Peer-to-Peer Systems

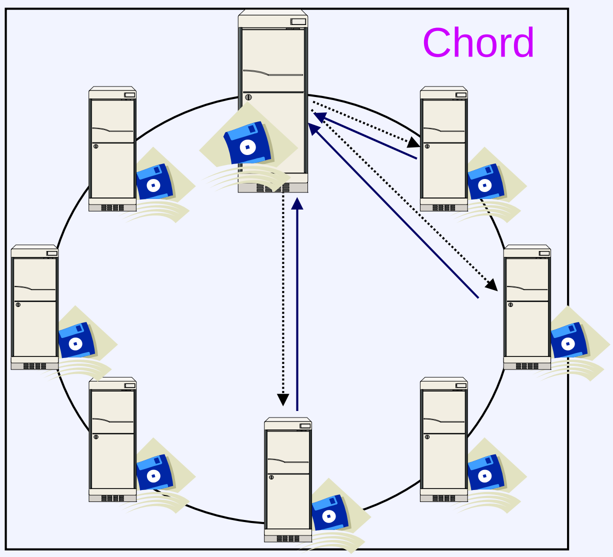
Searching

Michael Welzl michael.welzl@uibk.ac.at

DPS NSG Team http://dps.uibk.ac.at/nsg
Institute of Computer Science
University of Innsbruck, Austria

Applications and Issues

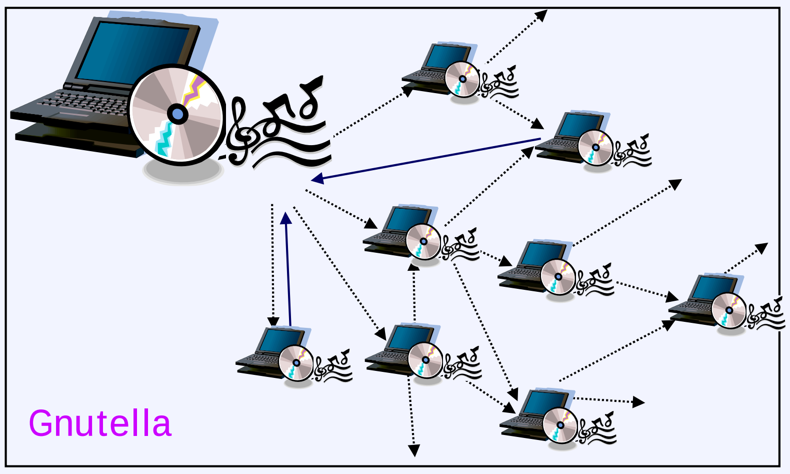
- P2P Lookup by unique ID (Identifier)
  - Distributed file systems
  - Instant messaging
  - PGP (Pretty Good Privacy)
  - P2P backup services



- Distributed Hash Tables (DHT) very suitable for lookup, however, not suitable for distributed search

Applications and Issues /2

- P2P Search by keyword or meta-data
  - File sharing
  - P2P trading
  - Classified ads
  - Grid resource discovery
  - Knowledge marketplaces



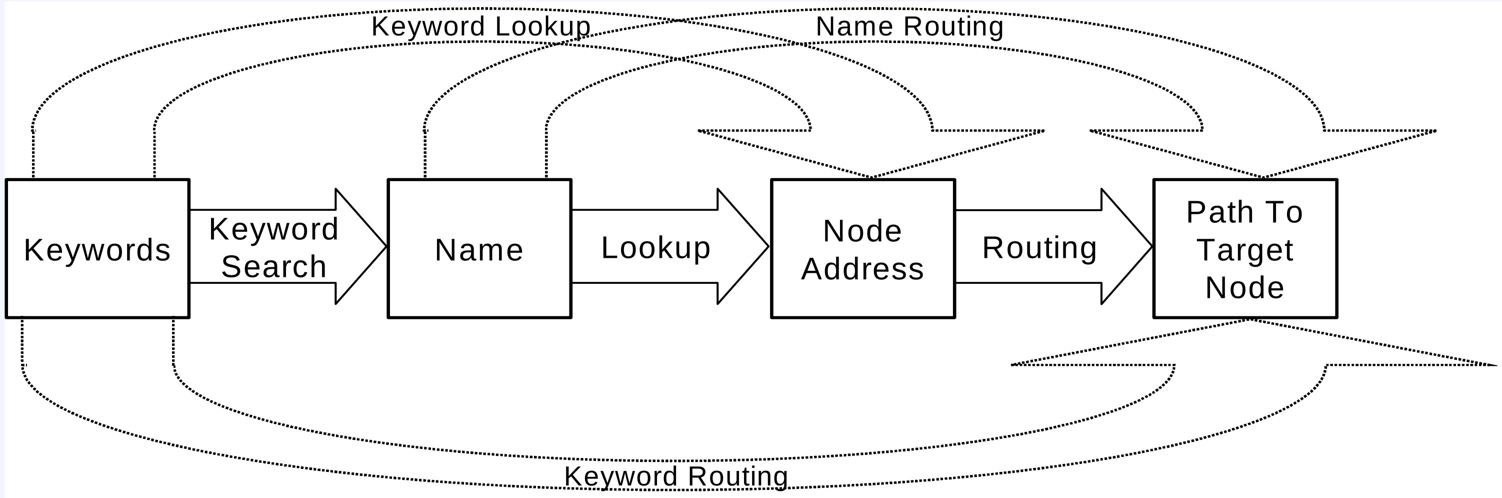
Gnutella

- Flooding approach suitable for distributed search, however, not scalable
  - Although improvements exist:
    - Query Routing Table (QRT) from leaf peers to ultrapeers lets ultrapeers limit query traffic to leaf peers; QRT compressed by hashing
    - Dynamic Query Protocol (extension of Query Routing Protocol (QRP) adjusts TTL based on popularity of search terms (number of received results)

Search and Lookup — Definitions

- Keywords
  - Simple: One or more terms appearing in the content desired
  - Complex: Content and resource meta information based on Attribute Value Pairs (AVP), *e.g.*, Resource Description Framework (RDF)
- Names
  - Unique IDs or file names, *e.g.*, Uniform Resource Locator (URL)
- Search
  - Refers to a wide range of operations and values stored in the network
    - Uni- and multidimensional search
    - Full-text search
    - Aggregate operations
- Lookup
  - Refers to finding the node hosting data for a particular ID

