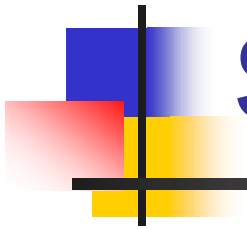
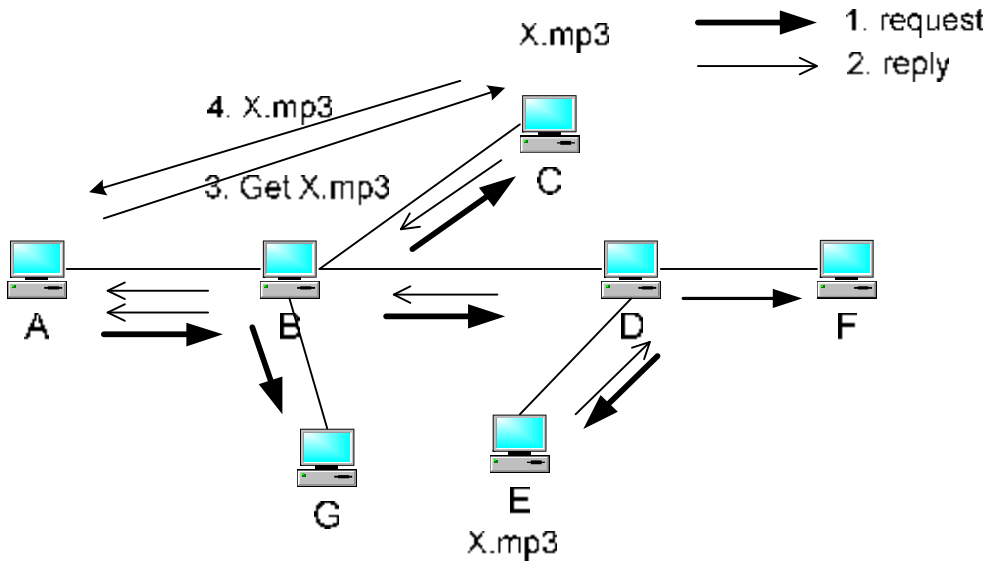


Using of Semantic Friends for Search in P2P Network

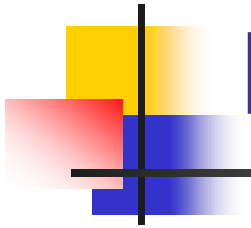
Evgeny Linsky

Unstructured P2P Network

- "Search": performed using flooding
- Unstructured: neighbors are chosen in arbitrary manner

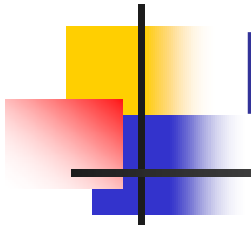


- Advantages
  - easy leave/join procedure
  - support complex search queries (a*ccc?ff.mp3)
- Disadvantages
  - scales badly, i.e. many nodes --- huge control traffic
  - search: ~n, n --- number of nodes



Problem Statement

- Goal: optimize flooding using model of requests
- General ideas
  - All documents and nodes are divided into thematic classes (by interest)
  - All documents have different popularity
- Assumptions
  - Network is stable: no one joins and leaves the network
  - Network is fully connected (one-hop network)
  - Node searches the same query periodically

