Using Structured P2P Overlay Networks to Build Content Sensitive Communities

Presented by 江文德

Outline

- Introduction
- Ingredients of the Design
- Putting the Ingredients Together
- Conclusions

Introduction

- Community

  a body of people having common rights, privileges, or interests

- Problems with Current P2P Systems

  Napster：vulnerable

  Gnutella：poor scalability

Introduction（Cont.）

- Existing Structured P2P Overlays

  DHT：Pastry, Tapestry, CAN, Chord

  advantages：decentralization, robustness, scalability

- Content Networks

  a content network is an overlay IP network that supports *content routing*

Ingredients of the Design

- Pastry
  - A circular id space
  - Each node in it is assigned a 128-bit identifier
  - Messages routed before are produced hash codes by SHA-1 hashing algorithm
- Vector Space Modeling
- Clustering
  - Hierarchic clustering
  - Linear time clustering

Pastry

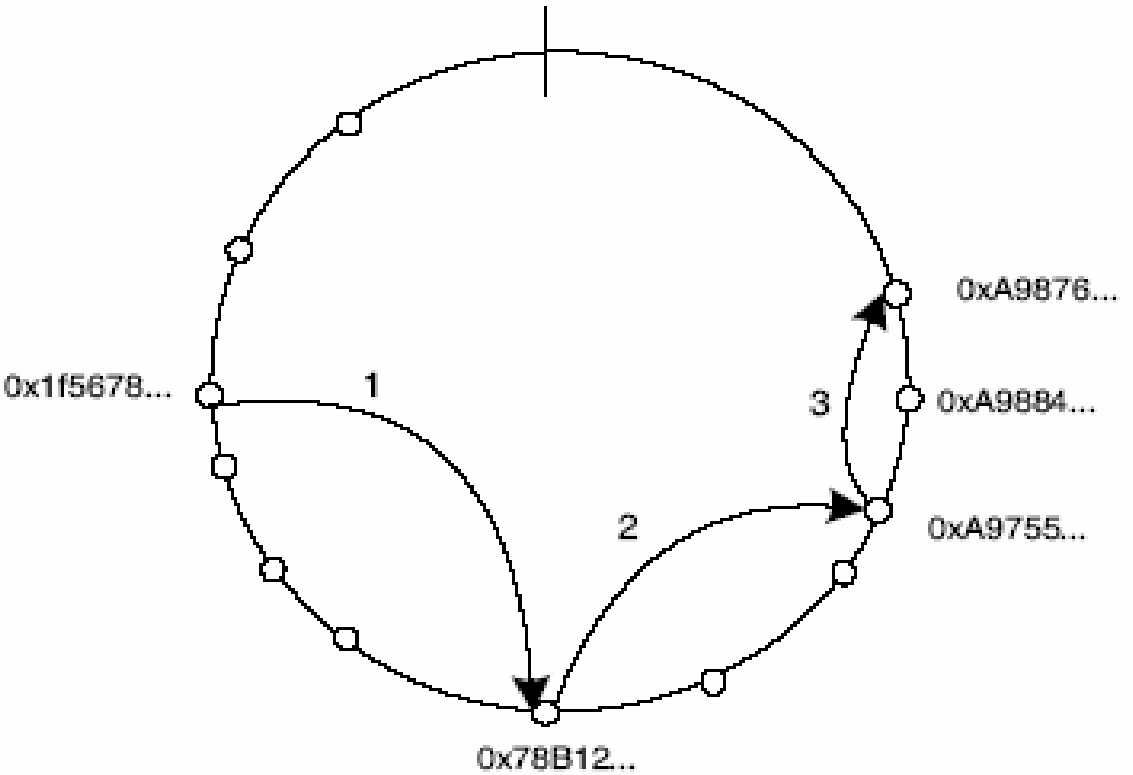

Figure 1. Pastry routes messages to nodes whose nodeIds are progressively closer to the message key.