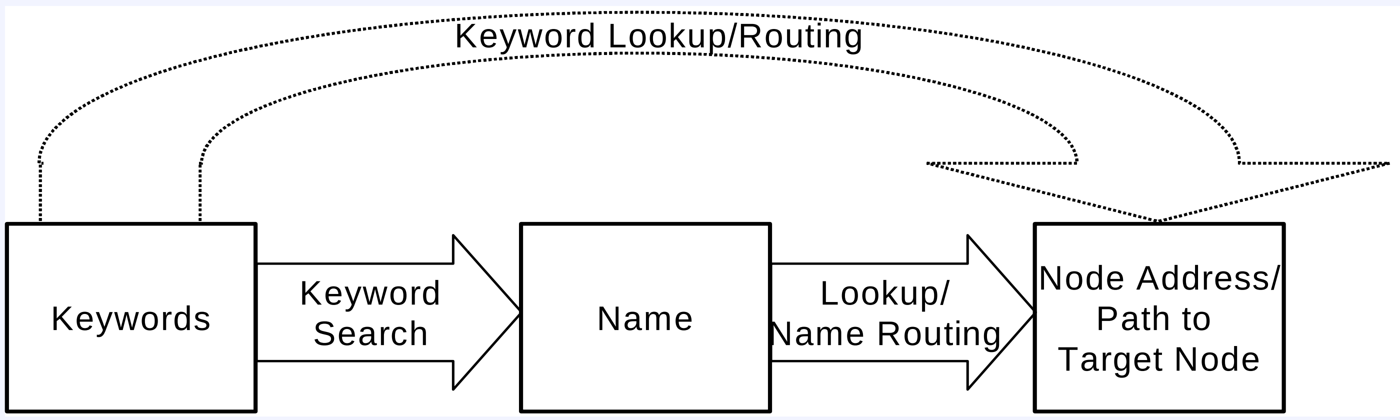
Search and Lookup — Distributed Systems



- Search disaggregated into a series of mappings
- Integrated (shortcut) approaches possible

Search and Lookup — P2P Systems

IN OVERLAY NETWORKS



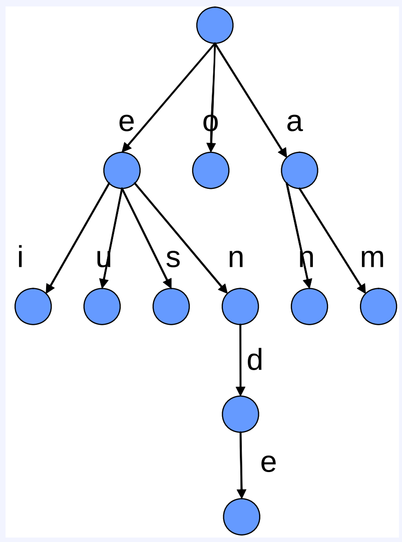
Evaluation of Functional Design Options

- Two major design options
  - Integrated keyword routing *versus*
  - Separate keyword search and name routing
- Integrated keyword routing superior choice for P2P
  - More efficient
  - Dynamic re-routing based on keywords reflects the fast changing nature of P2P networks
  - Reasons to decouple names and addresses as in the web not applicable for P2P
    - No hierarchical ownership structure available that should be reflected in the name space (to allow for delegation and browsing)
    - No slowly updated centralized search engines requiring a separate, faster name resolution system to allow for network changes

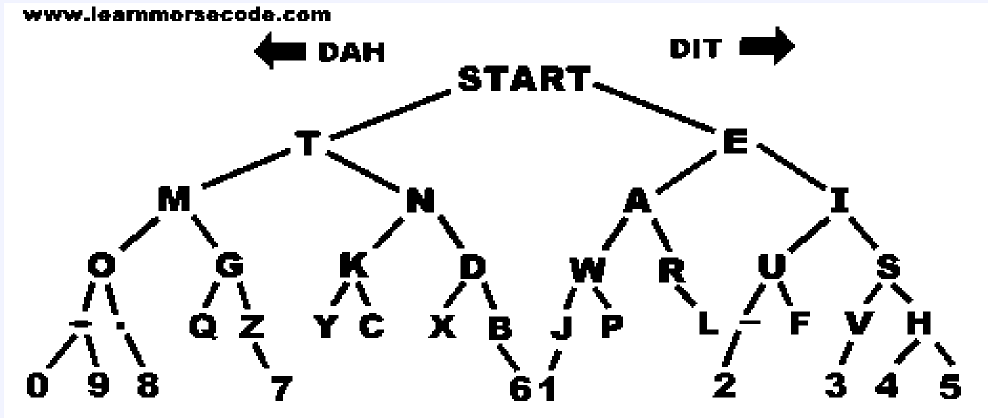
P-Grid

- Karl Aberer, P-Grid: A Self-Organizing Access Structure for P2P Information Systems, LNCS 2171, 2001, 179-194
- Problem addressed
  - DHTs erase data relationships
  - Hashing works against locality
- Goal
  - Structured overlay network
  - Like a tree
- Basic principles
  - Recursively separate the key space in partitions with
    - approximately equal number of keys (data pointers)
    - Approximately equal number of peers
  - Recursive partitioning generates tree structure
    - In fact, a TRIE...

