

---

# Recent Work in Evergrow/SP3 by TUC

---

**Manolis Koubarakis**

Dept. of Informatics and Telecommunications  
National and Kapodistrian University of Athens  
and

Dept. of Electronic and Computer Engineering  
Technical University of Crete

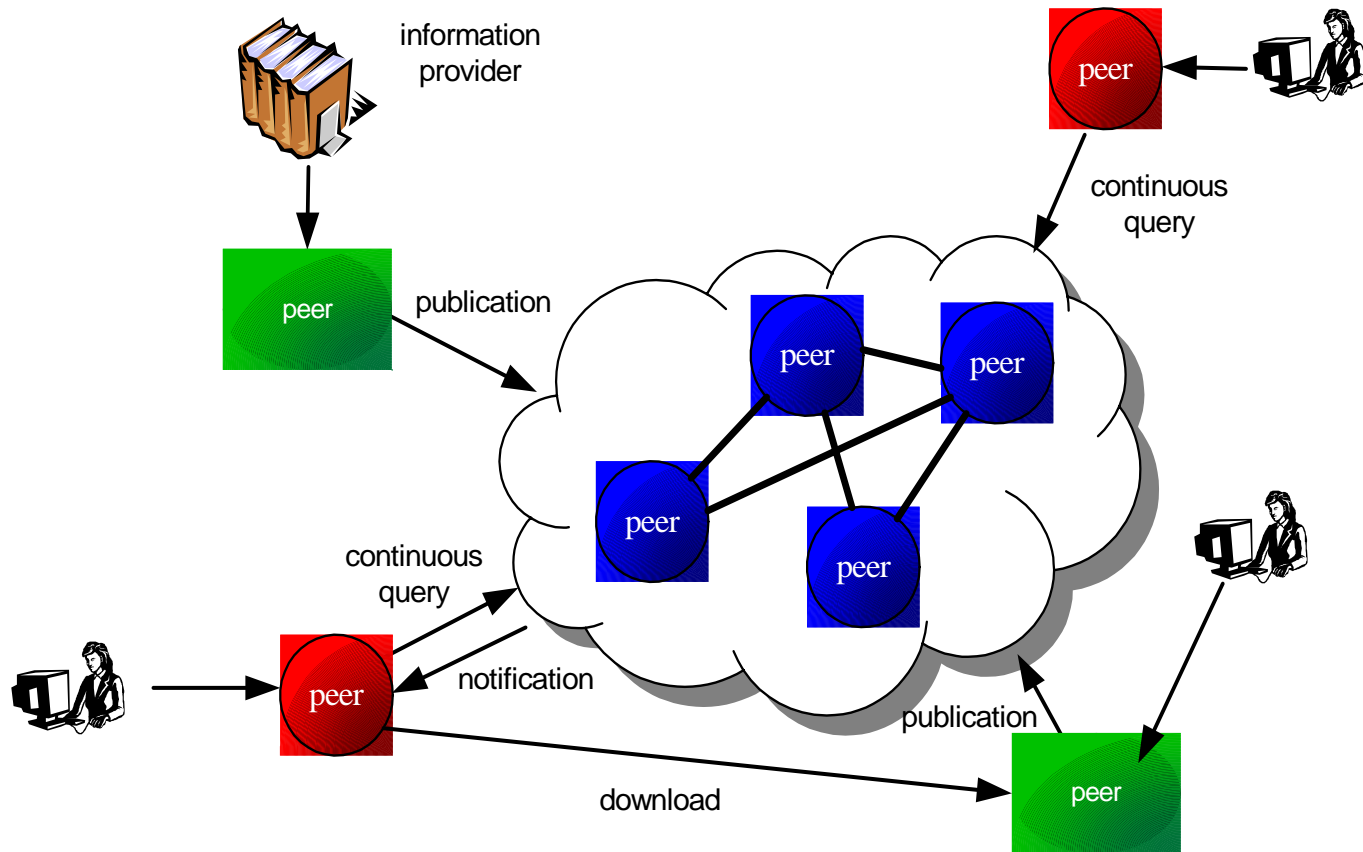
(joint work with C. Tryfonopoulos, S. Idreos and E. Liarou)

---

# Outline

- **Focus:** High-level services on top of overlay networks and especially DHTs (WP31 - Services and overlay systems).
  - **Theme:** Information Push (Publish/Subscribe, Continuous Query Processing)
  - **Topics** studied (papers):
    - DH Trie: Publish/Subscribe functionality in IR Environments. [SIGIR 2005]
    - LibraRing: An Architecture for Distributed Digital Libraries Based on DHTs. [ECDL 2005]
    - Evaluation of continuous equi-join queries on top of DHTs. [ICDE 2006]
-

# Digital Library Example – Publish/Subscribe Scenario



---

# DHTrie [SIGIR, 2005]

- The data model:

- **Publication**: a set of (named attribute, value) pairs where the value is of type **text**.
- Interpret text values under the **Boolean** or **vector space model**, depending on their use in queries.

