Self-Organizing Information Systems

Karl Aberer

karl.aberer@epfl.ch
lsirwww.epfl.ch

Inaugural Lecture – July 8th 2002
Overview

1. P2P Systems: Decentralizing Information Systems

2. P-Grid: Efficient Decentralized Search

3. Semantic Gossiping: Emergent Global Semantics
Overview

1. P2P Systems: Decentralizing Information Systems

2. P-Grid: Efficient Decentralized Search

3. Semantic Gossiping: Emergent Global Semantics
Centralized Information Systems

- Web search engine
  - Global scale application
- Example: Google
  - 150 Mio searches/day
  - 1-2 Terabytes of data (April 2001)

Google: 15000 servers

Result: home page of Karl Aberer ...
Google Assessment

• Strengths
  – Global ranking
  – Fast response time

• Weaknesses
  – Infrastructure, administration, cost
  – A new company for every global application?
(Semi-)Decentralized Information Systems

- P2P Music file sharing
  - Global scale application
- Example: Napster
  - 1.57 Mio. Users
  - 10 TeraByte of data
    (2 Mio songs, 220 songs per user)
    (February 2001)

Request and transfer file f.mp3 from peer X directly

Napster: 100 servers
Lessons Learned from Napster

• Strengths
  – global information system without huge investment
    • exploit unused resources at nodes (space)
    • exploit users knowledge at nodes (annotation)
  – decentralization of cost and administration

• Weaknesses
  – business model: copyrighted material
  – server is single point of failure
  – therefore it can, for example, be shut down
Fully Decentralized Information Systems

- P2P file sharing
  - Global scale application
- Example: Gnutella
  - 40,000 nodes, 3 Mio files (August 2000)

Strengths
- Good response time
- No infrastructure

Weaknesses
- High network traffic
- No structured search

Gnutella: no servers

Gossiping

I have "brick_in_the_wall.mp3"
....
### Napster vs. Gnutella

<table>
<thead>
<tr>
<th>Resources</th>
<th>Napster</th>
<th>Gnutella</th>
</tr>
</thead>
<tbody>
<tr>
<td>search</td>
<td>central</td>
<td>decentralized</td>
</tr>
<tr>
<td>file exchange</td>
<td>decentralized</td>
<td>decentralized</td>
</tr>
<tr>
<td>schema</td>
<td>central</td>
<td>trivial</td>
</tr>
<tr>
<td>annotation</td>
<td>decentralized</td>
<td>decentralized</td>
</tr>
</tbody>
</table>

---

**Knowledge**

- Partially decentralized
- Self-Organizing
Decentralization – Self-Organization

- Decentralization avoids bottlenecks
  - Resources (cost, administration)
    → avoid performance bottleneck
  - Design (knowledge and control)
    → avoid single point of failure

- Full decentralization requires self-organization
  - Local information, local operation, local decision
  - Global behavior emerges from local behavior
Motivations

- **Technical**
  - Too many nodes
  - Mobile ad-hoc networks

- **Business**
  - content distribution (e.g. Bertelsmann-Napster)
  - knowledge management
  - scientific data and resource sharing (e.g. seti@home)

Original Internet designed as decentralized system
P2P ~ application-level Internet on top of the Internet
Self-Organization and Efficiency

- Self-organization
  - Can be costly if done wrong

- Example: Search Efficiency in Gnutella
  - Search requests are broadcasted
  - Anecdote: the founder of Napster computed that a single Gnutella search request (18 Bytes) on a Napster community would generate 90 Mbytes of data transfers
Overview