TRIE = „reTRIEval TREE“

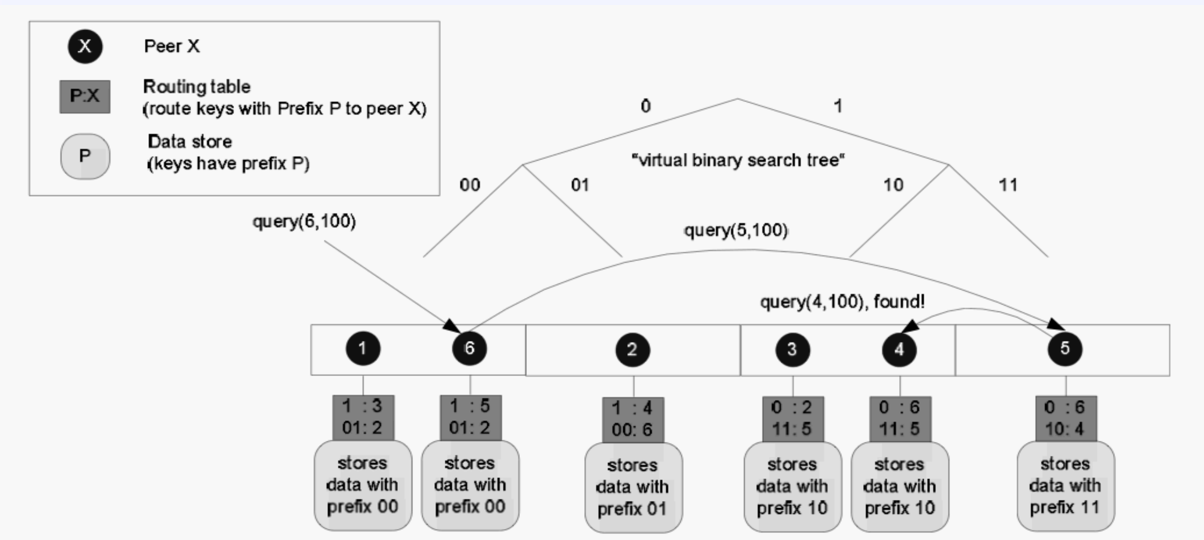


- Trie (pronounce: „tree“)
  - is a tree
  - Used for storing or encoding of text
  - Effficient prefix based search
- Structure
  - Edges labeled with letters / symbols, nodes carry words / data
- Searching
  - Follow tree structure according to letters in the text
  - Node found when word ends
  - Each node is a word (and vice versa)



P-Grid Structure

- Peers are assigned to leaves of a binary tree
- Each peer has a pointer to at least one peer in the neighboring subtree
  - Similar to Pastry/Tapestry
- Data key is binary sequence
  - Data mapped to leaves of the tree
- Number of peers in neighboring subtrees balanced according to data key
  - Achieved via pairwise interaction of peers



Balancing

- Dynamic path length
  - Subtree partitioning stops when there is only one peer left in it
- Number of peers in subtree proportional to number of data entries
- Joining
  - New peer initiates interaction with randomly chosen peers
  - New peer selects subtree with a certain probability, depending on
    - state of peers
    - amount of data they keep
  - Then it does a depth search on that subtree
- Search complexity: O(log n)

P-Grid Properties

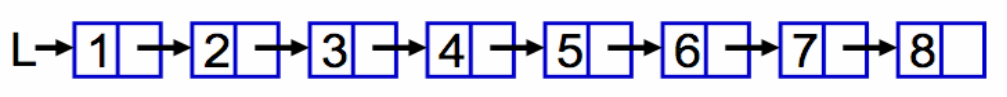
- Local load balancing
  - Peers are reassigned based on their load
  - Hence, tree structure adapts to load
- Dynamic addresses
  - Peer addresses change as they are reassigned in tree
- Decentralized trust management
  - Algorithm based on self-organization (as in unstructured P2P systems)
- Updates
  - Relies on epidemic distribution of information (Rumor Spreading)
  - Maintains consistent view of the tree

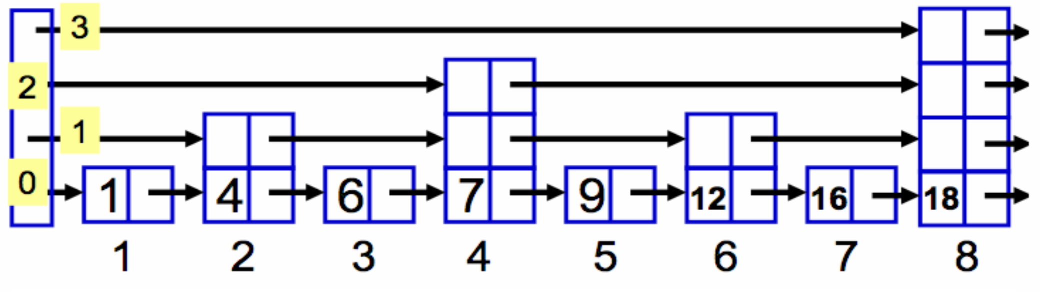
Skip-Net

- J. Aspnes and G. Shah. Skip graphs, 2003
- Harvey, Jones, Saroiu, Theimer, Wolman, SkipNet: A Scalable Overlay Network with Practical Locality Properties 2003
- Notions
  - Data assigned to peers along the ring in an ordered fashion
  - Node ID used as random bits in Skip-Graph
  - No complex self-organization (balancing) foreseen
- Based on Skip-Graph
  - Based on Skip-List
- Diameter, degree, searching: O(log n) with high probability

Skip-List

- Additional links added to a simple linked list
- Enables efficient search
- Level of link given by coin flip
  - X times head in series ⇒ level = X
- Search complexity: O(log n)

