RUTGERS UNIVERSITY

TASSL - The Applied Software Systems Laboratory

A Dimension Reducing Indexing Scheme for Guaranteed Keyword Searches in Peer-to-Peer Storage Systems

Cristina Schmidt Manish Parashar

TASSL, Department of Electrical and Computer Engineering, Rutgers University

Goal

Develop a P2P storage system that enables complex keyword searches, guaranteeing that all data elements matching a query will be found with reasonable costs in terms of number of messages and queried nodes.

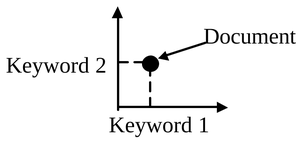
Motivation

- Recent proliferation of P2P storage systems
- The need to retrieve data stored in P2P systems, using partial keywords

System architecture – components

Locality preserving mapping

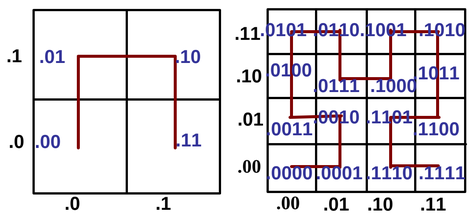
- Maps data elements to indices based on the document's keywords
- Use Space-Filling Curves (SFC)



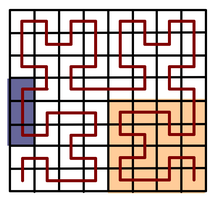
A document is a point in a multidimensional keyword space.

Space-Filling Curves

- Continuous mapping from *d*-dimensional space to 1-dimensional space
- Properties:
  - Self-similarity: can be generated recursively
  - Digital causality
  - Locality preserving: points close together in the 1-dimensional space are close together in the *d*-dimensional space



Digital causality: points on the segment contained in a cell have the first digits identical to the digits of the previous approximation of the line in that cell

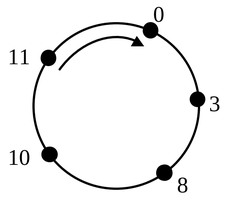


Clusters on a 3rd order space-filling curve (d = 2, n = 2). The colored regions represent clusters: 3-cell cluster and 16-cell cluster

Note: Cluster = a group of grid points that are consecutively connected by a mapping (or a curve)

Note: SFCs are used to map a document (a point in a *d*-dimensional space) to an index in an 1-dimensional space.

The Overlay Network - Chord



Each node has a unique identifier ranging from 0 to 2^m-1, arranged as a circle modulo 2^m, and stores the keys that map to the arc of the circle between itself and the predecessor node.

Each node maintains information about at most *m* neighbors in a *finger table*, the i^{th} *finger* node is the node that succeeds the current node by at least 2^{i-1}, where $1 = i = m$.

Operations

- **Node Joins**: cost $O(\log_2^2 N)$ messages, N is the number of nodes in the system
- **Node Departures**: cost $O(\log_2^2 N)$ messages
- **Node Failures**: to maintain the correct state of the system, each node periodically runs a stabilization algorithm
- **Data Lookup**: cost $O(\log_2 N)$

Load Balancing

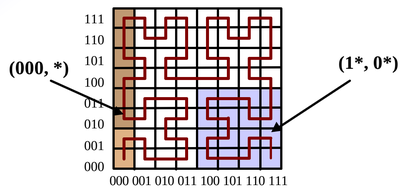
- The *d*-dimensional keyword space is sparsely populated, and data elements form clusters in this space rather than being uniformly distributed.
- Hilbert SFC-based index space preserves keyword locality => it will be sparsely populated with clusters => the nodes cannot be uniformly distributed in the node identifier space.

Load Balancing Mechanisms

- At node join: try to find the least crowded region of the system, O (*n**log N)
- Local load balancing: exchange load information with neighbors, O ($\log^2$ N)
- Virtual nodes: migrate virtual nodes from heavily loaded nodes to lightly loaded ones

The Query Engine

Query = combination of keywords, partial keywords and wildcards.
Example: (computer, net*), (comp, *)



Regions in the 2-dimensional space defined by queries (000, *) and (1*, 0*). The vertical region contains 3 clusters.