Vector Space Modeling

- A document would be represented as a *vector*
- The simplest of the vector：

  $d_{tf} = (tf_1, tf_2, \ldots\ldots, tf_n)$

  where tf_i is the frequency of the *ith* term in the document

Vector Space Modeling

- A widely used refinement to this model is weight each term by multiplying the frequency：

$d_{tf} = (tf_1, tf_2, \ldots\ldots, tf_n)$

$d_{tfidf} = (tf_1 \log(N/df_1), tf_2 \log(N/df_2), \ldots\ldots, tf_n \log(N/df_n))$

where N is the total number of documents, and

df_i is the number of documents that contain the *ith* term

Vector Space Modeling

- To compare documents and establish a similarity index：
  - Step1：normalize each document vector

$$\| d_{tfidf} \|_2 = 1$$

  - Step2：use the cosine function to check the angle between vectors

$$cos\,(d_i, d_j) = d_i \cdot d_j$$

Putting the Ingredients Together

- Building a Decentralized Indexing Service
- Building Content Sensitive Communities
  - Comparing Nodes Based on Content Stored
  - Organizing Network into Communities
  - Two Layer Network
- Searching for Documents

Building a Decentralized indexing service

- Any node joining the network will have a set of *keyphrases*.
- Through *keyphrases* routed, each destination node is required to register the new node.
- A *registry-message* contains the node's details such as *IP-Address* and *node vector*

Building a Decentralized indexing service

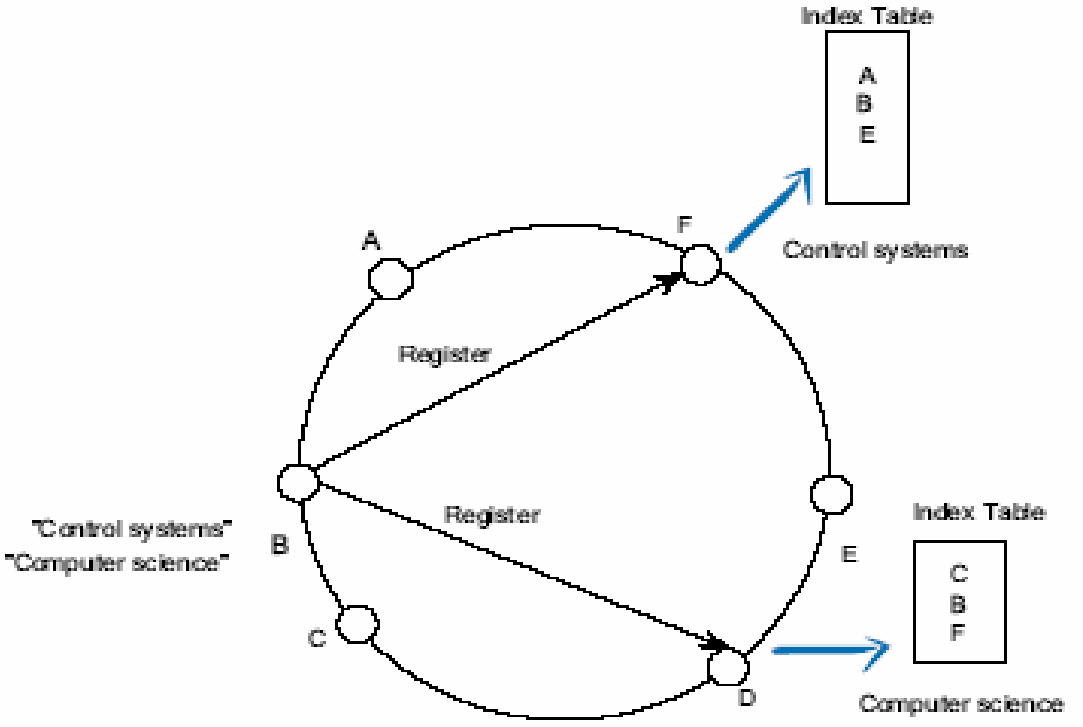

Figure 2. Node B routes a register message with 2 message keys that are the cryptographic hash of the keyphrases throughout the Pastry ring. Index tables registering all nodes sharing the same keyphrases are constructed.