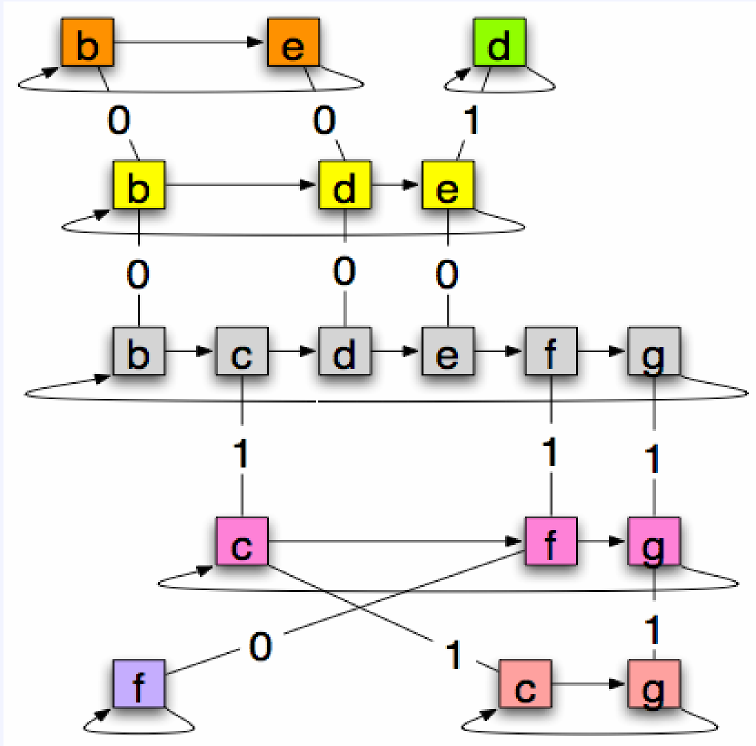




Source: Lecture of Petra Mutzel, Univ. Dortmund

Skip-Graphs

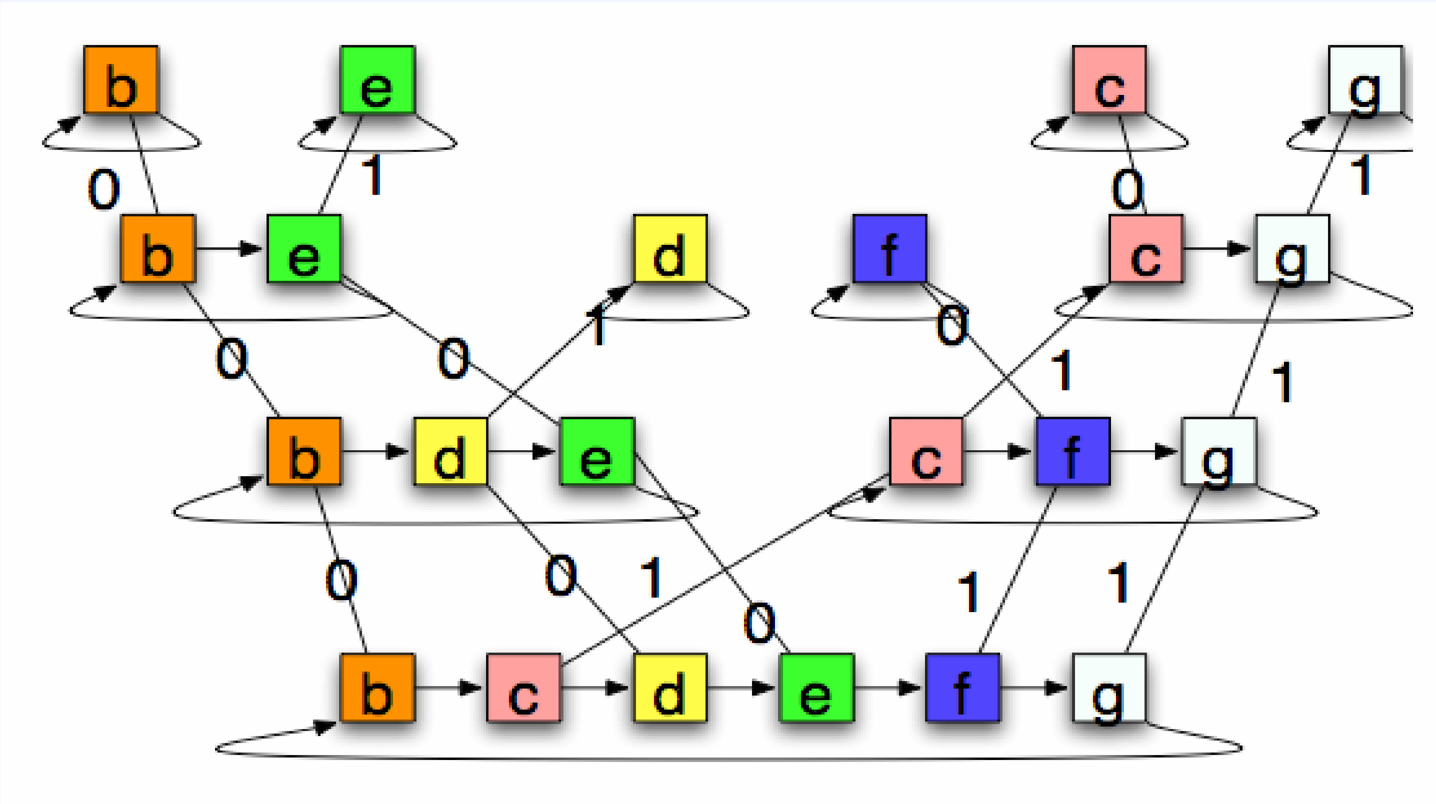
- Idea
  - The “losers” of coin flipping form a list of their own
- Properties
  - Higly resilient
    - Large number of nodes can be removed before network is partitioned
  - Diameter, degree: O(log n) with high probability
  - Data order preserved



Skip-Graphs

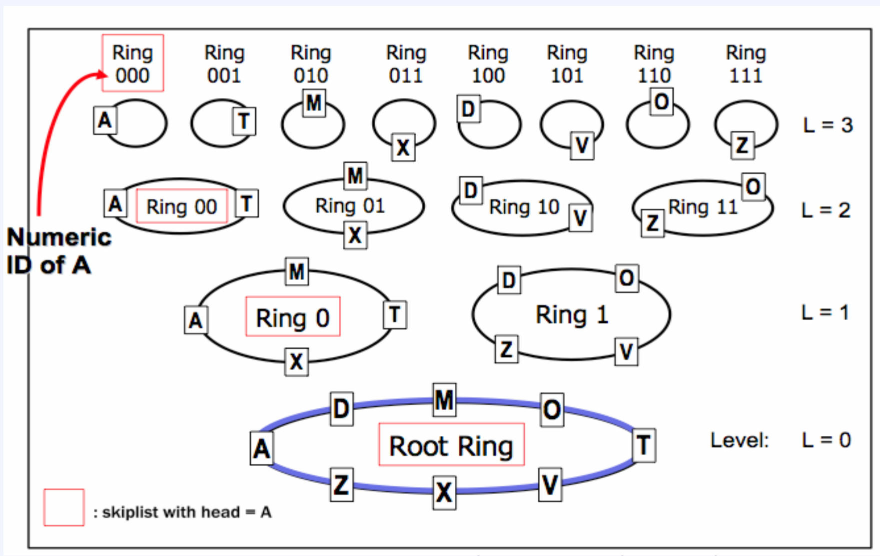
- Notions
  - Node name (name-id) used for ordering peers
  - Node ID (num-id) used as random bit in Skip-Graph
- Search for a node name (name-id)
  - Select the longest-distance link which does not take us “behind“ result
- Search for a (numeric) node ID (num-id)
  - Visit neighbor until first digit matches
  - Then visit next level
  - Repeat with the next digit
- Join
  - Search for appropriate position according to node name
  - Join upper levels via coin flipping
- Number of hops/messages: O(log n) for search or join with high probability
  - If data indices are evenly distributed
  - If Node IDs randomly chosen

