Efficient Content Location Using Interest-Based Locality in p2p Systems

Παρουσίαση: Κώστας Στεφανίδης, Βίκυ Τζιοβάρα, Τάσος Καραγιάννης, Θοδωρής Τσώτσος

Authors: Kunwadee Sripanidkulchai, Bruce Maggs, Hui Zhang

Infocom 03

Abstract

- A significant problem in decentralized peer-to-peer systems is locating content
- Gnutella, a popular file-sharing application, relies on flooding queries to all peers
- Flooding is:
  - simple and robust
  - not scalable
- This study is about retaining the simplicity of Gnutella, while addressing its inherent weakness: scalability

Abstract

- Here, authors propose a content location solution:
  - peers loosely organize themselves into an interest-based structure on top of the existing Gnutella network
- *Interest-based locality* posits:
  - if a peer has a particular piece of content that one is interested in, it is very likely that it will have other items that one is interested in as well
- Using the algorithm *interest-based shortcuts*:
  - a significant amount of flooding can be avoided, making Gnutella a more competitive solution

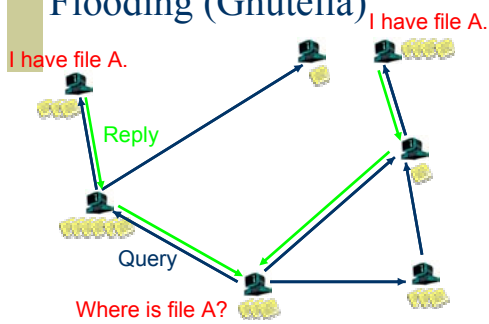
Abstract

- Shortcuts are modular:
  - can be used to improve the performance of other content location mechanisms including distributed hash table schemes (DHT)

Gnutella

- Unstructured content location, used by Gnutella, relies on flooding queries to all peers
- Peers organize into an overlay
- A peer sends a query to its neighbors on the overlay
- In turn, the neighbors forward the query on to all of their neighbors until the query has traveled a certain radius

Flooding (Gnutella)

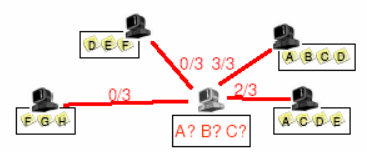


- State: No state maintained (no index)
- Search scope: flooding reaches O(N) hosts
- Simple, robust, but not scalable

Interest-Based Shortcuts

- The *interest-based shortcuts* protocol defines:
  - Peers that share similar interests create shortcuts to one another
- Peers use these shortcuts to locate content
- When shortcuts fail, peers resort to using the underlying Gnutella overlay
- Shortcuts provide a *loose* structure on top of Gnutella's unstructured overlay
- Shortcuts are also compatible with many other content location mechanisms
  - DHTs and hybrid centralized-decentralized architectures

Interest-Based Locality



- The peer in the middle is looking for files A, B, and C
- The peer on the upper right-hand corner has all three files
- It and the peer in the middle share the most interests, where interests represent a group of files, namely {A, B, C}
- The goal is to identify such peers, and use them for downloading files directly

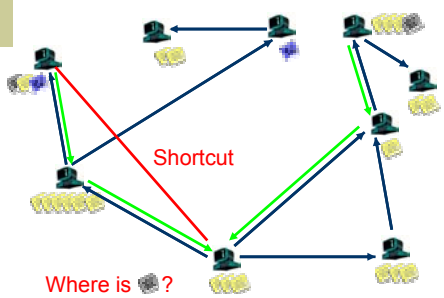
Advantages of Shortcuts

- The benefits of such an implementation are twofold:
  - First, shortcuts are modular in that they can work with any underlying content location scheme
  - Second, shortcuts only serve as performance-enhancement hints. If a document cannot be located via shortcuts, it can always be located via the underlying overlay
- In general, shortcuts are a powerful primitive that can be used to improve overlay performance

Using Shortcuts

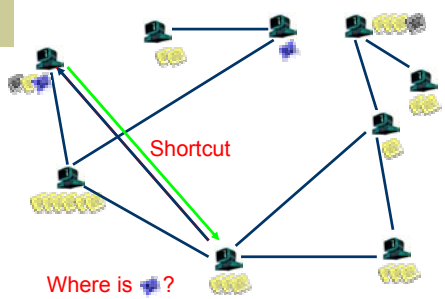
- To avoid flooding, first the query is entered to peers through shortcuts
- A query is flooded to the entire system only when none of the shortcuts have the content

Discover interest-based shortcuts



No shortcut.
Discover and add shortcut.

Use interest-based shortcuts



Use shortcut. Success!
O(1) scope for most searches.
No index (state) maintained.

