  - If $\text{sim}(R_i, R_j) > \text{minSim} \rightarrow R_i$ and $R_j$ share a label w.h.p.,
  - If $\text{sim}(R_i, R_j) < \text{minSim} \rightarrow R_i$ and $R_j$ do not share a label w.h.p.
Locality Sensitive Hashing over a DHT

- LSH-based inverted index
  - $\text{LSH}(R_i) \rightarrow$ Labels $\{\text{label}_1, \text{label}_2, \ldots, \text{label}_l\}$

  - Indexing: $\text{DHT}.\text{put}(\text{label}_x, R_i)$, for $1 \leq x \leq l$, for all resources
  - Querying for near duplicates of query $R_i$: $\text{DHT}.\text{get}(R_i, \text{label}_x)$, for $1 \leq x \leq l$ $\Rightarrow$ union is potential near duplicates
  - Possible false positives
Locality Sensitive Hashing

- LSH-based inverted index
  - LSH($R_i$) $\rightarrow$ Labels \{$label_1, label_2, ..., label_l$\}

- Existing works: inverted index over DHT using the labels as keys \[LSHForest, Haghani09\]

- Crucial parameters
  - $l \uparrow$ $\rightarrow$ false positives $\uparrow$, network cost $\uparrow$, detection probability $\uparrow$
  - $k \uparrow$ $\rightarrow$ false positives $\downarrow$, network cost $\downarrow$, detection probability $\downarrow$

- Focus of our work:
  - find the optimal combination of $l$, $k$ that provides the desired detection probability for the given network $\rightarrow$ minimize network cost and make the algorithm more efficient and scalable

Optimizing Near Duplicate Detection for peer-to-peer networks
POND: Peer-to-peer Optimized Near duplicate Detection

• Coordinator
  1. Collect network statistics
  2. Compute optimal parameters
  3. Propagate optimal parameters to network

• All peers:
  1. Re-compute labels for all resources
  2. Re-index labels to DHT

• Periodic repetition to compensate for churn
POND: Peer-to-peer Optimized Near duplicate Detection

• Coordinator
  1. Collect network statistics
  2. Compute optimal parameters
  3. Propagate optimal parameters to network

• All peers:
  1. Re-compute labels for all resources
  2. Re-index labels to DHT

• Periodic repetition to compensate for churn
POND: Peer-to-peer Optimized Near duplicate Detection

- Coordinator
  1. Collect network statistics
  2. Compute optimal parameters
  3. Propagate optimal parameters to network

- All peers:
  1. Re-compute labels for all resources
  2. Re-index labels to DHT

- Periodic repetition to compensate for churn
Collecting network statistics

• Coordinator collects network statistics
  – Network size [Ganesh07]
  – Number of resources per peer
  – Probability distribution function (PDF) for all pairwise similarities in the corpus

• Sampling of a small number of neighbors
  – Pairwise similarities: peers transmit only the media representations (a few kbytes per peer)
  – PDF: represented as equi-width histogram
POND: Peer-to-peer Optimized Near duplicate Detection

• Coordinator
  1. Collect network statistics
  2. Compute optimal parameters
  3. Propagate optimal parameters to network

• All peers:
  1. Re-compute labels for all resources
  2. Re-index labels to DHT

• Periodic repetition to compensate for churn
Computing the optimal parameters (I)

• Coordinator computes optimal configuration
• Input parameters:
  – minimum similarity \( \text{minSim} \), detection probability \( \text{pr}_{\text{min}} \)
• Required statistics:
  – average #queries, number of peers \( N \)
• Cost (to minimize)
  – Maintenance: indexing the resources in the DHT
  – Query:
    • querying the DHT for the labels
    • cost for retrieving the false positives
    • cost for retrieving the true near duplicates
• Constraint
  – Detection probability \( \geq \text{pr}_{\text{min}} \)
Computing the optimal parameters (II)

Probabilities

- Reduce false positive probability: $\uparrow k$, $\downarrow l$
- Increase detection probability: $\downarrow k$, $\uparrow l$
- Optimal combination (proof in the paper)

$$k_0 = \frac{\log \left( 1 - \left(1 - pr_{min}\right)^{1/l} \right)}{\log \left( 0.5 - \frac{1}{2 \cdot \text{minSim}} \right) + \log (\text{minSim})}$$

Cost function convex $\rightarrow$ convex optimization to identify the combination with minimum cost

Querying too expensive

Maintenance too expensive
POND: Peer-to-peer Optimized Near duplicate Detection

• Coordinator
  1. Collect network statistics
  2. Compute optimal parameters
  3. Propagate optimal parameters to network