Example Skip-Graph



Skip-Net

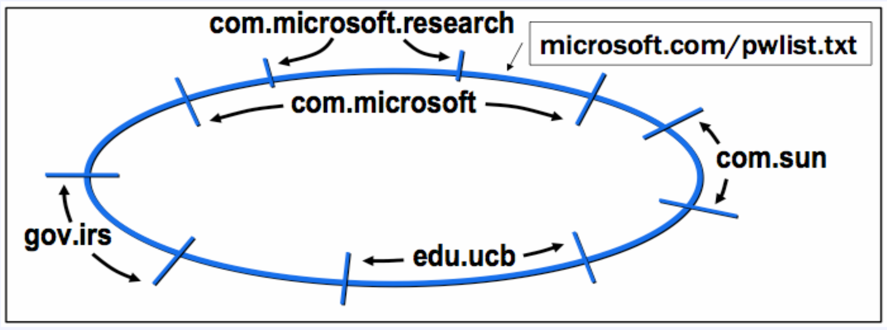
- Originally, no ring in Skip-Graphs
- Skip-Net extends Skip-Graphs
- Properties
  - Locality of data for range queries
  - Scalable
  - Routing locality
  - Control of data placement



Source: P2P Network Structured Networks,
Pedro Garcia Lopez, Universitat Rovira I Virgili

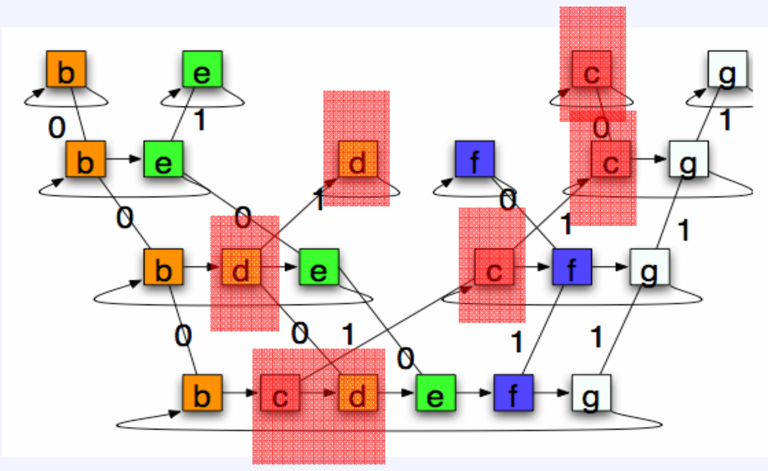
Content and Routing Locality

- Content locality
  - Given by ordering
- Data mapping
  - Data can be stored according to num-id
  - But also range of the Skip-Graph (e.g. domain)
- Routing-locality
- Example:
  - john.microsoft.com, jack.microsoft.com
  - Represented as com.microsoft.john
  - First order by com, dann nach microsoft, then john
  - DNS maps IP-address-hierarchy
  - Therefore, search path stays local (in lowest level)



Fault Tolerance

- Random faults
  - Can be compensated by using rings at higher levels
- Clustered faults (part of the network failing)
  - Few faults in the overall network
  - Affected ring is removed
  - Network stays connected in upper levels ⇒ possible to repair the problem
- All this comes at the cost of a high degree

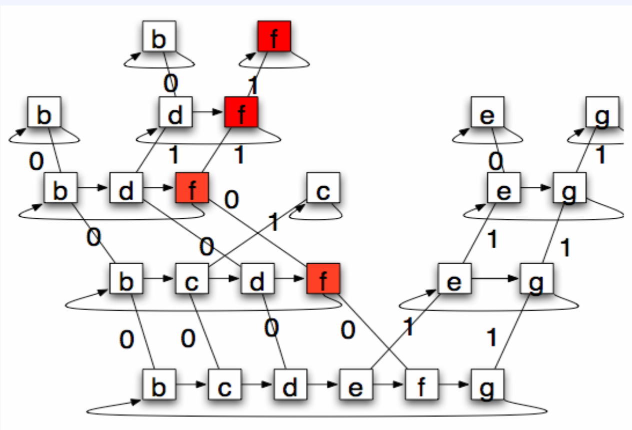


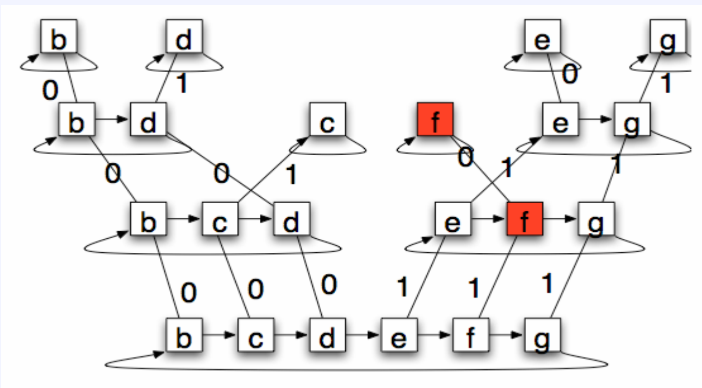
Extensions

- Increase numerical base (e.g. three-sided coin)
  - Reduces degree
  - Increases diameter
- Remove duplicate edges
  - Replace with other edges (performance improvement)
- Combination of Skip-Net and traditional DHTs: remove hash table
  - Single Overlay
    - Use numbering in Chord
  - Multiple Overlay
    - Use multiple indices (and P2P networks) at the same time

Skip-Net without randomness

- Coin flipping leads to somewhat balanced network most of the time
  - Solution: Harvey, Munro, „Deterministic Skip-Net“
- Rotation of nodes in case of imbalance
  - Remove node from subtree at level X and above
  - recursively insert it in other subtree
- Rebalances Skip-Net
  - Enables network construction without any randomness ⇒ precise (non-probabilistic) analysis possible





