k-NN Multimedia Retrieval in Unstructured Peer-to-Peer Networks

Bo Yang
Department of Computer Science and Engineering
The Pennsylvania State University, University Park, PA 16802-6106

Abstract
AbstractRecent years saw the rapid development of peer-to-peer (P2P) networks in a great variety of applications. However, similarity-based k-nearest-neighbor retrieval (k-NN) is still a challenging task in P2P networks due to the multiple constraints such as the dynamic topology and the unpredictable data updates. Caching is an attractive solution that reduces network traffic and hence could remedy the technological constraints of P2P networks. However, traditional caching techniques have three major shortcomings when dealing with nearest-neighbor retrieval: First, they rely on exact match and therefore are not suitable for approximate and similarity-based queries. Second, the description of cached data is defined based on the query context instead of data content, which leads to inefficient use of cache storage. Third, the description of cached data does not reflect the popularity of the data, making it inefficient in providing QoS-related services. To facilitate the efficient similarity search, we propose semantic-aware caching scheme (SAC) in this paper. Several innovative ideas are used in the SAC scheme: 1) describing a collection of data objects using constraint-based expression showing the content distribution, 2) adaptive data content management, and 3) non-flooding query processing. By exploring the content distribution, SAC drastically reduces the cost of similarity-based k-NN retrieval in P2P networks. The performance of SAC is evaluated through simulation study and compared against several search schemes as advanced in the literature.

1. Introduction

In recent years, peer-to-peer (P2P) networks are becoming popular in providing the ability of sharing data sources at a large scale. Conceptually, a P2P network is a collection of cooperative nodes that communicate with each other without the intervention of centralized indexing servers. These nodes are capable of not only storing and processing data, but also performing complex operations through their communications, such as P2P lookup [18] or multimedia data streaming [19]. Although many of the previous research focuses on the path finding protocols that adapt to the dynamic network topology, information retrieval is becoming a hot issue in the applications of P2P networks. From the viewpoint of information retrieval, the organization of P2P networks can be classified into three categories:

Some earlier P2P networks, such as the Napster, are linked to centralized data source nodes (data centers) that host constantly updated directories of data contents. Queries issued from the client nodes are resolved at the data centers and the results are forwarded back to the requesting nodes through unicasting. Such centralized organization does not scale well and has the single points of failure. Moreover, the data center behaves as a hotspot and its data updates could increase the network traffic.

The more recently proposed P2P network frameworks are decentralized and have no data centers. The most commonly used frameworks are unstructured P2P networks, where the nodes form peer-to-peer connections among them and resolve queries through the cooperations with peers. Flooding is the most common approach for information retrieval in such P2P networks, since the requesting node does not have any information of the data contents of other nodes and has to employ the blind search. However, the flooding approach achieves good performance only when dealing with text information [21] due to its drastic consumption of system resources — storage, bandwidth, and energy. Considering the sheer size of the multimedia data, the performance deterioration is more drastic. In addition, the flooding strategy may cause duplicated queries and retrieval results, which may further increase the cost of the query processing.

To overcome the shortcomings of the blind search, the structured P2P networks were proposed in the recent literature as an alternative framework [20]. In such networks, the data objects are placed not at random nodes but at specified locations that will make subsequent queries easier to satisfy. Moreover, the topology of such networks is strictly controlled and does not change drastically. Such designs improve the efficiency of information retrieval in some cases; nevertheless, at the cost of sacrificing the flexibility and scalability of the P2P networks. In practical applications, the network topology and the data contents of the nodes are constantly changing, which increase the difficulty of efficient data retrieval.

Due to the aforementioned reasons, P2P networks cannot utilize classical content-based retrieval methods that are based on centralized or flooding mechanisms. To overcome this difficulty, this paper describes a semantic-aware caching scheme (SAC) that facilitates k-NN retrieval in unstructured P2P networks. We address the

The Office of Naval Research under contract N00014-02-1-0282 and National Science Foundation under contract IIS-0324835 in part have supported this work.

fundamental problem of supporting nearest-neighbor retrieval within the network, in which each node caches a semantic content of earlier queries. By analyzing the cache content, an overview of data distribution in the network is obtained, and the later queries are resolved with optimized search cost.

The rest of this paper is organized into five sections: Section 2 introduces the background knowledge and related work. Section 3 outlines the preliminary concepts and the caching rationale. Section 4 evaluates the proposed scheme using experimental analysis. Section 5 draws the paper into conclusions.

2. Background and Related Work

2.1 Similarity-Based Retrieval

For years, similarity search on numeric data has attracted considerable research interest, especially in the multi-dimensional spaces, e.g. image feature space. The queries of semantically similar data are performed by conducting nearest-neighbor retrieval in the multi-dimensional spaces. The similarity search approaches that have advanced in the literature can be categorized into three classes: partition-based approaches, region-based approaches, and annotation-based approaches.

The partition-based approaches recursively divide the multidimensional spaces into disjoint partitions, with clustering or classifying algorithms, while generating a hierarchical indexing structure on these partitions. The earlier models in this class include quad-tree [8], k-d-tree [9], and vp-tree [9]. The recent research has focused on models based on clusters [7,10]. The partition-based approaches normally employ very complex computations, which make them inefficient for real-time data retrieval applications.

The region-based approaches employ small bounding regions (either in the form of minimum bounding rectangles or spheres) to cover all the data objects. Based on these bounding regions, a balanced tree is constructed as the indexing structure. This class of indexing models includes the R-tree and its variations (R*-tree, R+-tree) [11], and SR-tree [12]. Relative to partition-based models, this indexing model improves storage utilization by avoiding forced splits. However, the data objects grouped in the same region may not share common semantic contents. Moreover, the performance of the region-based approaches degrades rapidly due to the existence of dimensionality curse [10].