Comparing Nodes Based on Content Stored

- A centroid vector C：

  The *average vector* of the set of S documents, and the *average vector* is also called the *node-vector*

$$C = \frac{1}{|S|}\sum_{d \in S} d$$

where d is the vectors of S documents

Organizing Network into Communities

- Community tables：

  Containing a list of nodes that are similar

  （based on the similarity metric from the *vector space model*）

- The *boundaries* of communities are not defined strictly

Organizing Network into Communities

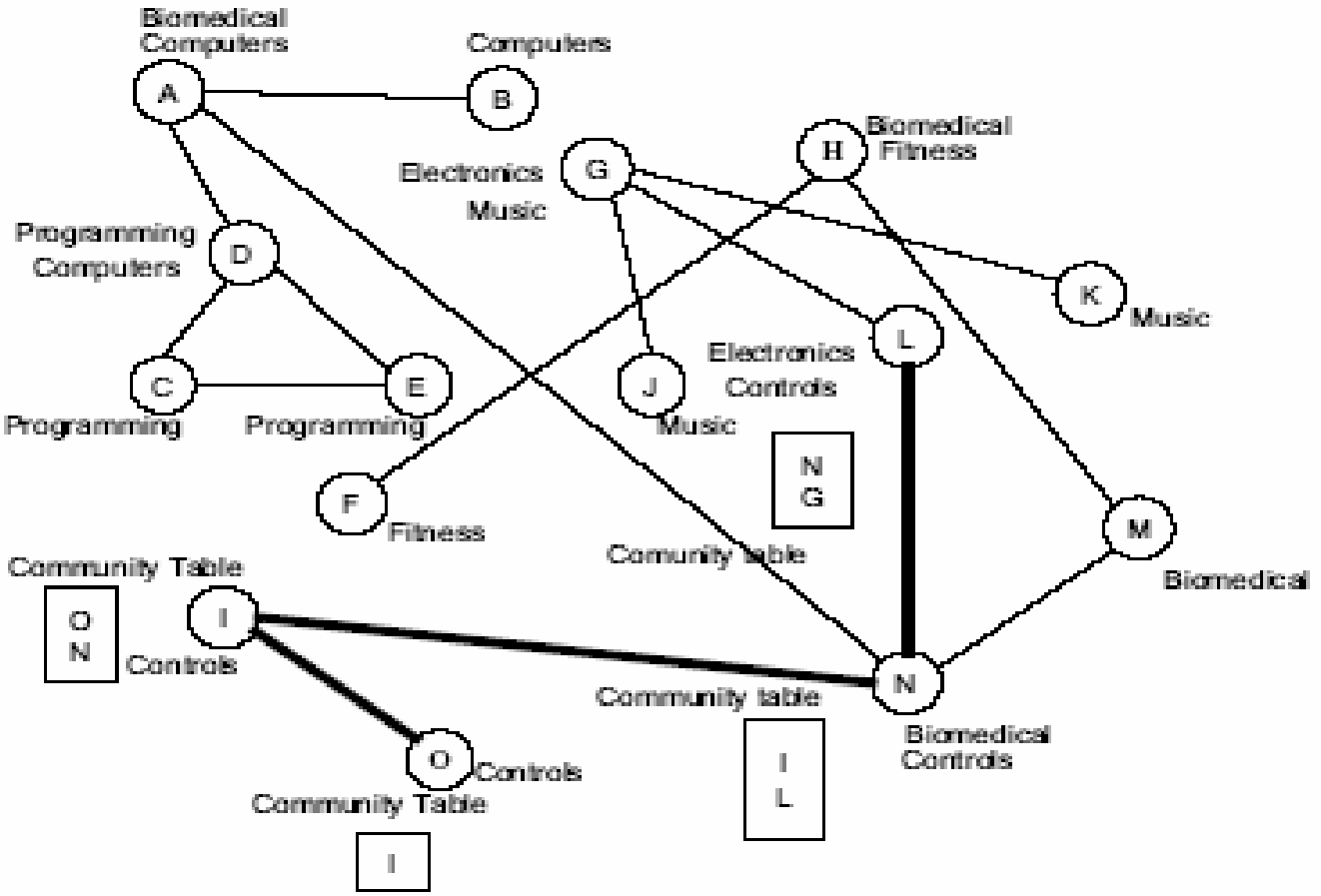

Figure 3. Shows the effect of the community layer formed by community tables

Two Layer Network

- Pastry layer
  - Maintaining routing tables and leaf sets
  - A means for nodes to find indexing nodes of certain subject areas
- Community layer
  - The ability of this layer to organize itself is a direct result of the indexing service built on top of Pastry