1. P2P Systems: Decentralizing Information Systems

2. P-Grid: Efficient Decentralized Search

3. Semantic Gossiping: Emergent Global Semantics
Overview

1. P2P Systems: Decentralizing Information Systems

2. P-Grid: Efficient Decentralized Search

3. Semantic Gossiping: Emergent Global Semantics
Data Access Structures

- Sequential search does not scale
Data Access Structures

- Search tree (Prefix Tree)

```plaintext
[Diagram showing a binary search tree with nodes labeled 00?, 01?, 10?, and 11? representing possible prefixes. The tree branches out with extra data nodes.]```
Tree-based Data Search

- For N data objects
  - Sequential search requires on average N/2 steps
  - Tree search requires $\log_2(N)$ steps

<table>
<thead>
<tr>
<th>data objects</th>
<th>sequential</th>
<th>tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>500</td>
<td>10</td>
</tr>
<tr>
<td>1.000.000</td>
<td>500.000</td>
<td>20</td>
</tr>
<tr>
<td>1.000.000.000</td>
<td>500.000.000</td>
<td>30</td>
</tr>
</tbody>
</table>
Scalable Data Access Structures

"Napster" bottleneck

peer 1 peer 2 peer 3 peer 4
Scalable Data Access Structures

- Associate each peer with a complete path
Scalable Data Access Structures

- Associate each peer with a complete path

Peer 1 and Peer 2 know more about this part of the tree.

Peer 3 knows more about this part of the tree.

Peer 4 knows more about this part of the tree.
Result: P-Grid [Coopis 2001]

- Peers cooperate in search
### Table 1. Performance comparison of Gridella and Gnutella.

<table>
<thead>
<tr>
<th>Peers</th>
<th>Gridella messages</th>
<th>Gnutella messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>20,000</td>
<td>61</td>
<td>8,744</td>
</tr>
<tr>
<td>40,000</td>
<td>63</td>
<td>26,240</td>
</tr>
<tr>
<td>60,000</td>
<td>65</td>
<td>26,240</td>
</tr>
<tr>
<td>80,000</td>
<td>65</td>
<td>78,728</td>
</tr>
<tr>
<td>100,000</td>
<td>68</td>
<td>78,728</td>
</tr>
<tr>
<td>120,000</td>
<td>69</td>
<td>78,728</td>
</tr>
<tr>
<td>140,000</td>
<td>68</td>
<td>78,728</td>
</tr>
<tr>
<td>160,000</td>
<td>69</td>
<td>78,728</td>
</tr>
<tr>
<td>180,000</td>
<td>69</td>
<td>78,728</td>
</tr>
<tr>
<td>200,000</td>
<td>72</td>
<td>78,728</td>
</tr>
</tbody>
</table>
P-Grid Construction

• Idea
  - Replace the coordinator by a random process
  - P-Grid construction algorithm [Coopis 2001]
  - distributed, decentralized, randomized

  when two peers meet
  if a maximal path length is not reached
  try to extend "their" path in tree
  do the necessary bookkeeping

• Requires some care in order to work efficiently
Efficiency of P-Grid Construction

- Constructing a tree of depth 6 (64 leaves)

![Graph showing the number of "meetings" per peer versus the number of peers. The graph indicates a decrease in the number of meetings per peer as the number of peers increases.]
Non-uniform Data Distribution

- Construct a tree of depth 2 for the following data: 10, 01, 001, 0001, 00001, 000001, 000000

bottleneck
P-Grid Construction Balancing Storage Load

- It makes no sense to create leaves in the tree for data occurring rarely

- Every peer stores initially some data

  when two peers meet

  peer extends path only if #data items > ε

  do the necessary bookkeeping

  otherwise data exchange (duplicate generation)

- Requires some care in order to work efficiently and to correctly balance the storage load
Unbalanced Search Trees

- Problem:
  tree will be deeper where more data items
  ⇒ more work for answering search request

worst case: $O(N)$ steps!
Analysing Required Messages

• Assume only the number of messages required for a search is relevant
  – Multiple nodes in the tree can be traversed without sending a message

Theorem [WDAS 2002]

IF the probability for a reference to another node occurring in the reference lists is equal for all references that possibly can occur,
THEN the number of messages required for a search is $O(\log_2(N))$ no matter what shape the P-Grid (tree) is.

• Equal probability can be achieved by systematically merging the reference lists
Self-Organization in P-Grid

• Nodes decide through local agreement
  – on their position in the search tree when they meet
  – whether to deepen a search tree based on storage load

• Nodes balance data through local operations
  – Reference distribution
    • required for search efficiency
  – Replica distribution of data objects
    • required for search reliability

• Global "agreements" are only on
  – Type of search requests
  – P-Grid organisation
Practical Aspects of P-Grid

- Implementation exists: feasible  
  [IEEE Internet Computing 2002]
- Analysis shows that indexing overhead is reasonable for typical setting  
  [Coopis 2001]
- Algorithms for additional replication of more frequently requested data objects  