- Example:

( *AUTHOR* = “Manolis Koubarakis”,  
*TITLE* = “Peer-to-Peer Publish/Subscribe Networks with  
Languages from Information Retrieval”,  
*ABSTRACT* = “We study ...” )

---

# DHTrie (cont'd)

- **Query Language:** Conjunctions of attribute-operator-value atomic formulas.
- Example:

$$(AUTHOR = "John Brown") \wedge$$
$$(TITLE \sqsupseteq (algorithms \prec_{[0,2]} complexity) \wedge filtering) \wedge$$
$$(ABSTRACT \sim_{0.6} "Information alert in structured P2P...")$$

---

---

# The DHTrie protocols

- **Index and store** subscriptions in the DHT. Make sure publications **meet** relevant subscriptions.
  - Two levels of indexing:
    - **Global**: Use an extension of Chord.
    - **Local**: A trie-like structure to store compactly queries with common words (plus some additional data structures). [SIGIR 2004]
-

# The DHTrie protocols (cont'd)

- Subscribing with **Boolean** atomic queries

- Assume query  $q$  of the form:

$$(A_1 = s_1) \wedge \dots \wedge (A_m = s_m) \wedge (A_{m+1} \supseteq wp_{m+1}) \wedge \dots \wedge (A_n \supseteq wp_n)$$

1. Select a single word  $w$  contained in any of  $s_i$  or  $wp_j$ .
2. Forward and store query in node **responsible** for  $id=H(w)$ .

---

# The DHTrie protocols (cont'd)

- Subscribing with **vector space** atomic queries and **mixed type** queries is a little bit more involved.



# The DHTrie protocols (cont'd)

## ■ Publishing a resource

- Assume a publication  $p$  of the form:

$$\{(A_1, s_1), (A_2, s_2), \dots, (A_n, s_n)\}$$

1. Construct  $D_1, \dots, D_n$ , the sets of distinct words in  $s_1, \dots, s_n$ .
  2. Construct list  $L = \{H(w_j) : w_j \in D_1 \cup \dots \cup D_n\}$
  3. Remove duplicates from  $L$ .
  4. Forward publication in **all** nodes responsible for an  $id=H(w_j)$ .
  5. Nodes will find all matching queries using their local data structures and algorithms and notify subscribers.
-

---

# The DHTrie protocols (cont'd)

- Other protocols for pub/sub scenario:
    - Remove a query
    - Update a query
    - Join or leave the network
  - Iterative vs. recursive sending of message to a group of nodes.
  - Frequency cache
  - Load balancing
  
  - Protocols for **one-time** queries.
-

---

# LibraRing [ECDL 2005]

- An **application** of the DH Trie protocols to digital libraries.
  - **Super-peer** network.
  - **Clients** are digital library **providers** or **consumers**.
  - **Super-peers** run the DH Trie protocols.
  
  - Our paper got the **DELOS research exchange award (best student paper award)**. The award supports the student author to visit one of the DELOS NoE partners for a month.
-

# Continuous Equi-join Queries over DHTs [ICDE 2006]

- The Setting: An overlay network (Chord) stores data in the form of **data tuples**. This data can be queried with **continuous SQL queries** (also stored in the network).
- We concentrated on **two-way equi-join** queries.
  - Example:

*Select \* From R,S Where R.A = S.B*

---

# Continuous Equi-join Queries (cont'd)

- Experiment with various algorithms. The high level approach is:
    1. Index a query using one or both of the join attributes (**attribute-level indexing**).
    2. **Rewrite** the query into one that has only **selection** as related tuples arrive (**rewriter nodes**).
    3. **Re-index** the rewritten query to **evaluator nodes** (**value-level indexing**).
    4. Publish tuples so that they **meet** relevant queries.
  
  - Other ideas:
    - Queries are **grouped**.
    - An **extra routing table** minimizes the cost of re-indexing a rewritten query.
    - A simple **replication** scheme (similar to DH Trie) helps to balance the load of rewriter nodes.
    - `send` and `multiSend` in Chord
-

---

# Future Plans

- Continue the work we have done so far:
    - Distributed computation of tf/idf values and distributed computation of rankings.
    - Extensions to more expressive languages e.g., XML+text.
    - Continuous multi-way join queries.
    - ...
-