A flexible and robust lookup algorithm for P2P systems

Mauro Andreolini, Riccardo Lancellotti

University of Modena and Reggio Emilia

Motivations

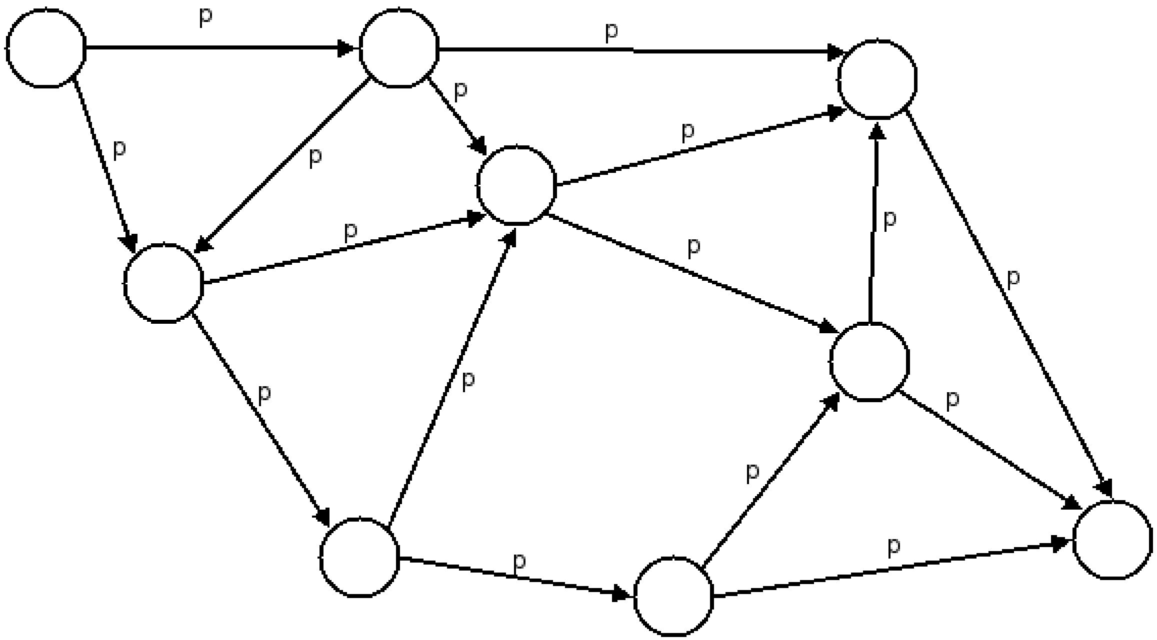
- **Wide popularity of P2P paradigm**
  - File sharing
  - Multimedia streaming
  - File systems
  - Middleware architectures (e.g., P2P+Grid)
  - Cloud computing
- **Focus on P2P lookup algorithms**
  - Need to request resources and obtain suitable responses

Requirements of P2P lookup algorithms

- **Flexibility**
  - Support for complex query semantics
  - Resource identified through multiple keywords
- **Effectiveness**
  - Queries can identify all the suitable resources
  - High query hit rate
- **Efficiency**
  - Low query overhead
  - Low number of messages exchanged per query
- **Robustness**
  - Fault tolerance
  - Queries must be answered even is some node is unavailable