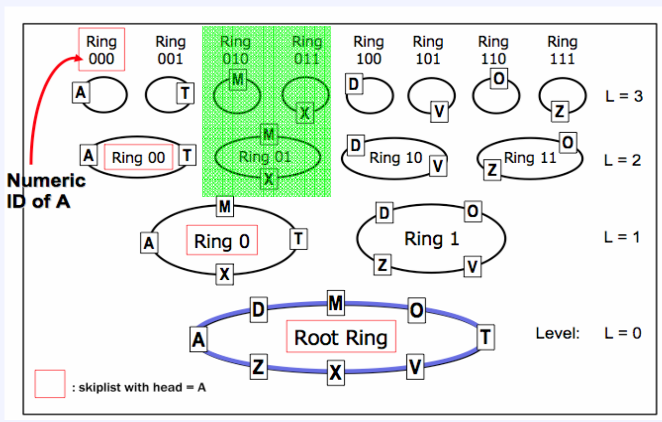
Beyond simple keyword search

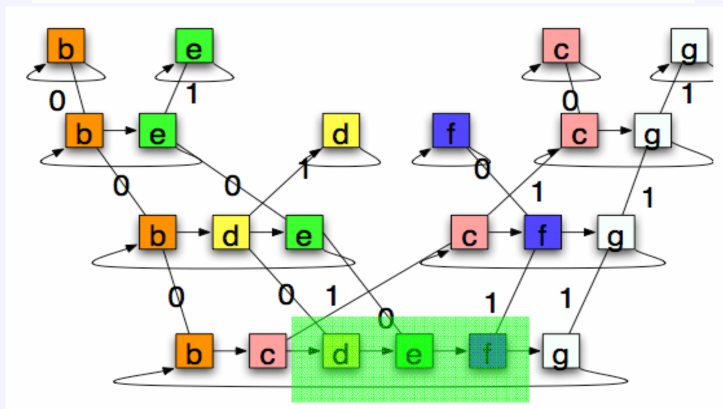
- Consider databases: simple keyword search not enough
  - Complicated queries neither supported by DHT nor by P-Grid or Skip-Net
  - Consider: SQL over P2P…
- Triantafillou and Pitoura 2003 classified queries by number of relations (single vs multiple), number of attributes (single vs multiple) and query types
  - Four major query types identified

1. Range queries
   - E.g. longest prefix match
   - Proposals to implement this on top of a DHT
   - But DHT may be a bad match; P-Grid and SkipNet proposals solve this…

Range queries in Skip-Net

- Range in Num-ID
  - Use appropriate sub-rings
- Range in Name-ID
  - Use section of lowest level ring
- Intersection of Num-ID and Name-ID range
  - Use section of lowest level ring
  - Only use appropriate sub-rings starting from this section
- Search time O(log n) for both search types





Query types

2. Multi-attribute queries
   - E.g. All students with MatrNr = 0100000 and KZ = 880
3. Join queries
   - See databases: merge tables
4. Aggregation queries
   - E.g. count or sum functions

- Other query types identified by other authors: continuous queries, recursive queries, adaptive queries…
- Each query type raises ist own issues
  ⇒ calls for different underlying P2P system

Far end of the spectrum

- Distributed Database Management Systems (DDBMS)
  - Huge body of work
  - Long history of success
  - Why not use this?
- P2P networks are…
  - larger (tens or hundreds of thousands of nodes)
  - more dynamic (node lifetime measured in hours)
  - Hence less reliable
  - Usually homogeneous – no „“mediators“ which are responsible for selecting collections, rewriting queries and merging ranked results
  - More symmetric – peers often information consumers and producers
- Gap in the design space that can be filled by P2P systems
  - Support moderate levels of data independence, consistency and query flexibility
  - Provide probabilistically complete query responses
  - Support very large numbers of low-cost, geographically distributed dynamic nodes

Peer Data Management Systems (PDMS)

- “In a PDMS, every peer is associated with a schema that represents the peer‘s domain of interest, and semantic relationships between peers are provided locally between pairs (or small sets) of peers. By traversing semantic paths of mappings, a query over one peer can obtain relevant data from any reachable peer in the network. Semantic paths are traversed by reformulating queries at a peer into queries on ist neighbors.“
- Efforts to even remove the global registry or schema and single database administrator
  - E.g. Local Relational Model (Bernstein, Giunchiglia et al. 2002):
    - Concept of peers, acquaintances (associated peers), coordination formulas (semantic dependencies between a peer and its acquaintances), domain relations (data translation rules between a peer and its acquaintances)
    - Necessary work
      - Establish acquaintances between nodes, exchange peer names, schemas and privileges
      - Semi-automatic maintenance of coordination formula and domain relations
      - New query optimizations and constraints on query propagation
      - Data dvertisement to spawn interest groups
  - Multiple other examples: Piazza, Chatty Web, Hyperion, PeerDB…

Vector model

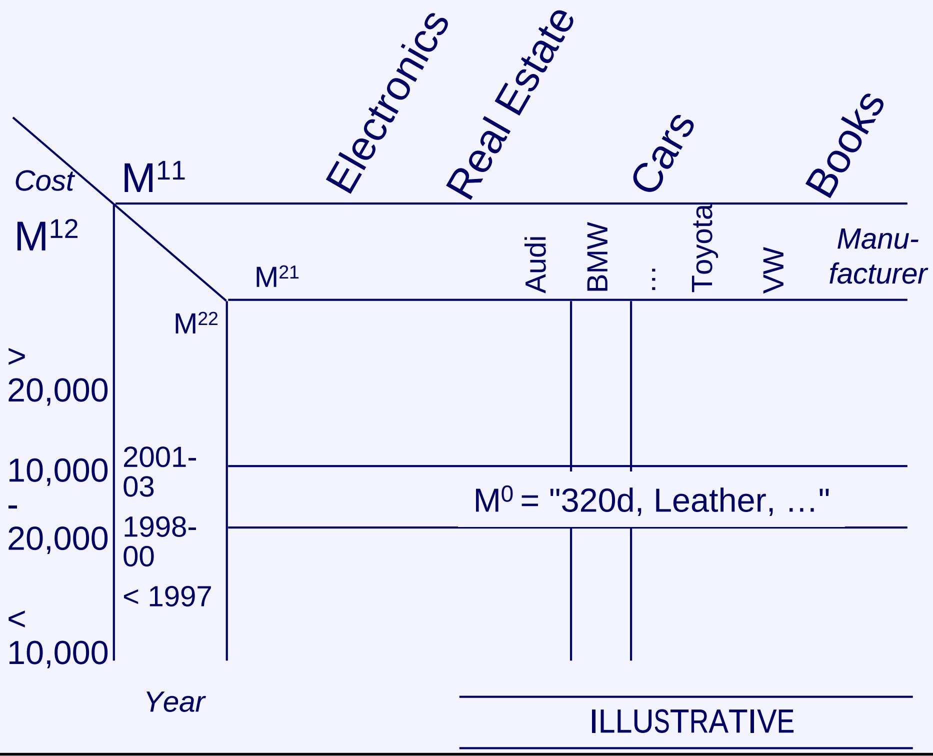
- Supports queries such as “Muhammad AND Ali NOT Boxer“
- Build term vector using a weighting scheme
  - Weighting method critical; common metrics:
    - Term frequency (how often is a term repeated in a document?)
    - Inverse document frequency (terms which appear in many documents give less information about the content of a document)
    - Document length (larger documents bias term frequencies)
    - TFIDF weighting = combination of above
    - Distributed version of Google‘s Pagerank algorithm was also proposed
- Peers can, e.g., calculate similarity between term vector and local documents, and forward to the best downstream peer(s)
  - Done in Fault-tolerant, Adaptive, Scalable Distributed search engine (based on Freenet)