The annotation-based approaches add some attached information to the data objects to facilitate similarity search. Gionis et al. [22] studied the relevance problem of similarity search for set data by embedding them into binary vectors in Hamming space using Min Hashing technique and error correcting codes. Sarwar et al. [23] used relevance feedback to improve the accuracy of similarity comparisons. Aggarwal et al. [24] propose the Signature table to map multi-dimensional data into strings called Super-coordinates, which are used as indexing entries. Tousidou et al. [25] and Nanopoulos et al. [26] proposed tree-structure indices called Signature trees, which employ bitmap signature to indicate the content relationships between parent/child nodes in the indexing trees. The annotation-based approaches achieve high accuracy in similarity search, and are efficient in handling data of different dimensionalities. However, these approaches are still based on centralized systems, and cannot be directly applied to wireless P2P networks.

In a P2P network with heterogeneous distributed data sources, traditional information retrieval methods, i.e., centralized or flooding strategies, may not work efficiently mainly because of the changes in network topology.

- Due to the heterogeneity and node mobility, the centralized search strategy is not an efficient choice in performing similarity search. Moreover, this strategy may also cause single point of failure, which results in low robustness.

The flooding search strategy achieves good performance only when dealing with text information. Due to the sheer size of the multimedia data, the flooding strategy may drastically consume system resources.

2.2 Caching Models

Caching has been widely used in P2P environment to reduce network traffic and improve system performance. Most of the previous study of caching for P2P networks focused on the efficient exploration of routing information [2] with only a few caching schemes to address the data retrieval issue [3].

2.2.1 Data caching

The data caching in P2P networks is a natural extension of the caching schemes in centralized networks ─ to keep a copy of the data items that have recently been accessed. Traditional schemes let a node cache either the results of its recent queries or the data that have been forwarded though it to other nodes [3]. The data caching scheme proposed in [15] allows the caching of queries as the semantic descriptions of the cached data. Such a caching scheme is efficient only for small-size data items, and cannot effectively deal with large-size data such as

images. In general, the classical semantic caching approaches have three major drawbacks in dealing with image retrieval: First, the traditional techniques rely on exact match of data items; however, image queries are usually imprecise and similarity-based. This characteristic of image retrieval makes it difficult for the traditional techniques to exploit the cached data. Second, the semantic description of the cached data is obtained in the context of queries, and any change of queries will cause reorganization of the cache, which leads to the inefficient utilization of the cache space. Third, the semantic description does not reflect the popularity of data, making it inefficient in providing QoS-related services.

2.2.2 Path caching

Path caching is another application of caching in P2P networks ─ to record a path to the data source. The scope of path caching was further extended to the domain of data replica allocation in [13, 16]. The CachePath scheme proposed in [3] dynamically caches the path information of passing-by data. These schemes, in general, consider the data items as independent entities and do not utilize the semantic locality among them. As a result, they do not explore content distribution in the P2P network.

The work presented in this paper differs from the previous efforts since it is intended to devise a caching scheme that facilitates the content-based image retrieval in a dynamic distributed environment such as a P2P network.

3. Problem Formulation

3.1 Overview

We consider a P2P network where each node has a set of data objects to share with other nodes in the network. Each data object $\boldsymbol{x}$ is represented as a n-dimensional semantic vector $\boldsymbol{\varphi_x} = (\omega_{1x}, \omega_{2x}, \ldots, \omega_{nx})$. The attributes in the semantic vector could be manually added keywords, automatically extracted features, or relevance feedback from users. The P2P network can be considered as an undirected connected graph $\boldsymbol{G} = (\boldsymbol{N}, \boldsymbol{C})$, where $\boldsymbol{N} = \{\boldsymbol{n_1}, \boldsymbol{n_2}, \ldots, \boldsymbol{n_r}\}$ is the set of nodes and $\boldsymbol{C} = \{\boldsymbol{c_1}, \boldsymbol{c_2}, \ldots, \boldsymbol{c_s}\}$ is the set of wireless connections between nodes. Each node $\boldsymbol{n_i}$ may have a collection of data objects $\boldsymbol{x_1}^i, \boldsymbol{x_2}^i, \ldots, \boldsymbol{x_r}^i$, denoted as the node content $\boldsymbol{\chi(n_i)}$. A data object $\boldsymbol{x_j}$ may exist in more than one nodes (i.e. replications), which are collectively denoted as a node set $\boldsymbol{\psi(x_j)}$. Figure 1 illustrates the relationship between data objects and nodes.

Without loss of generality, we assume the data objects $\boldsymbol{X} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_m}\}$ are represented as data points in a n-dimensional semantic space $\boldsymbol{R^n}$. The semantic similarity between two data objects is defined based on the Euclidean distance between their corresponding data points in $\boldsymbol{R^n}$. Formally, the distance between two data objects $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ is defined as a cosine distance function $dist(\boldsymbol{x_i}, \boldsymbol{x_j}) = \cos^{-1} \dfrac{\boldsymbol{\varphi_{xi}} \bullet \boldsymbol{\varphi_{xj}}}{\| \boldsymbol{\varphi_{xi}} \|_2 \| \boldsymbol{\varphi_{xj}} \|_2}$, where $\|.\|_2$ denotes the Euclidean vector norm.