Available alternatives: Flood-based

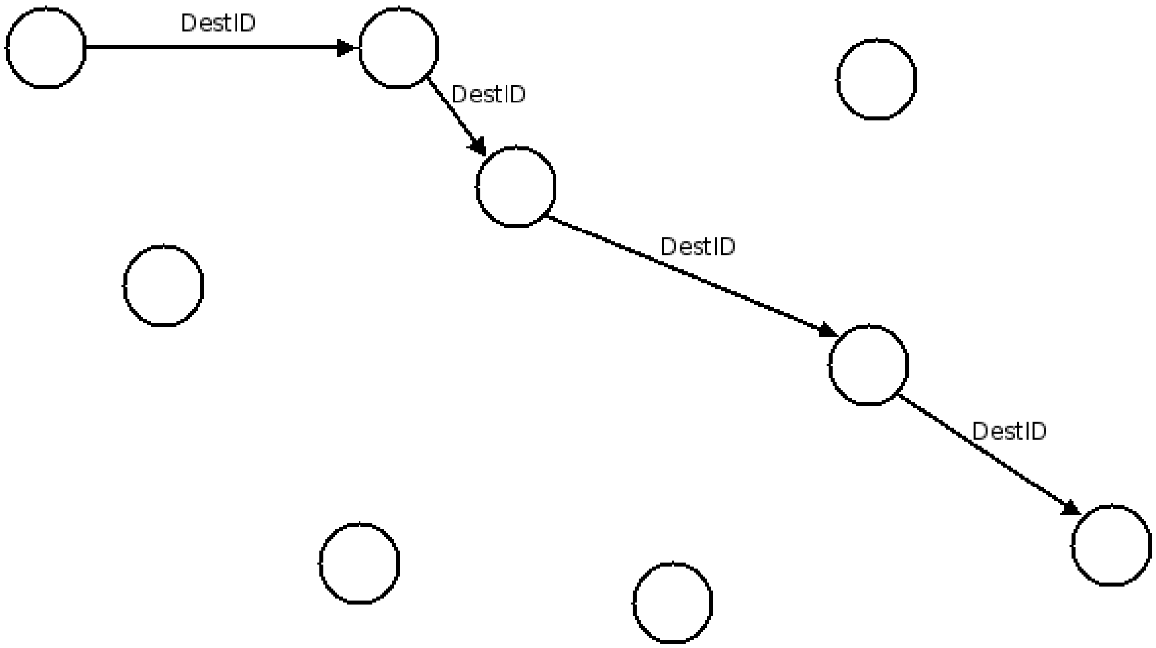
- **Flood-based / Probabilistic flood algorithms**
- **Exploration of the network through neighbor propagation (exploits characteristics of power law networks)**
- **Probabilistic flood explores each neighbor with probability *p***



- **Characteristics:**
  - Flexibility
  - Effectiveness
  - Efficiency
  - Robustness

Available alternatives: DHT

- **Distributed Hash Tables (DHTs)**
- **Query routing within an hash space**
- **Need to know exact Destination ID**
- **Characteristics:**
  - Flexibility
  - Effectiveness
  - Efficiency
  - Robustness



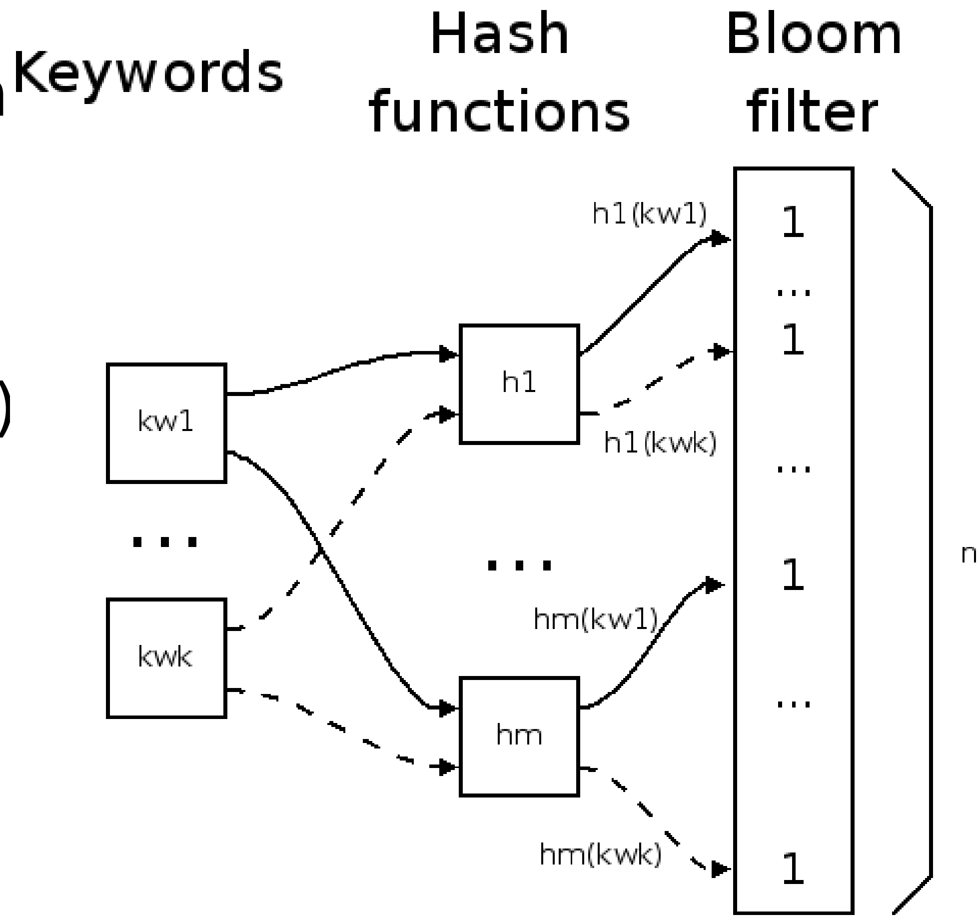
→ Goal: merge the benefits of existing solutions without disrupting existing protocols