SHARK: concept and meta-data structure

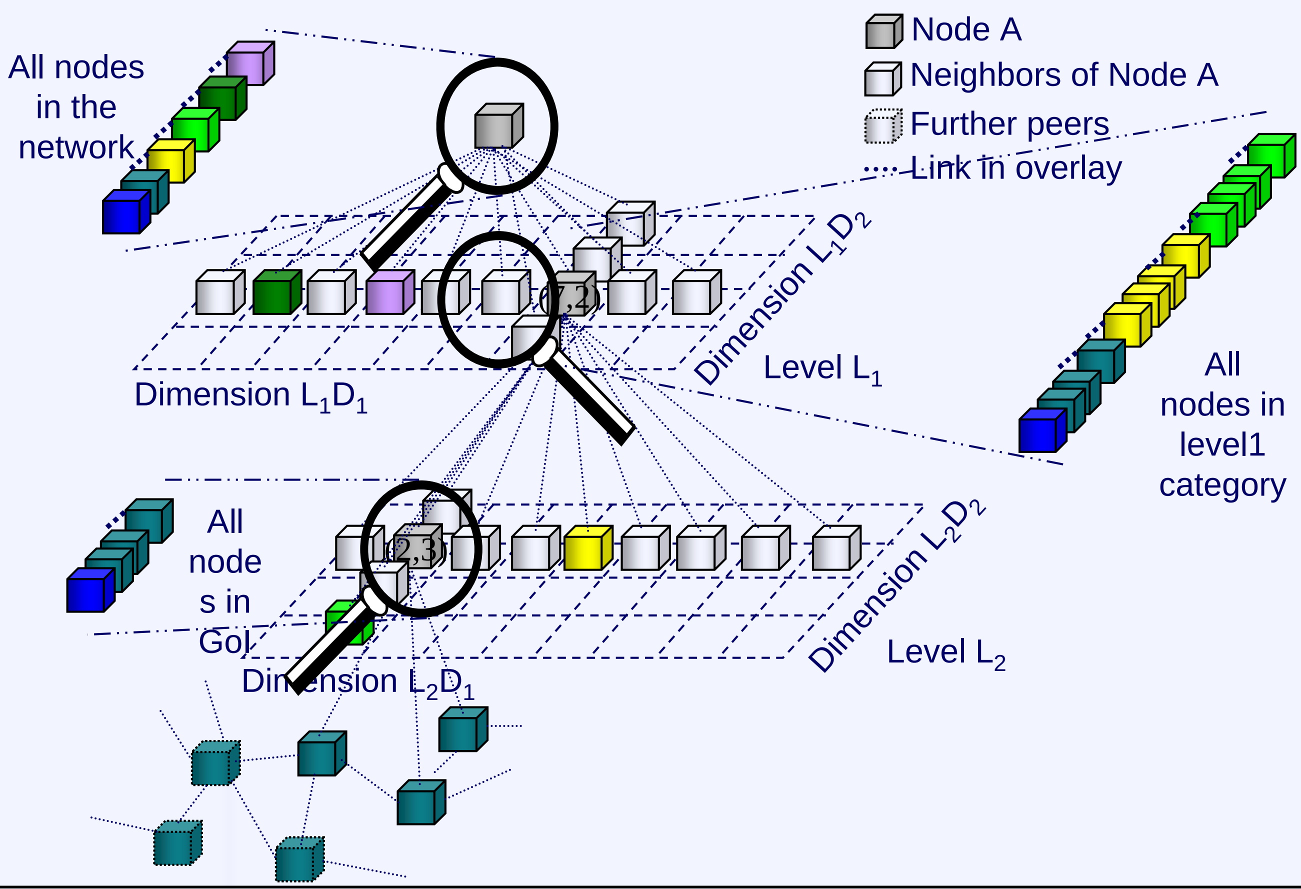
- Goal:
  - Keyword, range, and meta-data search functionality
  - Scalability
  - Multidimensional meta-data structure

$$M_j = \begin{bmatrix} M_j^{11} & \ldots & M_j^{\lambda d_1} \\ M_j^{21} & \ldots & M_j^{2d_2(M_j^1)} \\ \ldots & \ldots & \ldots \\ M_j^0 & 0 & 0 \end{bmatrix}$$