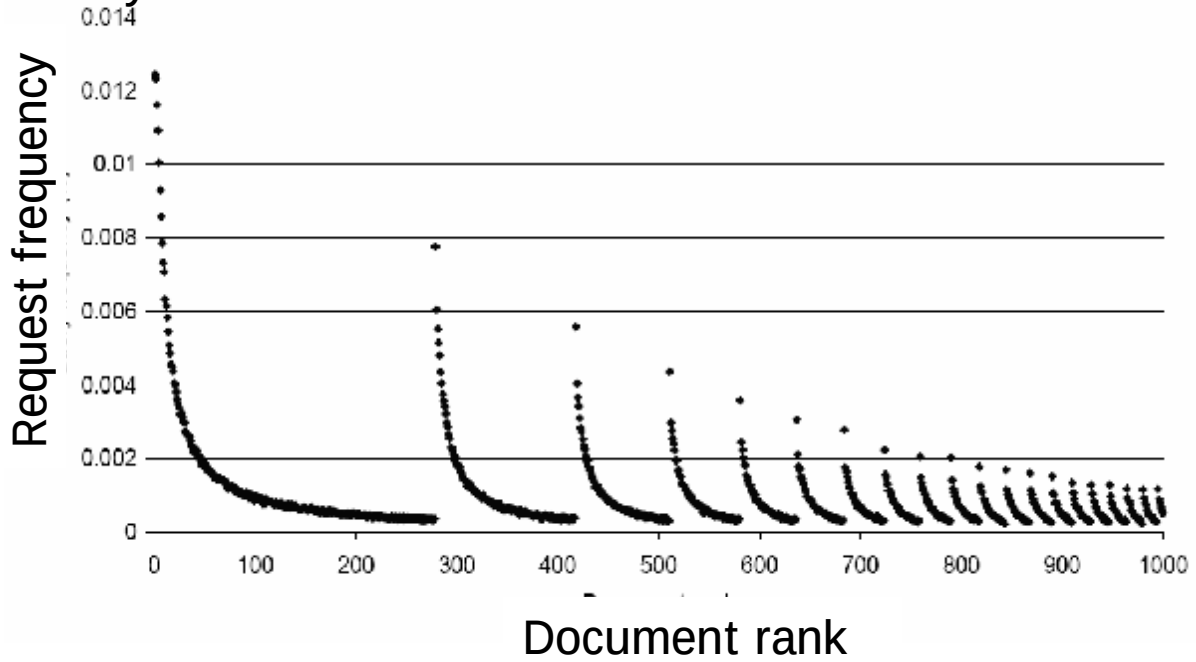


Paper Used as Base

- “Exploiting semantic proximity in peer-to-peer content searching”
  - Spyros Voulgaris, Anne-Marie Kermarrec, Laurent Massouli´e, Maarten van Steen
- Contribution
  - Model of requests
  - Algorithm for the given model

Model of Requests

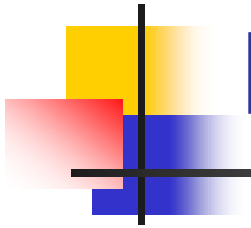
- Ideas
  - Documents (d_i) are divided into classes (c_j)
  - Documents and classes have different popularity (Zipf law)
    - $P(d_i) = 1/I$, $Size(c_j) \sim 1/j$
  - Every user is interested in one class
    - $P=\alpha$ (0.8) --- user requests document in his own class
    - $P=1-\alpha$ (0.2) --- user searches for document according to general popularity
  - Popularity is static



Base Algorithm

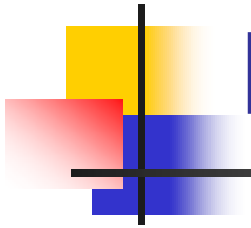
- During “warming up” use flooding
  - If document was found on some node, add this node to semantic friends list
- When the list of semantic friends is formed
  1. Send request to semantic friends
  2. If this search fails, use flooding

- List of semantic friends has finite size
- List updating policy --- modification of LRU



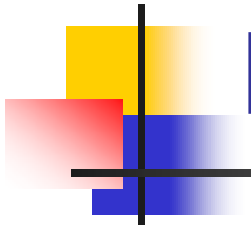
Base Algorithm

- Disadvantages
  - Performance criterion – Hit Ratio of semantic friends list
  - Maximization of Hit Ratio does not have big impact on search delay
    - Requests to low probability class are rare
    - Increase of Hit Ratio for this class does not influence on total overheads
- Our goal: minimize average number of transmissions for one served request



Proposed Solution

- FileList Algorithm
  - Node, which successfully served the request, sends FileList in addition to requested file
  - FileLists are stored and used during search in semantic friends
  - Advantages
    - Lead to decreasing of average number of transmissions for one served query
    - Adaptable to changes in request model
  - Disadvantages
    - Storage and analysis of large data arrays is hard for restricted devices => limitation of the list size