  [ICME 2002]
- Update mechanism based on gossiping  
  [EPFL-TR 2002]
- Application for storing reputation data  
  [CIKM 2001]
- More complex queries can be supported (regular expressions, paths, joins)
Overview

1. P2P Systems: Decentralizing Information Systems

2. P-Grid: Efficient Decentralized Search

3. Semantic Gossiping: Emergent Global Semantics
Overview

1. P2P Systems: Decentralizing Information Systems

2. P-Grid: Efficient Decentralized Search

3. Semantic Gossiping: Emergent Global Semantics
### Decentralization: Napster vs. Gnutella

<table>
<thead>
<tr>
<th>Resources</th>
<th>Napster</th>
<th>Gnutella</th>
</tr>
</thead>
<tbody>
<tr>
<td>search</td>
<td>central</td>
<td>decentral</td>
</tr>
<tr>
<td>file exchange</td>
<td>decentral</td>
<td>decentral</td>
</tr>
<tr>
<td>schema</td>
<td>central</td>
<td>trivial</td>
</tr>
<tr>
<td>annotation</td>
<td>decentral</td>
<td>decentral</td>
</tr>
</tbody>
</table>

- Partially decentralized
- Self-Organizing
Why are Schemas Important?

- Example: Searching biological databases
  - Without schema (like Google, Gnutella)

- Searching for data on "anglerfish"
  - Results will be precise

- This seems easy, but the same for "leech"
  - Organism leech
  - Authors: "Bleech", "Leechman", ...
  - Protein sequences: ...MNTS LEECH MPKGD...

- Search for "257" ...
Schema Heterogeneity

- Different databases – Different schemas
  - SwissProt: Find `<Species>` leech `</Species>`
  - EMBLChange: Find `<Organism>` leech `</Organism>`

- Standardization (global schema)?
  - Music files: clear scope, simple semantics 😊
  - Scientific databases: different scope, distributed knowledge, little agreement, etc. 😞

- Hardest problem in information systems: semantic interoperability
Translating Heterogeneous Schemas

• A non-expert may be able to relate
  - <Organism> ⇔ <Species>
  - <Author> ⇔ <Authors> etc.

• But what about
  - <AaMutType> ⇔ <DnaMutType>
  - <FtKey> ⇔ <FtKey>

in Swisschange and EMBLChange?

• The answers can only be given by the experts ... sometimes only by the data owners!

**Approach:** ask them to provide their translations from some "known" schema to their "own" schema (local step)
Local Semantic Interoperability (Translation)

Q1=
<ID>$sp/ID</ID>
FOR $sp IN /SP_entry
WHERE "anglerfish" IN $sp/organism

Q2=
<ID>$sp/ID</ID>
FOR $sp IN T12
WHERE "anglerfish" IN $sp/organism

SwissProt
(known schema)

EMBLChange
(own schema)

Computer-processable languages: XML, XQuery

©2002, Karl Aberer, EPFL-I&C-IIF,
Laboratoire de systèmes d'informations répartis
Global Semantic Interoperability

SwissProt peers
authors, titles, organism, ...

other peers
authors, ...

EMBLChange peers
species, ...

Semantic Gossiping
How to Detect a Semantic Agreement?

SwissProt peers
authors, titles, organism, ...

other peers
authors, ...

EMBLChange peers
species, ...

Check what is preserved in cycles (semantic kernels)!

Query posted at EPFL

organism → organism

organism = organism
OK!

species → organism

species

Swissprot at Geneva

organism

EMBLChange site at Cambridge

species
Research Questions

• Many fundamental problems
  – Erroneous agreements
  – Agreement on schema but not on data
  – Complex data types and mappings
  – Overlapping of data collections

• Approach: algorithms and tools
  – to automatically generate, detect and use local translations
  – identify which are correct with a high probability (via semantic kernels)
  – control of global search (via semantic gossiping)
Conclusion

• New information services without requiring new infrastructure
  – Sharing of existing resources
  – Sharing of existing knowledge

• Technical perspective
  – Self-organization is feasible, efficient, ...

• Social perspective
  – Self-organization preserves autonomy
  – Self-organization requires trust
  – Global behavior based on local agreements

• Models from sociology, economy and biology
 Publications on P-Grid

P-Grid: A Self-Organizing Access Structure for P2P Information Systems

Managing Trust In a Peer 2 Peer Information System

Improving Data Access in P2P Systems
Karl Aberer, Manfred Hauswirth, Magdalena Fungeva, Roman Schmidt, IEEE Internet Computing, 6(1), January/February 2002.

Peer-to-Peer Information Systems: Concepts and Models, State-of-the Art, and Future Systems
Karl Aberer, Manfred Hauswirth, 18th International Conference on Data Engineering, San Jose, California, 2002.
A preliminary version of this tutorial was held at the Joint 9th European Software Engineering Conference (ESEC) and 9th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-9), Vienna, Austria, 2001.

Scalable Data Access in P2P Systems Using Unbalanced Search Trees

A Decentralized Architecture for Adaptive Media Dissemination