As can be seen from figure 1, each data object $\boldsymbol{x_i}$ can be mapped to a set of nodes $\boldsymbol{N'} = \{\boldsymbol{n'_1}, \boldsymbol{n'_2}, \ldots, \boldsymbol{n'_s}\}$. If $|\boldsymbol{N'}| > 1$, then $\boldsymbol{N'}$ indicates exactly the set of replications of $\boldsymbol{x_i}$ in the P2P network $\boldsymbol{G}$. Conversely, each node $\boldsymbol{n_i}$ may contains a set of data objects $\boldsymbol{\chi(n_i)} = \{\boldsymbol{x_1}^i, \boldsymbol{x_2}^i, \ldots, \boldsymbol{x_r}^i\}$. $\boldsymbol{\chi(n_i)} \subset \boldsymbol{X}$, and can be partitioned into several subsets according to the semantic relationships between data objects. Several observations may also be obtained from figure 1.

Observations: If we divide the objects in $\boldsymbol{\chi(n_i)}$ into several clusters $\boldsymbol{\epsilon_1}, \boldsymbol{\epsilon_2}, \ldots, \boldsymbol{\epsilon_t}$, satisfying the following conditions:

- Data objects within the same cluster have semantic distance smaller than a pre-defined threshold $\boldsymbol{T_d}$.
- Distance between two data objects from different clusters is larger than $\boldsymbol{T_d}$.

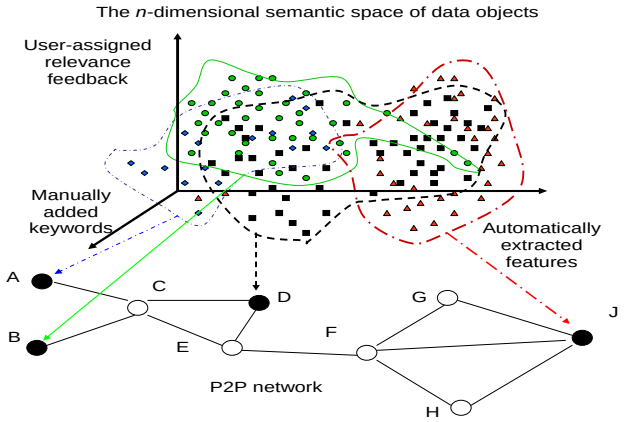

Figure 1:The relationship of data objects and nodes.

Then each cluster $\boldsymbol{\epsilon_i}$ corresponds to a node cover $\boldsymbol{N^c(\epsilon_i)}$ in the P2P network $\boldsymbol{G}$, and the node covers have the following characteristics:

- $\bigcup_{i=1}^{t} \boldsymbol{N^c(\epsilon_i)} = \boldsymbol{N}$. (Observation 1)
- Let $\boldsymbol{\chi(N^c(\epsilon_i))}$ denote the objects of the nodes in $\boldsymbol{N^c(\epsilon_i)}$, then $\bigcup_{i=1}^{t} \boldsymbol{\chi(N^c(\epsilon_i))} = \boldsymbol{X}$. (Observation 2)

- Given two clusters ϵ_i and ϵ_j, let N^* denote $N^c(\epsilon_i) \cap N^c(\epsilon_j)$. Then nodes in N^* contains data objects from both ϵ_i and ϵ_j, therefore the similarity query issued to nodes in N^* can be resolved as the set intersection of query results returned from both ϵ_i and ϵ_j. (Observation 3)
- Let $\{ \epsilon'_1, \epsilon'_2, \ldots, \epsilon'_z \}$ be a sub set of $\{\epsilon_1, \epsilon_2, \ldots, \epsilon_t\}$, and $\chi(N^c(\{\epsilon'_1, \epsilon'_2, \ldots, \epsilon'_z\})) = X$, then $\bigcup_{i=1}^{z} N^c(\epsilon'_i) = \bigcup_{i=1}^{t} N^c(\epsilon_i)$. (Observation 4)

- **Similarity query**

A similarity query (k-NN query) in a P2P network can be described as follows: Given the set of data objects $X \subset R^n$ and the P2P network G, for a specific integer constant k and a given query object x_q, return k data objects $X^* = \{x_1, x_2, \ldots, x_k\} \subset X$ such that for $\forall (x \in X \wedge x \notin X^*)$ satisfies $dist(x_q, x) \geq dist(x_q, x_i) \forall i \in [1,k]$.

In the context of P2P network, the resolution of k-NN query means flooding in the whole network and making pair-wise comparison on data objects in each node. The cost of this resolution is formidably high and impractical for real applications. Therefore, alternative approaches should be devised to perform the similarity search with optimized search operations and reduced cost.

Definition 1: The Nearest-Neighbor Retrieval (k-NN)

Given an object set $X = \{x_1, x_2, \ldots, x_m\}$ and a query object x_q, the nearest-neighbor retrieval of x_q within X, denoted as $\Xi^k(x_q, X)$, is the following set:

$$\Xi^k(x_q, X) = \{x_i \mid \forall x \notin \Xi^k(x_q, X), dist(x_q, x) \geq dist(x_q, x_i) \wedge |\Xi^k(x_q, X)| = k\} \quad (1)$$

Definition 2: The range-constrained k-NN search

Given a data object set $X = \{x_1, x_2, \ldots, x_m\}$, a query object x_q, and a distance threshold d, the range-constrained k-NN search returns a set:

$$\Xi^k{}_d(X, x_q, d) = \{x_i \mid \forall x \notin \Xi^k{}_d(X, x_q, d), dist(x, x_q) \geq dist(x_i, x_q) \wedge dist(x_i, x_q) \leq d \wedge |\Xi^k{}_d(X, x_q, d)| = k\} \quad (2)$$

The significance of range-constrained k-NN search is to express the exact match k-NN queries into spatial range queries. By restricting the search range with the sphere centered at x_q with a radius d, the search cost can be drastically reduced. In addition, it can be proven that the range-constrained k-NN search returns exactly the same query result as the original k-NN search. The reason is that the data objects with closest semantic distance with x_q are located within a sphere centered at x_q, by selecting appropriate distance threshold, these data objects can be found through a range-constrained k-NN search operation.