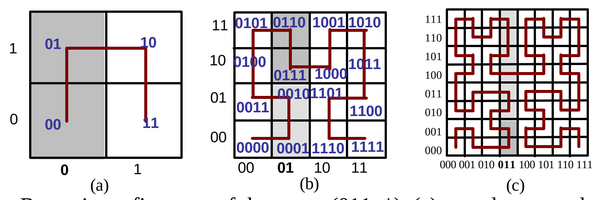
Query processing

- Translate the keyword query to relevant clusters of the SFC-based index space
- Query the appropriate nodes in the overlay network for data-elements

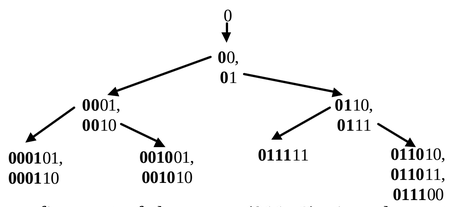
Note: the number of clusters can be large

Query optimization

- Not all the clusters that correspond to a query represent valid keywords
- Use the self-similarity property of SFC – clusters can be generated recursively, their construction can be viewed as constructing a tree

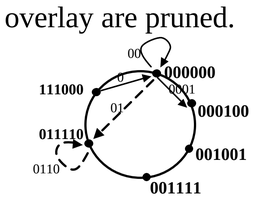


Recursive refinement of the query (011, *). (a) one cluster on the first order Hilbert curve, (b) two clusters on the second order Hilbert curve, (c) four clusters on the third order Hilbert curve.



Recursive refinement of the query (011, *) viewed as a tree. Each node is a cluster, and the bold characters are the cluster's prefixes.

- The optimization consists in pruning nodes from the tree during the construction phase. The tree is embedded into the ring topology and the sub-trees that "sink" into one node of the overlay are pruned.

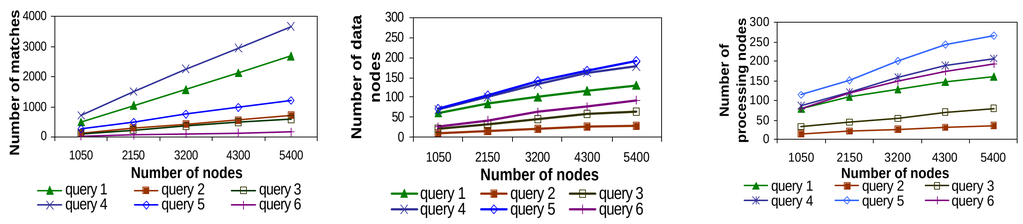


Embedding the leftmost tree path (solid arrows) and the rightmost path (dashed arrows) onto the overlay network topology.

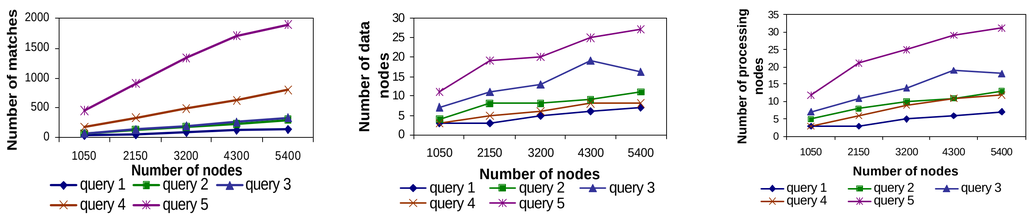
Experimental Evaluation

Evaluating the Query Engine

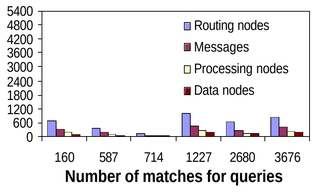
Experiment: 2D keyword space, the system size increases from 1000 nodes to 5400 nodes, and the number of keys stored increases from $2*10^5$ to 10^6.



Results for queries with one keyword or partial keyword, e.g. (computer, *).

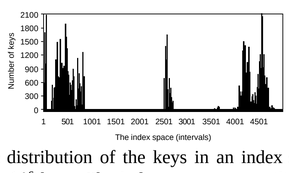


Results for queries with two keywords or partial keywords (at least one partial keyword), e.g. (comp*, net*)

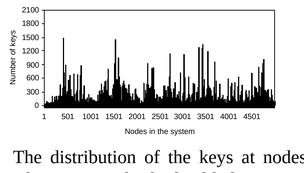


Results for a 5400 node system and 10^6 keys.

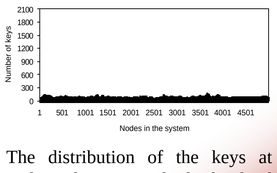
Evaluating the Load Balancing Mechanisms



The distribution of the keys in an index space with 10^6 keys. The index space was partitioned into 5000 intervals. The Y-axis represents the number of keys per interval.



The distribution of the keys at nodes when using only the load balancing at node join technique.



The distribution of the keys at nodes when using both the load balancing at node join technique, and the local load balancing.

Future work

- Improving the system's availability and response time (caching and redundancy)
- Making the system fault tolerant
- Developing new overlay topologies
- Address issues like hot-spots, ranking the documents, security