- SHARK = Symmetric redundant Hierarchy Adaption for Routing of Keywords



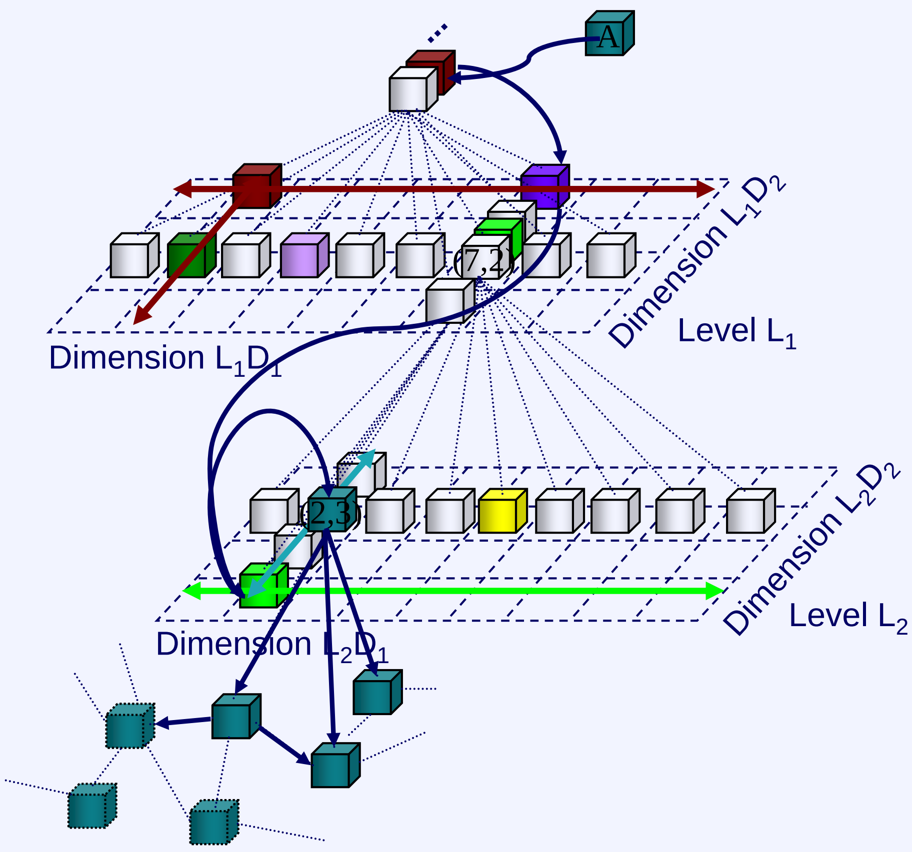
SHARK topology



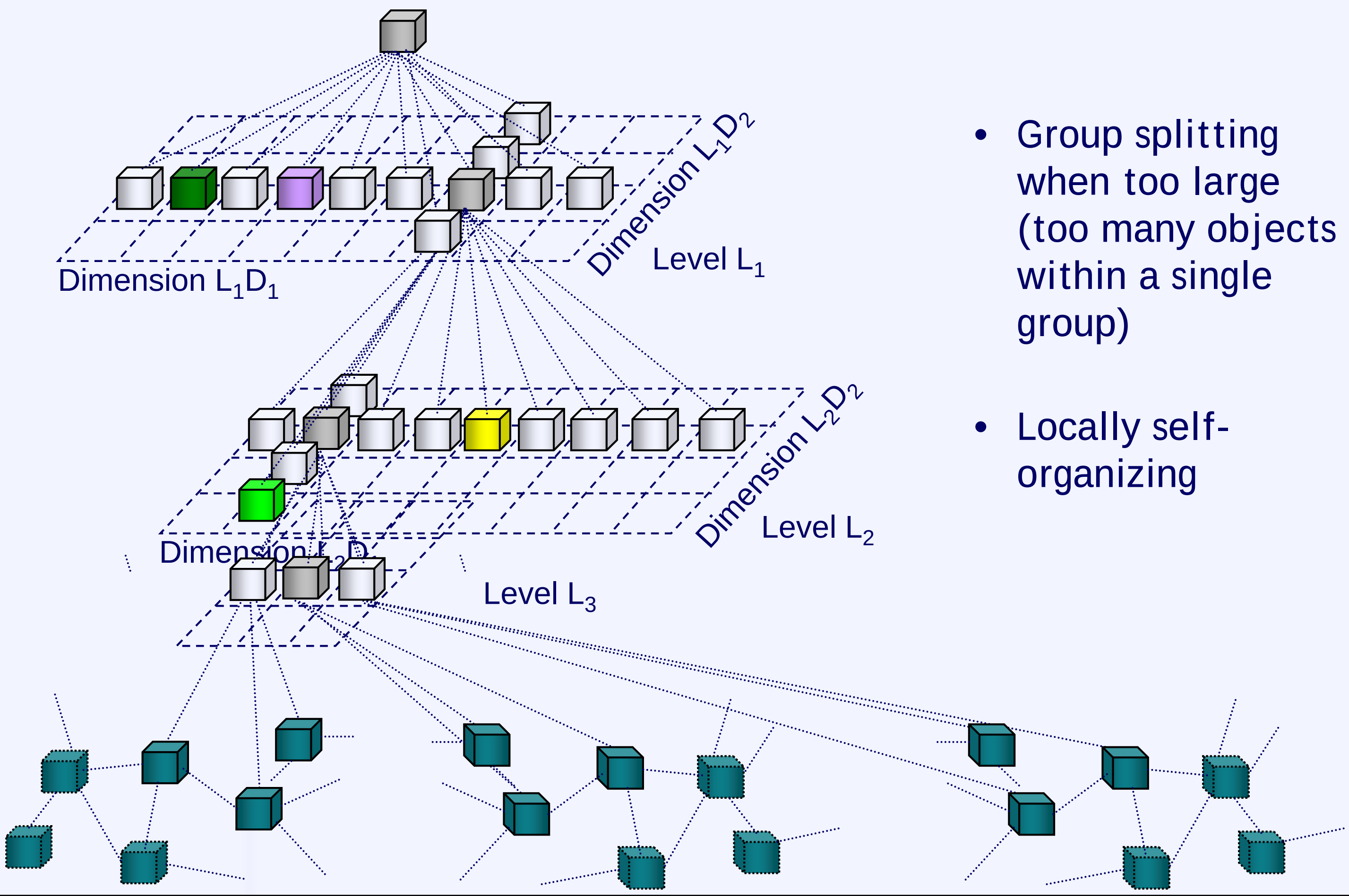
Searching, Joining, Document Insertion

- Routing by meta-data of nodes, objects, and queries

- Same principle for query, object insertion, and node insertion



Group splitting



- Group splitting when too large (too many objects within a single group)

- Locally self-organizing

Conclusion

- Usability spectrum from keyword lookup (DHT) to DDBMS
  - Or even beyond: PDMS proposals to eliminate global schema and single database administrator
  - This is where the semantic web meets P2P networks
- Suitable systems seem to exist for various points in this spectrum
  - Different query types: P2P network X ideal for query types A, B, but not C
- *The* solution does not seem to exist
- Well, gnutella and Napster can handle any query
  - But we know they have other problems
  - Some potential in hybrid solutions (e.g. mixture of unstructured + DHT)
- Still some interesting research left to be done

References / acknowledgments

- Slides from:
  - Christian Schindelhauer
  - Burkhard Stiller