Claim 1: Given a multimedia data object set $X = \{x_1, x_2, \ldots, x_m\}$ and a query object x_q, there always exists a distance threshold d, satisfying $\Xi^k(x_q, X) = \Xi^k{}_d(x_q, X, d)$.

Proof: For a given distance threshold d, the multimedia data object set X is divided into two partitions — the objects whose semantic distance to x_q is less than d (*i.e.* within the sphere centered at x_q with a radius d) and the objects outside the sphere. By adjusting the distance threshold, a sphere that encloses exactly the same data objects in $\Xi^k(x_q, X)$ can be obtained. In other words, given the multimedia data object set X, the distance threshold can be considered as a function whose parameter is the number of nearest neighbors. Therefore, the threshold d satisfying $\Xi^k(x_q, X) = \Xi^k{}_d(x_q, X, d)$ always exists. ■

- **Search cost**

A similarity-based k-NN query may travel several nodes before reaching the data source nodes containing the requested query result. A query may also be decomposed into multiple sub queries and forwarded to different nodes for query resolution. The data objects collected from different network branches are then merged together as the result of the original query. Therefore, we evaluate the query resolution in a P2P network from three aspects: search cost, accuracy, and system overhead. The search cost includes query response time, message complexity per query, and cache hit ratio when caches are employed. The accuracy means the percentage of the results generated by decentralized search strategy matching with the results generated by centralized search strategy. The system overhead includes the maintenance cost (messages and time) incurred in the process of adjusting the search index or cache content in accordance with the network topology and data content changes.

A desirable system should guarantee high search effectiveness with low search cost and system overhead. It should also provide scalability and robust-ness to network size and query frequency. Due to the space constraint, we focus on the aforementioned three aspects as the major performance metrics of similarity search.

3.2 Distance constrained nearest-neighbor retrieval

As mentioned before, each node contains a collection of data objects that occupy a sub region of the multi-dimensional semantic space R^n. The data objects are disseminated in the sub region with a specific density of distribution. Moreover, the sub regions of different nodes

may have overlapping, which is the main cause of resolving a similarity query on multiple nodes. In this sub section, we discuss the distribution density of the sub regions in the purpose of finding a concise representation of data content of nodes.

Definition 3: The hyper-rectangle constraint

Given a set of data objects $\boldsymbol{X}^* = \{\boldsymbol{x_1}^*, \boldsymbol{x_2}^*, \ldots, \boldsymbol{x_h}^*\}$, each object $\boldsymbol{x}$ can be represented as a vector of attributes $\boldsymbol{\varphi_x} = (\boldsymbol{\omega^1}_x, \boldsymbol{\omega^2}_x, \ldots, \boldsymbol{\omega^n}_x)$. The hyper-rectangle constraint $\boldsymbol{H}(\boldsymbol{X}^*)$ is a collection of constraints showing the n-dimensional minimum bounding region of $\boldsymbol{x_1}^*, \boldsymbol{x_2}^*, \ldots, \boldsymbol{x_h}^*$:

$$\boldsymbol{H}(\boldsymbol{X}^*) = ([min\{\ \boldsymbol{\omega^1}_1,\ldots,\ \boldsymbol{\omega^1}_h\},\ max\{\ \boldsymbol{\omega^1}_1,\ldots,\ \boldsymbol{\omega^1}_h\}]\ ,\ \ldots,\ [min\{\ \boldsymbol{\omega^n}_1,\ldots,\boldsymbol{\omega^n}_h\}, max\{\ \boldsymbol{\omega^n}_1,\ \ldots,\boldsymbol{\omega^n}_h\}]) \quad (3)$$

Definition 4: The sphere constraint

Given the aforementioned data object set $\boldsymbol{X}^*$, the sphere constraint is a binary tuple $\boldsymbol{S}(\boldsymbol{X}^*) = (\bar{x}, \boldsymbol{\Gamma})$, where $\bar{x}$ is a virtual data object whose attributes are the mean values of the attributes of $\boldsymbol{x_1}^*, \boldsymbol{x_2}^*, \ldots, \boldsymbol{x_h}^*$, and $\boldsymbol{\Gamma}$ is largest distance between $\bar{x}$ and any objects in $\boldsymbol{X}^*$.

Definition 5: The distribution density

Given the aforementioned data object set $\boldsymbol{X}^*$ locating in an n-dimensional sub region that is represented by the hyper-rectangle constraint $\boldsymbol{H}(\boldsymbol{X}^*)$, the function of distribution density within the region can be denoted as:

$$\boldsymbol{F_{dd}}(.)\text{: } \boldsymbol{R^n} \rightarrow \boldsymbol{R}\text{, which satisfies } \int_{H(X^*)} \boldsymbol{F_{dd}}(\boldsymbol{X}^*) = 1. \quad (4)$$

Claim 2: Given the aforementioned data object set $\boldsymbol{\chi}(\boldsymbol{n_i})$ of a node $\boldsymbol{n_i}$, the distribution density function $\boldsymbol{F_{dd}}(\boldsymbol{H}(\boldsymbol{\chi}(\boldsymbol{n_i})))$ can be computed as the combination of density functions of $\boldsymbol{\chi}(\boldsymbol{n_i})$ in each dimension.