Proposal: Fuzzy-DHT

- **Implements keyword-based search within a DHT** (Pastry)
- **Inherits efficiency from DHT**
  - Preserves low query overhead
- **Introduces a new query semantics**
  - Improved flexibility
- **Minor changes in the original routing algorithm**
  - no need for reverse index data structures
- **Changes with respect to original DHT:**
  - New hash function to represent keywords
  - Modified query routing algorithm

Fuzzy DHT hash function

- **Hash function must:**
  - Support the representation of multiple keywords *kw1, kw2, ..., kwk*
  - Have fixed length on *n* bit (compact representation)

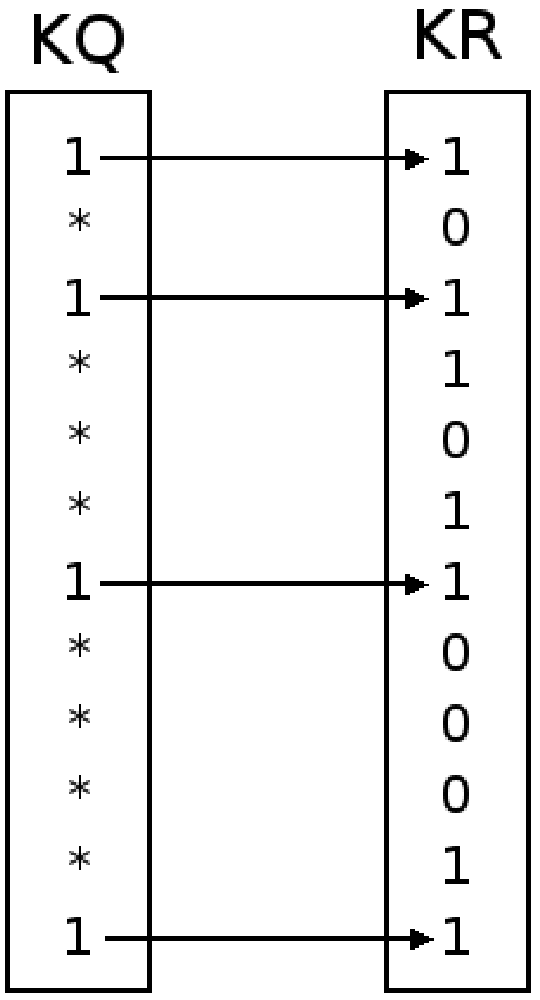


- **Use of a Bloom Filter as the hash function**
- **The ID of a resource depends on its keywords**

- **Bloom filter uses *m* hash functions to represent set contents as a string of *n* bits**