Two Layer Network

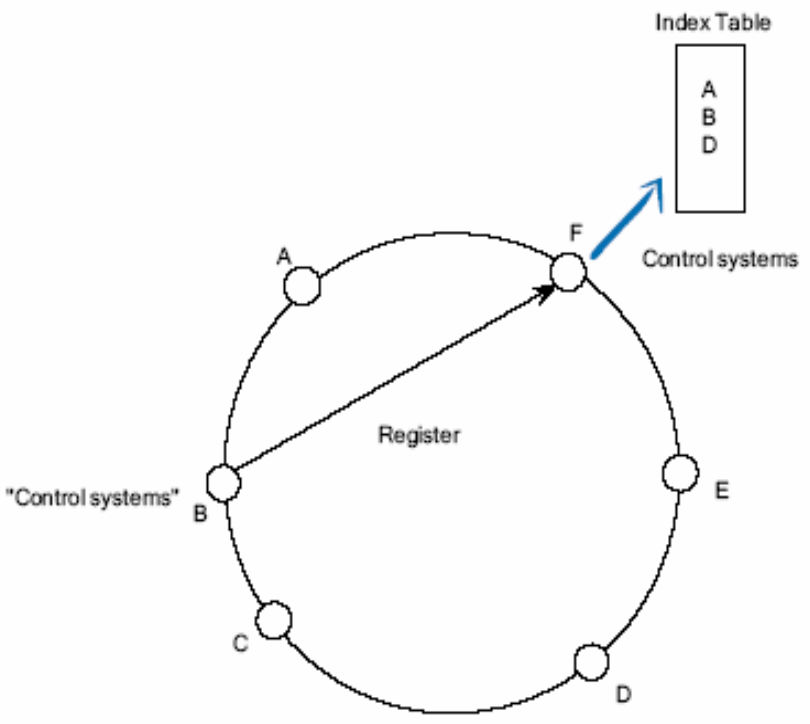

Figure 4A. The new node B populates its community table with the other nodes whose node-vectors are most similar to its own.

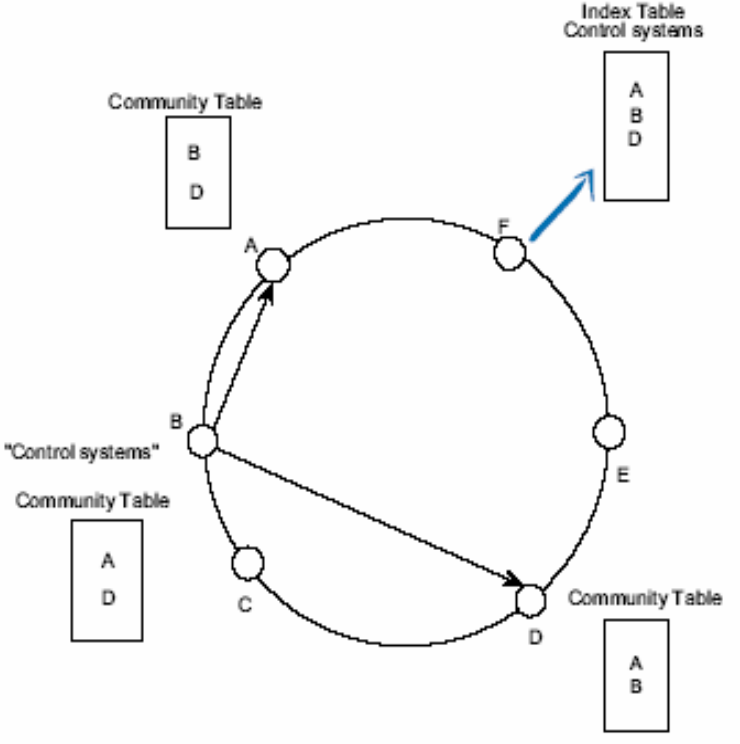

Figure 4B. Node B then informs nodes A and D that they must add B to their community tables.

Searching for Documents

- Step1：

  A search-vector can be produced in the same way a file vector was calculated.

- Step2：

  The search-vector is compared locally to nodes within the community table.

Searching for Documents

- Step3：

  The search request is then forwarded to those nodes whose node-vectors are the most similar to the search vector.

- Step4：

  The contacted node performs a flood-search on its community table using the original keywords as the search parameters.

Conclusions

- Combine the structured P2P overlay systems with Information Retrieval（IR）techniques.
- The use of keyphrases may not provide the accuracy.
- The use of node-vectors could be naive and a more accurate implementation may be need to investigated.