Proof: As explained in definition 5, $\boldsymbol{F_{dd}}(.)$ is a distribution density function in the n-dimensional semantic space $\boldsymbol{R^n}$. The $\boldsymbol{F_{dd}}(.)$ function can be defined using a deductive way: $\boldsymbol{F_{dd}}(\boldsymbol{R^n}) = \boldsymbol{F_{dd}}'(\boldsymbol{R^{n-1}}) * \boldsymbol{f_d^j}(\boldsymbol{R}|\boldsymbol{F_{dd}}')$, where $\boldsymbol{F_{dd}}'(\boldsymbol{R^{n-1}})$ is the distribution function in the n-1 sub space, while $\boldsymbol{f_d^j}(\boldsymbol{R}|\boldsymbol{F_{dd}}')$ is the conditional distribution density in the remaining dimension. Due to the independence among the dimensions, $\boldsymbol{f_d^j}(\boldsymbol{R}|\boldsymbol{F_{dd}}')$ is equivalent to the mapping of $\boldsymbol{F_{dd}}(.)$ in this dimension, which is denoted as $\boldsymbol{f_d^j}(.)$. Therefore, $\boldsymbol{F_{dd}}(\boldsymbol{R^n})$ can be represented as the product of the density functions of each dimension, denoted as $\boldsymbol{F_{dd}}(\boldsymbol{R^n}) = \prod_{j=1}^{n} \boldsymbol{f_d^j}(\boldsymbol{R})$. For the multimedia data objects in the n-dimensional hyper-rectangle region $\boldsymbol{\chi}(\boldsymbol{n_i})$, because each object $\boldsymbol{x}$ is represented as a vector of attributes $\boldsymbol{\varphi_x} = (\boldsymbol{\omega^1}_x, \boldsymbol{\omega^2}_x, \ldots, \boldsymbol{\omega^n}_x)$, the density function of the jth dimension $\boldsymbol{f_d^j}(\boldsymbol{H}(\boldsymbol{\chi}(\boldsymbol{n_i})))$ is described as the probabilistic distribution over the interval $[min\{\ \boldsymbol{\omega^j}_1,\ldots,\ \boldsymbol{\omega^j}_m\},\ max\{\ \boldsymbol{\omega^j}_1,\ \ldots,\ \boldsymbol{\omega^j}_m\}]$, and $\boldsymbol{F_{dd}}(\boldsymbol{H}(\boldsymbol{\chi}(\boldsymbol{n_i})))$ can be computed as the product of the 1-dimensional probabilistic distribution functions.

3.3 Proposed solution

3.3.1 Search methodology

The basic idea behind the caching scheme proposed in this paper, called Semantic-Aware Caching (SAC), is as follows: Each node in a P2P network is allowed to gradually record semantic descriptions of the query results passing by it. The semantic descriptions, in the form of hyper-rectangle constraints (definition 3), characterize the content distribution in the network. Suppose node $\boldsymbol{R}$ issues a k-NN query $\boldsymbol{x_q}$. Further assume that $\boldsymbol{x_q}$ is resolved in node $\boldsymbol{S}$ through flooding. The query result from $\boldsymbol{S}$ will be relayed by a series of nodes to $\boldsymbol{R}$. At this point, the chain of nodes from $\boldsymbol{S}$ to $\boldsymbol{R}$ could partition the network into two parts. Any future semantically similar query that crosses this chain will be resolved immediately without further communication. Suppose a node $\boldsymbol{Q}$ issues a query $\boldsymbol{x_q}^*$ that is semantically similar to $\boldsymbol{x_q}$. The $\boldsymbol{x_q}^*$ is flooded in the network and encounters one of the relaying node. As a result, the $\boldsymbol{x_q}^*$ is resolved at this relaying node and consequently, the query result from source node (i.e. $\boldsymbol{S}$ or $\boldsymbol{R}$) will be sent to $\boldsymbol{Q}$. Note that in this process the broadcasting of $\boldsymbol{x_q}^*$ could be restricted within a section of the network. In addition, subsequent queries would partition the network into smaller groups. Thus the content-distribution information is propagated in the process of query resolution, and flooding is avoided.

3.3.2 Cache usage

- **Cache structure**

The aforementioned definitions depict the data objects as a set of points in the semantic space, whose contents can be collectively described using hyper-rectangle constraints, sphere constraints, and distribution density. Basing on this representation method, we propose to cache the constraints as the concise description of query results, with the aim of increasing cache hit ratio and reduce query resolution cost.

Logically, the local cache of a node $\boldsymbol{n_i}$ is divided into a set of cache entries — each entry indicates one or multiple nodes in the network. A cache entry is a tri-tuple (*hit region*, *miss region*, *node list*). The hit region is the constraint-based description of resolved queries, which can be considered as the hyper-rectangle sub space covering the data points of earlier query results. The miss region shows the unresolved queries, which can be represented as the sphere sub space where no query results

are found. The node list shows the nodes whose data contents can be characterized by the hit region and the miss region. Figure 2 illustrates the cache structure.

Physically, we store the content descriptions through paging. The constraints and distribution density, in the form of polynomial inequations, are stored in one or multiple linked pages. Note that there are three relationships between the sub spaces described by the constraints: enclosure, overlapping, and isolation. Basing on these relationships, a hierarchical indexing structure can be built on the cache entries, which maintains the semantic descriptions as well as the physical storage information for every cache entry.