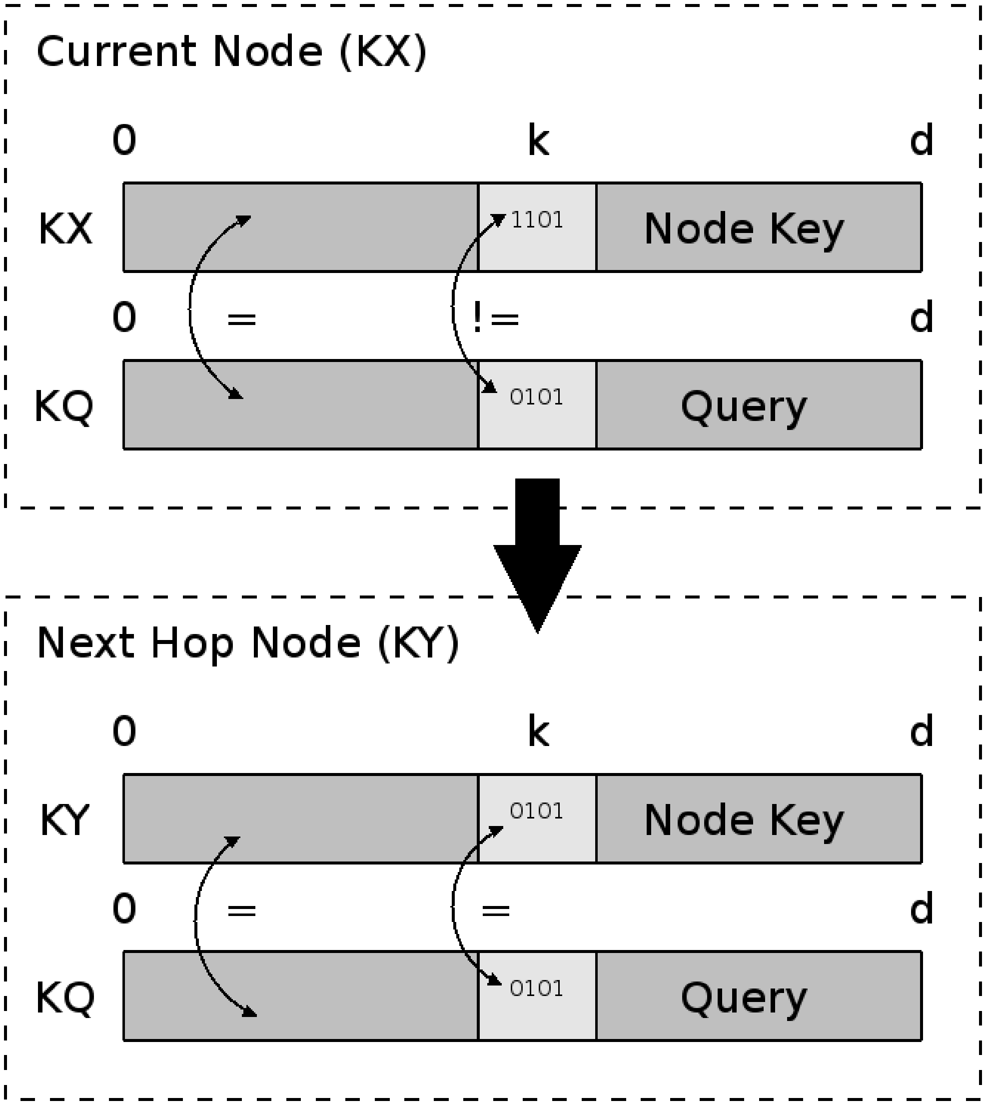
Support for keyword matching

- **Given**
  - a query KQ that represents a set of keywords
  - a resource ID KR with its keywords
- Query semantics:
  - Keywords in KQ are a subset of keywords in KR
- Returns a hit if and only if every bit set to 1 in KQ is set also in KR
- **Each “0” in the query is considered as a wildcard**