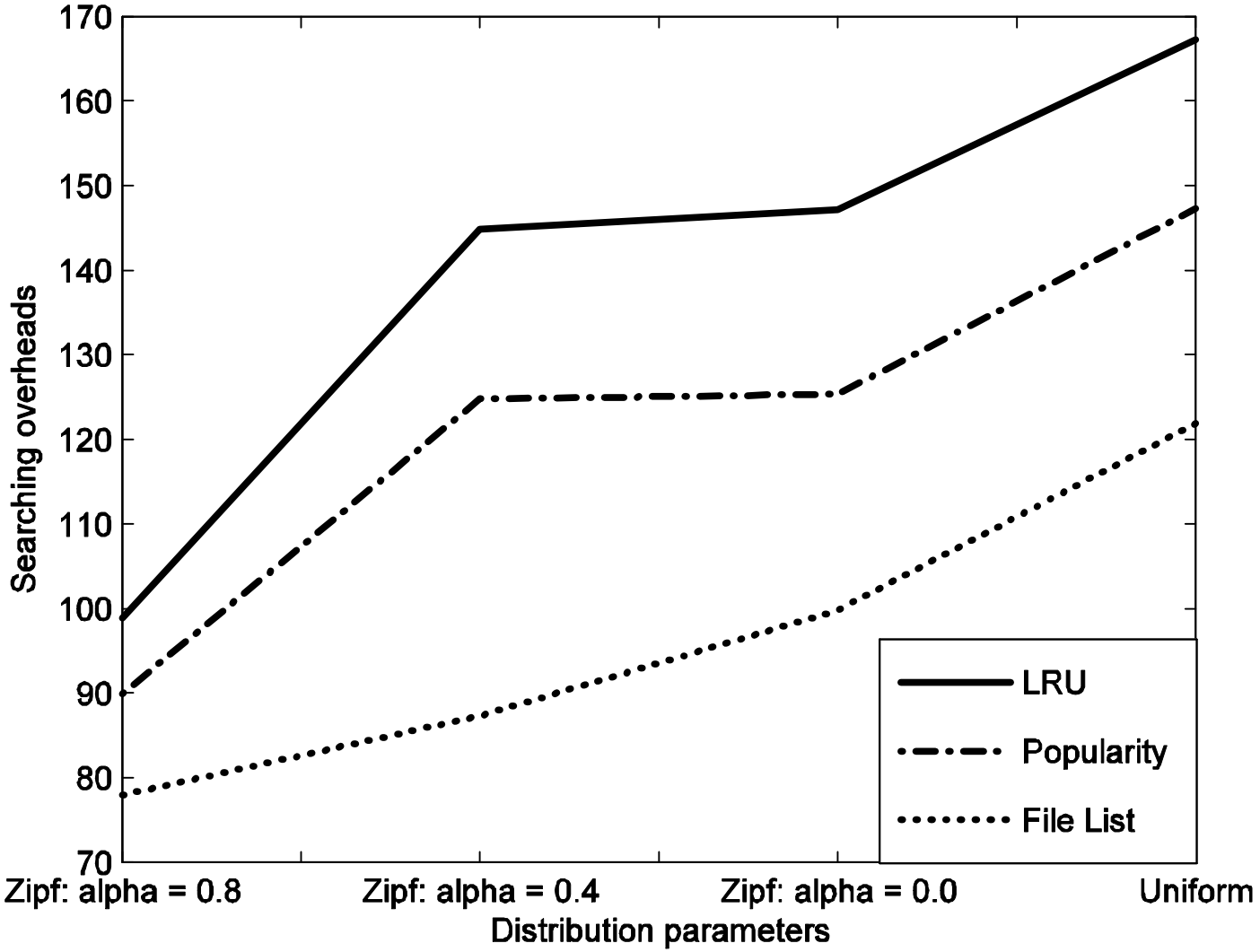
Limited FileList

- Node collects statistics about requests trying to reconstruct request model
  - Node estimates probability of documents in its own group
- Semantic friends list management policy
  - Include node, only if its FileList maximizes the future successful search probability
  - Probability is calculated using described above estimations

Simulation Results

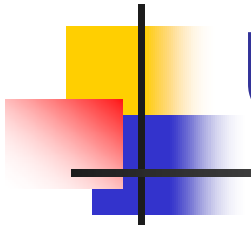
Algorithm name	Searching overhead
LRU	100
Popularity	90
FileList	78

Simulation Results



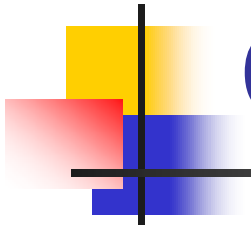
Next Step

Service Discovery in Smart Spaces



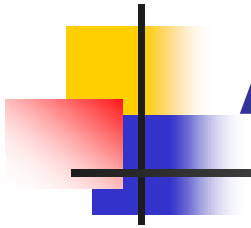
Ubiquitous Computing

- Many local wireless networks
  - including multi-hop
  - including mobile and infrastructureless
- Users roam between such networks
- Goal: find services in local proximity (Resource Discovery)
  - Closest free printer
  - Free parking slot
  - Web-service, which could convert RTF to PDF
  - Shared collection of mp3 files