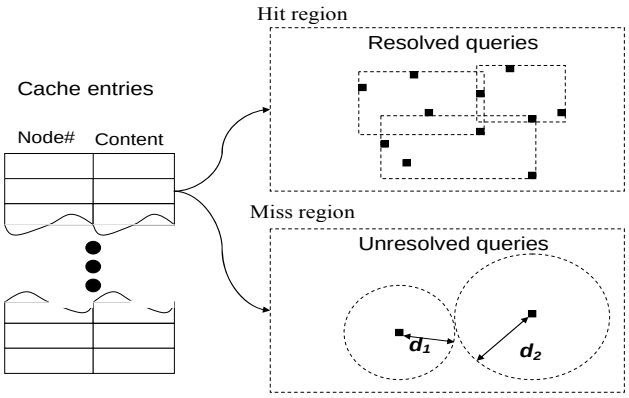

Figure 2: The cache structure.

- **Cache management**

In the aforementioned query resolution process in figure 2, for the simplicity of presentation, we divide the messages into two categories: control messages and data messages. The control messages are of small size and used for transmitting the concise representation of queries and data replies — semantic vectors. The data messages are used for the transmission of large-size data objects such as multimedia files. The cache management mechanism consists of three phases — initialization, cache checking, and cache replacement — which cooperatively reduce the network traffic by avoiding unnecessary data messages.

Initially, the local caches are empty and every query is flooded in the network. When a data source node is found, the query is resolved and the result is forwarded back through a collection of relaying nodes, where the data content description (i.e. the constraints and distribution density) is cached for future query processing.

Flooding can be avoided through analysis of cache content. When receiving a query $\boldsymbol{x}_q$, a node will compare the query with the hit regions and the miss regions in its local cache. If the query is enclosed in the miss region of a node $\boldsymbol{n}_i$, then label $\boldsymbol{n}_i$ as "irrelevant" node to the query, and avoid forwarding the query to $\boldsymbol{n}_i$. Considering the semantic locality of queries, the miss regions generated by earlier queries will block the broadcasting of later queries with similar semantic contents.

Due to the limitation of cache size, the local cache may not have enough space for new data results. The replacement policy of SAC can support various cache granularities according to different quality of service (QoS) requirements. Data objects that are not visited frequently (i.e. lower QoS requirement) can be represented using a coarse description, which is represented as a larger semantic sub space with sparse distribution density. On the other hand, frequently accessed data objects are represented with finer descriptions in smaller regions with dense distribution density. In this way, different QoS requirements are met and at the same time the size of cached items is reduced.

4. Performance Study

A simulator was implemented in *ns*-2 environment (version 2.26) to evaluate the proposed semantic-based search scheme. Performance of the proposed scheme is compared against the flooding, CacheData, and CachePath as described in [3] based on various metrics, such as accuracy, search cost, and system overhead.

4.1 Experimental setup

The evaluation consists of a series of experiments conducted using both real world data sets and simulated environments. The purpose of experimental analysis using both real data and synthetic data is to evaluate the performance gain of the proposed caching scheme in the context of different network scales and data sources. Our comparative analysis is based on various performance metrics such as accuracy, search cost, scalability, and physical characteristics of the mobile nodes.

The real world data set comprises up to 3000 images (435 features) of 100 semantic categories from the Corel dataset, which is similar as the dataset used in [12]. 2000 images in the test bed are used to train a LPP subspace learning module that partitions the semantic space into 100 orthogonal regions, and the remaining 1000 images are used as test images for *k*-NN queries. To examine the effect of semantic locality, we randomly select 200 images as a "hot" dataset, and let the probability of queries on the hot dataset abiding a given parameter.

The simulated environment consists of up to 512 nodes where nodes join/leave the network according to a Poisson process during a period of 5000 seconds. The synthetic data object set comprises up to 65,536 data points whose semantic vector values are assigned by a random number generator abiding by normal distribution in the interval [0, 1). A node has a random number of data objects from 0 to 8,000, each object being a 256-dimensional point in $[0, 1)^{256}$. Each node in the simulator randomly selects waypoints within a 2000m*600m flat area. The node density can be adjusted by changing the number of nodes in the flat area. The node moving pattern follows the random waypoint movement model [2]. The query generation time follows the exponential distribution, which is similar to the previous work [3]. The access pattern in the queries follows *Zipf-like* distribution, which is widely used in modeling non-uniformly distributed queries [4,10,15].

4.2 Experimental result

- **Accuracy**

In terms of accuracy, we ran our simulator on 3,000 real images randomly scattered on 128 nodes. We set the number of *k*-NN search to 10. For the three search schemes (SAC, CachePath, and flooding), we varied the number of visited nodes from 2 to 96 to evaluate the matching percentage of returned content-similar data objects in comparison with the centralized search results based on latent semantic analysis [17]. Figure 3 illustrates the trends of matching percentage with different numbers of visited nodes. For instance, over 80% of the top 10 images returned by SAC match with the centralized search results when 48 nodes are visited, under the similar condition the CachePath and flooding strategy only retrieves 50% and 20% of the top 10 images. The superior performance of SAC in contrast with the other two strategies stems from its effective exploitation of the semantic locality. The search scope is restricted to a few nodes that are semantically most related with the query, hence SAC can achieve higher accuracy by accessing only a smaller number of nodes.