• All peers:
  1. Re-compute labels for all resources
  2. Re-index labels to DHT

• Periodic repetition to compensate for churn
Propagating the optimal parameters

• Propagating the optimal parameters
  – Dissemination over DHT [El-Ansary03]
  – Cost: $O(N)$ messages, $O(\log(N))$ time

• Each peer
  – Computes the updated labels of all its resources
  – Indexes them in the DHT: $O(\log(N))$ per resource
POND: Peer-to-peer Optimized Near duplicate Detection

• Coordinator
  1. Collect network statistics
  2. Compute optimal parameters
  3. Propagate optimal parameters to network

• All peers:
  1. Re-compute labels for all resources
  2. Re-index labels to DHT

• Periodic repetition to compensate for churn
Query execution

- Finding all near duplicates of a resource $R_q$
  - Compute the labels of the resource, according to $l$ and $k$
  - Lookup all labels at DHT $\rightarrow$ potential near duplicates
  - For each potential near duplicate
    - Send a *compact representation* of $R_q$ to the peer (a few Kbytes)
    - Retrieve the file only if it is a near duplicate
    - Large multimedia files are never transmitted over the network
Evaluation

• Datasets:
  – Reuters RCV1: 802 thousands documents, ~1 Gbyte
  – 22455 videos (TubeKit [Shah08]), 144 Gbytes
  – 22455 audios (82 Gbytes)
• Compare with non-optimized LSH
  – Network Cost
  – Retrieval effectiveness – Recall
Comparison with non-optimized alg.

- RCV1, $p_{\text{min}} = 0.8$, $\text{minSim} = 0.9$, 100000 peers
- Vary #queries per republishing period

- POND derives configuration with *minimal* cost
- Same probabilistic guarantees and recall with non-optimized LSH
Effect of desired detection probability:: Network cost

- Maintenance cost per resource/query cost per query
- Cost can be controlled using $p_{\text{r}}_{\text{min}}$
- Manageable for large collections, e.g., for indexing 100 videos with $p_{\text{r}}_{\text{min}}=0.9$, only $\sim2000$ small messages required
- All messages are equi-sized and below 1Kbyte $\rightarrow$ transfer volume proportional to #messages
Effect of desired detection probability:: Recall

- Probabilistic guarantees always satisfied
- Recall:cost tradeoff fine-tuned with $p_{r_{\text{min}}}$
- Recall insensitive to $\text{minSim}$: algorithm adapts the parameters to satisfy $p_{r_{\text{min}}}$
Conclusions

• Target: Determine the $l$ and $k$ values that minimize the network cost and satisfy the probabilistic guarantees

• Performance improvements easily reaches an order of magnitude

• Additional information in the paper
  – Compact representations for text, audio, video
  – Video linkage, with extensive evaluation

• Future work
  – Repeat analysis using different network configurations [LSHForest05, Haghani09]
  – Effect of similarity function
  – Possible extension to other application scenarios, such as tag recommendation and annotation sharing
Thank you

Questions?
Evaluation of video linkage

• Video linkage:
  – Experimental evaluation:
    • Split video to X parts (X={2,3,4})
    • $p_{r\text{min}}=0.9$, $\text{minSim}=0.9$
    • Use any one of the parts as a query, and try to detect the original file
    • Cost: At most 110 messages, for the largest videos
Effect of desired detection probability

- **Recall:**

- **Cost:**

Optimizing Near Duplicate Detection for peer-to-peer networks
Related work

Existing work on NDD

– P2P MACSIS [Yang03]
  • NDD for audio files
  • Based on gossiping

– Optimizing LSH for centralized systems [Dong08]
  • Focuses on computational cost

– LSH with p-stable distributions [Haghani09]

– LSH Forest [LSHForest05]
  • Repeating the analysis of POND for these network configurations
Further details (I)

• Extensions presented in the paper
  – Compact representations for text, audio, video
    • Independent of binary encoding and resolution
    • $|\text{representation}(R_i)|$ only a few Kbytes, even for videos
    • DHT.put($R_i$. label$_x$, $\text{representation}(R_i)$)
    • Instead of exchanging the resources, peers exchange representations
Further details (II)

• Extensions presented in the paper
  – Video linkage
    • For practical reasons, users may break large videos e.g., titanic.avi → titanic-part1.avi and titanic-part2.avi
    • Use keyframes to *conceptually* split each video to smaller segments
    • Expected number of segments configurable
    • Each video segment is handled individually, w.r.t. indexing and query execution
    • Discovering one segment sufficient for full linkage
    • Experimental evaluation