Current Solution

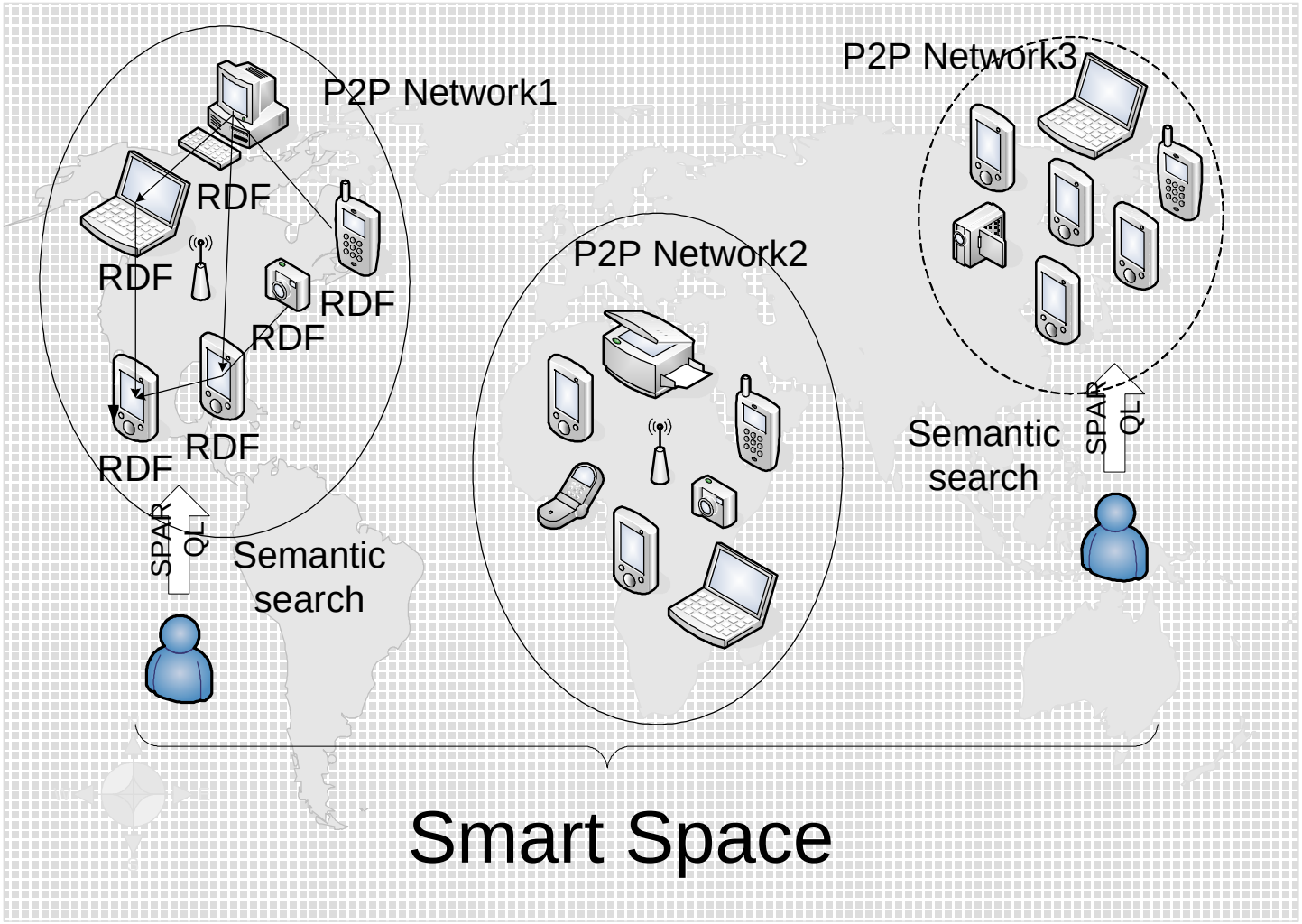
- Documents and services are described using natural language
- Dedicated server scans the network indexing the documents
- Users sends requests to dedicated server
- Requests are formulated using natural queries
  - Server uses complex algorithms for finding relevant documents
  - Results are analyzed manually by human

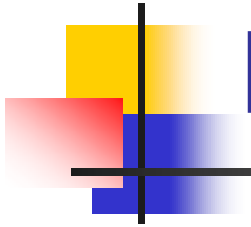


Approach

- Documents and services has formal description on RDF language
  - Simplify relevancy analysis
  - Results could be interpreted by program
- No dedicated server
  - Single point of failure
  - Additional overheads for maintaining infrastructure
- Search is implemented as distributed algorithms, e.g. using flooding

Smart Space





Problem Statement

- Flooding search
  - Large searching overheads
  - Large delay
- Optimization idea
  - Directed search
    - Request is rebroadcasted only to selected neighbors
    - Selection is done using history of previously performed requests
  - Disadvantages
    - Not all possible documents can be find
    - Tradeoff: search delay vs QoS
    - QoS = (Number of Results)/(Number of matching documents)