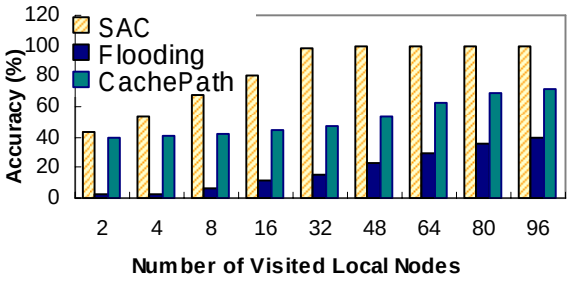

Figure 3: Impact of the number of visited nodes.

- **Search cost**

Cache hit ratio is an important metric for evaluating the performance of caches. Traditional caching schemes rely on large caches and complex replacement policies to achieve high hit ratio. In contrast, SAC employ constraint representation and distribution density to describe a collection of data objects, which increases hit ratio without large cache size requirement. Figure 4 shows the minimum cache sizes required by CachePath, CacheData, and SAC to achieve the specified hit ratios. The default parameters are used in this simulation run. In comparison with CachePath and CacheData, SAC has much less requirement on cache size, and does not drastically increase its requirement as the hit ratio increases. The better performance of SAC in contrast with the other two schemes stems from its capability of exploiting the semantic locality of data objects.

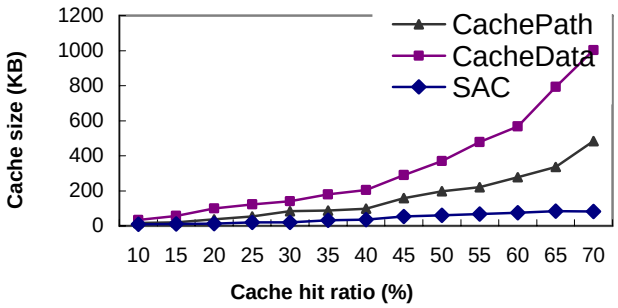

Figure 4: Comparison on cache space requirement.

In another simulation run, we examined the impact of cache size on the query response delay. The result is depicted in figure 5. As can be seen from figure 5, the CacheData scheme has the largest average query delay, especially when the cache size is smaller. As the cache size increases, more data and paths are stored, and the delay is reduced. This simulation run also confirms the capability of SAC in reducing query delay, because of its higher cache hit ratio.

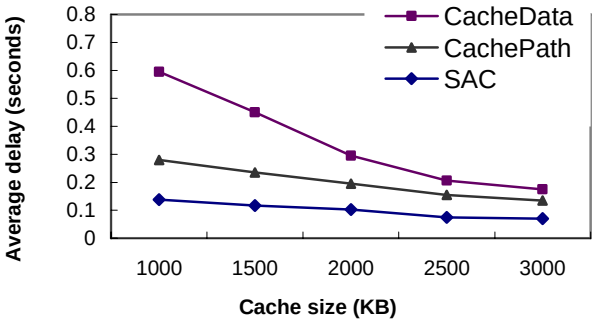

Figure 5: The impact of cache size on delay.

- **Traffic overhead**

In a P2P network, the number of messages is often used to evaluate the cost of resolving a query. The messages involved in the query processing can be divided into two categories: the control messages used for transmitting queries (i.e. semantic vectors) and the data messages used for transmitting query results (i.e. data objects). The data messages are comparatively much larger than control messages, thus causing more network traffic. The average number of control messages required to resolve a query is shown in figure 6, while figure 7 depicts the average number of data messages per query. As shown in figure 6, CachePath and SAC require more control messages than CacheData because they need to forward the query to the data source nodes. However, due to the limitation of cache size, CacheData incurs more cache misses, causing much more data messages. In contrast, SAC incurs the least of data messages because of its capability of forwarding the query only to the nodes with relevant data contents.

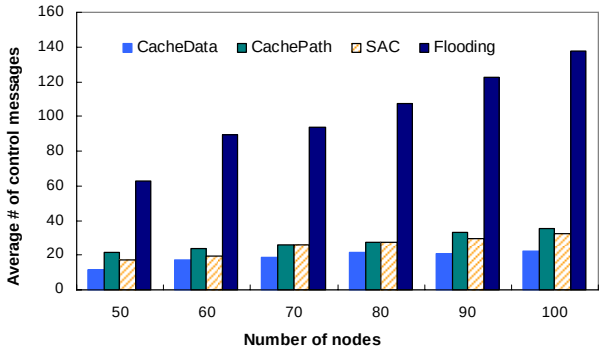

Figure 6: Average control messages per query.

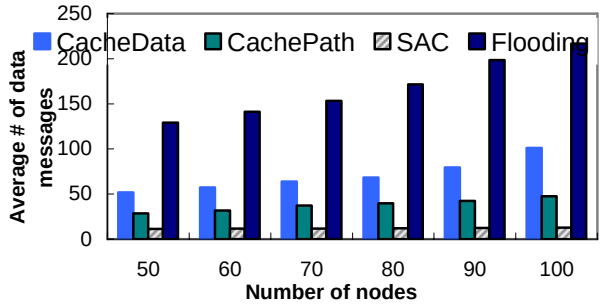

Figure 7: Average data messages per query.

- **Scalability**

In a separate simulation run, the search cost of the three caching schemes was evaluated as the network scales up. In this simulation run, 65,536 synthetic data points are randomly disseminated on 512 nodes. We varied the number of visited nodes from 32 to 256. Figure 8 illustrates the result. Similar to our earlier observation (Figure 3) one can conclude that the SAC is scalable to large network sizes and large number of data objects.