Discover Shortcuts

- When a peer joins the system, it may not have any information about other peers' interests
  - Its first attempt to locate content is executed through flooding
  - The lookup returns a set of peers that store the content
  - These peers are potential candidates to be added to a shortcut list
  - One peer is selected at random from the set and added
  - Subsequent queries for content go through the shortcut list
  - If a peer cannot find content through the list, Gnutella is used, and repeats the process for adding new shortcuts

Discover Shortcuts

- Each peer allocates a fixed-size amount of storage to implement shortcuts
- Shortcuts are added and removed from the list based on their perceived utility
  - Metrics are used to compute a ranking of shortcuts
- Shortcuts that have low utility are removed from the list when the list is full

Discover Shortcuts

- New shortcuts may be discovered by:
  - Exchanging shortcut lists between peers
  - Establishing more sophisticated link structures for each content category
- Multiple shortcuts, as opposed to just one, may be added to the list at the same time

Select Shortcut

- Given that there may be many shortcuts on the list, which one should be used?
- Here there is a ranking of shortcuts based on their utility
- Useful shortcuts are ranked at the top of the list
- A peer sequentially asks all shortcuts on its list, starting from the top, until content is found

Select Shortcut

- Metrics used for ranking shortcuts:
  - probability of providing content
  - latency of the path to the shortcut
  - available bandwidth of the path
  - amount of content at the shortcut
  - load at the shortcut
- A combination of metrics can be used based on each peer's preference

Select Shortcut

- Each peer continuously keeps track of each shortcut's performance and updates its ranking when new information is learned
- This allows for peers to adapt to dynamic changes and incrementally refine shortcut selection

Metrics

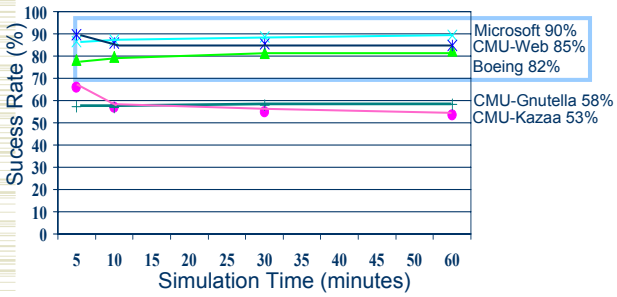
- The metric used here to express the benefits and overhead of interest-based shortcuts is:
  - Success rate: Success rate is defined as the number of lookups that were successfully resolved through interest-based shortcuts over the total number of lookups

Methodology – query workload

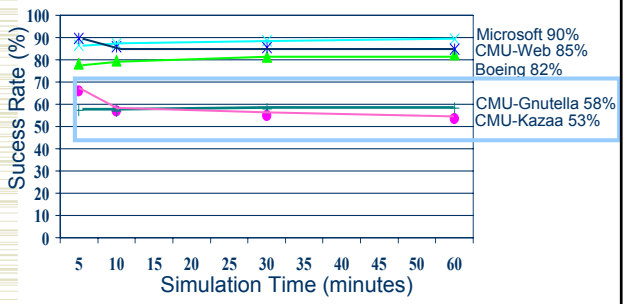
5 diverse traces of download requests are used:

- Web content sharing
  - Boeing proxy
  - Microsoft proxy
  - CMU-Web
- Multimedia file-sharing (songs and movies)
  - CMU-Gnutella
  - CMU-KaZaa

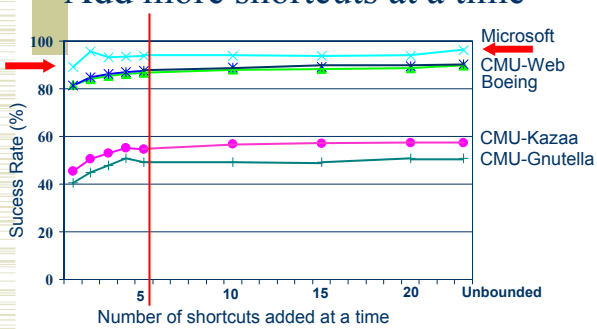
Successfully find content through shortcuts



Successfully find content through shortcuts

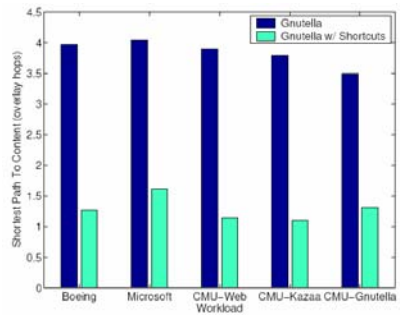


Add more shortcuts at a time



Basic algorithm (add 1) is surprisingly good.
Add 5 at a time is comparable to adding more.

Shortest Path to Content



Replication Techniques

Replication Model

- The first copy of content is placed at the peer who makes the first request for it
- Subsequent copies of content are placed based on accesses
- For instance, if peer P_1 downloaded file A at time t_0, P_1 creates a replica of file A and makes it available for others to download after time t_0

Replication Techniques

- Two methods for replication could be the following:
  - Replicate shortcut list
  - Replicate the actual file

Replicate Shortcut List

- When a peer A joins the system executes flooding to create his shortcut list
- The lookup returns a set of peers
- He chooses randomly one of them (peer B) to be added to the list
- In addition, he *copies* B's list to initiate his own list

→ The above consideration is based on the principle of interest-based locality

Replicate Shortcut List

- Using shortcuts' shortcuts
- In case a peer A uses frequently the shortcut list of a peer B (B included in A's shortcut list)
  - A replica of B's list is added to peer A

Replicate The Actual File

- If many shortcuts exist for a specific peer A (i.e., A has many popular files, so receives many requests)
  - A makes replicas of his files to some other peers (especially peers with not many requests to serve)

→ A replies to about half of the requests he receives. He informs the rest of the peers to update their shortcut list with the peers containing those replicas

Thank You