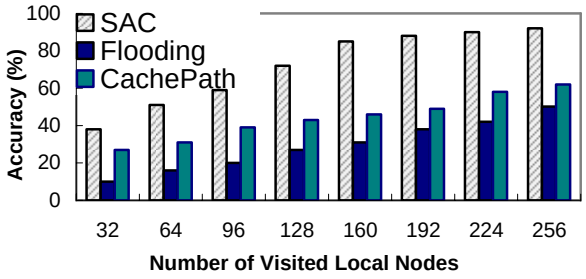

Figure 8: Impact of the number of visited nodes in large-scale network.

5. Conclusion

We proposed a dynamic semantic-aware caching scheme that facilitates content-based multimedia retrieval in peer-to-peer networks. This scheme is based on analysis of cached query results to represent the data contents in each node. It has several innovative characteristics such as content distribution representation and non-flooding query processing.

The proposed scheme makes use of the data content distribution in P2P networks to reduce the search cost of *k*-NN queries without incurring high maintenance overhead. We have studied the efficiency and effectiveness of our scheme with respect to various performance metrics — accuracy, cache hit ratio, query delay, and message complexity. Through extensive experimental analysis, we found that the semantic-aware caching methodology has the following features:

- The SAC is a decentralized non-flooding strategy facilitating content-based multimedia retrieval in P2P networks. As shown in our simulation results, it can achieve high hit ratio while visiting only a small portion of nodes.
- We employed content distribution information in the organization of cached data — the *k*-NN queries are forwarded only to the content-related nodes. As witnessed by simulation results, this approach improves the performance (accuracy, hit ratio, and query delay) dramatically relative to the traditional flooding strategy and two of the state-of-the-art caching schemes.

- Our model is dynamic and capable of self-organizing itself as the network status changes. This further offers scalability and robustness in large-scale networks.

References

[1] C. E. Perkins, E. M. Royer, S. R. Das. Performance comparison of two on-demand routing protocols. IEEE Personal Communications, 2001.

[2] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. ACM Mobicom, 1998: 85-97.

[3] L. Yin and G. Cao. Supporting cooperative caching. IEEE INFOCOM, 2004.

[4] A. Gionis, D. Gunopulos, and N. Koudas. Efficient and tunable similar set retrieval. Proc. ACM Sigmod 2001.

[5] B. Sarwar, G. Karypis, and J. Riedl. Analysis of recommendation algorithms for e-commerce. ACM Conference on Electronic Commerce 2000, pp 158–167.

[6] J. He, M. Li, H. Zhang, H. Tong, C. Zhang. Manifold-ranking vased image retrieval, ACM Multimedia, 2004.

[7] M. Swain and D. Ballard. Color indexing. Int. Journal of Computer Vision, 7(1):11-32, 1991.

[8] H. Samet. The Quadtree and related hierarchical data structures. ACM computing surveys, 1984: 187-260.

[9] A. W. Fu, P. M. Chan, Y. Cheung, and Y. S. Moon. Dynamic VP-tree indexing for n-nearest neighbor search given pair-wise distances. VLDB 2000: 154–173.

[10] K. Beyer, J. Goldstein, and U. Shaft. When is "nearest neighbor" meaningful? VLDB, 1994:487-499.

[11] N. Beckmann, H. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An Efficient and Robust Access Method for Points and Rectangles, Sigmod, 1990:322-331.

[12] N. Katayama, and S. Satoh. The SR-tree: An Index Structure for High-dimensional Nearest Neighbor Queries, Sigmod, 1997, 26(2):369-380.

[13] X. Tang and J. Xu. On replica placement for QoS-aware content distribution. Infocom, 2004.

[14] E. Tousidou, A. Nanopoulos, and Y. Manolopou-los. Improved methods for signature tree construction. Journal of Computing, 2000.

[15] Q. Ren and M.H. Dunham. Using semantic caching to manage location dependent data in mobile computing. ACM Mobicom, 2000: 211-221.

[16] T. Hara. Efficient replica allocation for improving data accessibility. Infocom, 2001.

[17] C. Tang, Z. Xu, and S. Dwarkadas. Peer-to-peer information retrieval using self-organizing semantic overlay networks. ACM Sigcomm, 2003:175-186.

[18] I. Stoica, R. Morris, D. Karger, M. Kaashock, and H. Balakrishman. Chord: A scalable peer-to-peer lookup protocol for internet applications. ACM SIGCOMM, 2001.

[19] G. Auffret, J. Foote, C. Li, B. Shahraray, T. Syeda-Mahmood, and H. Zhang. Multimedia access and retrieval: The state of the art and future directions, ACM Conference on Multimedia, 1999: 443-445.

[20] V. Dheap, M. Munawar, and S. Ward, Parameterized neighborhood based flooding for ad hoc wireless networks. IEEE Milcom, 2003: 1048-1053.

[21] B. Yang and A. R. Hurson, Adaptive content-aware multimedia clustering in wireless ad hoc networks, MoMM 2004: 35-44.

[22] A. Gionis, D. Gunopulos, and N. Koudas. Efficient and tunable similar set retrieval. Proc. ACM Sigmod 2001.

[23] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. Proc. ACM Conference on Electronic Commerce 2000 (EC'00), pp 158–167.

[24] C. Aggarwal, J. Wolf, and P.S. Yu. A new method for similarity indexing of market basket data. Proc. ACM Sigmod, 1999.

[25] E. Tousidou, A. Nanopoulos, and Y. Manolopou-los. Improved methods for signature tree construction. Journal of Computing, 2000.

[26] A. Nanopoulos, D. Katsaros, and Y. Manolopou-los. A data mining algorithm for generalized web prefetching. IEEE Transaction of Knowledge and Data Engineering, 2002.