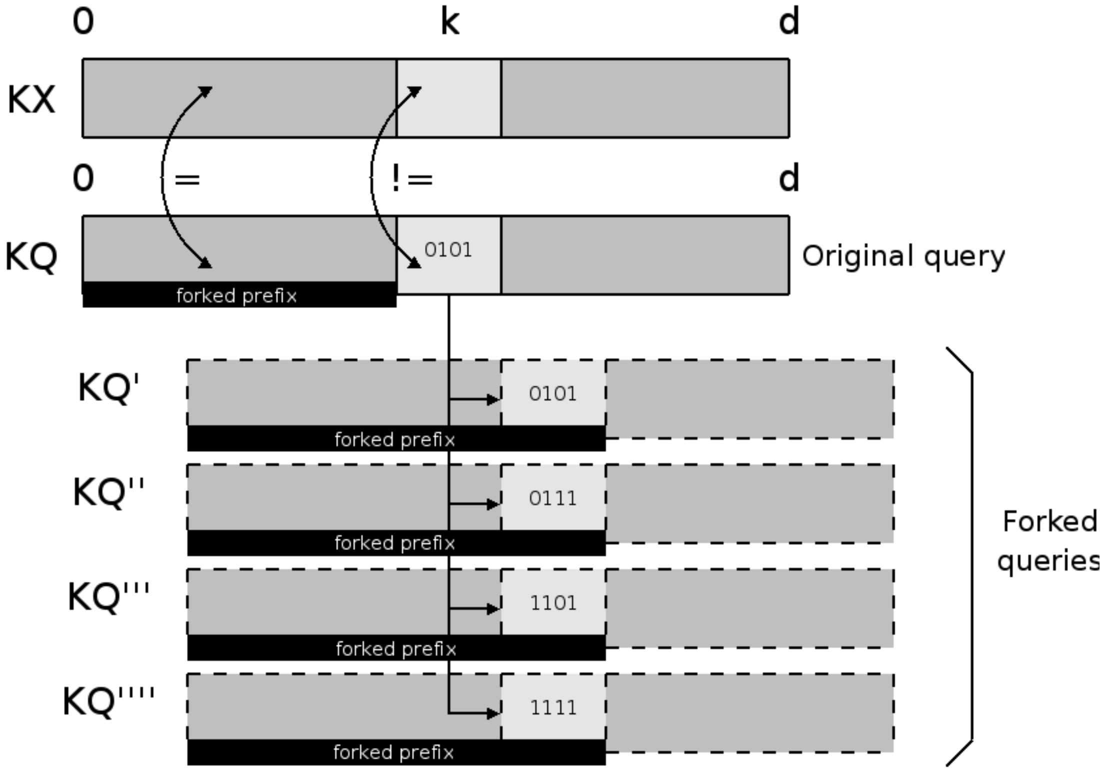
Pastry lookup algorithm

- **Lookup based on Plaxton algorithm** (*n* bits → *d* digits)
- **Routing of query *KQ*, step *k***
  - Receiving node (*KX*) has the first *k-1* digits equal to *KQ* (*shared prefix*)
  - The next hop (*KY*) is selected in order to have a shared prefix of k digits
- **This algorithm must be adapted to lookup based on multiple keywords**



Fuzzy-DHT lookup algorithm

- **At each lookup step the original query is forked**
- **Example: step k**
  - Digit k is the first digit after shared prefix
  - For each “0” in digit k we split the query (query forking)
  - Two forked queries, with bit set to 0 and 1
  - No need to fork first k-1 digits: fork already occurred
- **Forked queries are routed according to Plaxton algorithm**



- **Digit k=0101 → 4 forked queries**
  - Digit k=0101
  - Digit k=0111
  - Digit k=1111
  - Digit k=1101

Fuzzy-DHT evaluation

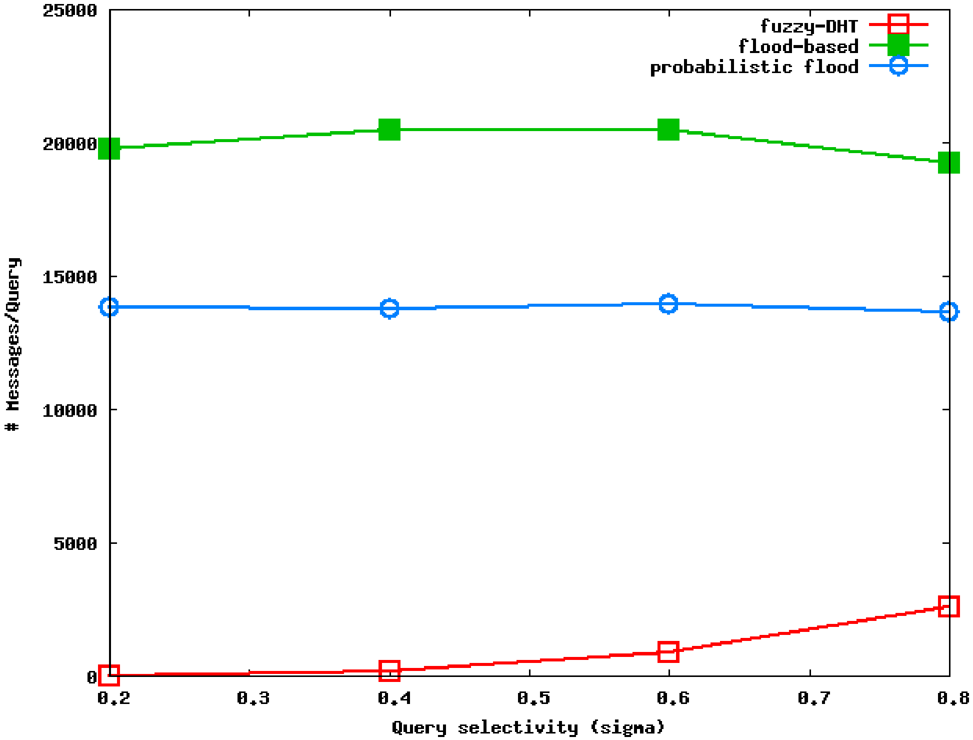
- **Fuzzy-DHT satisfies flexibility requirements by design**
- **Evaluation of:**
  - Effectiveness
  - Efficiency
  - Robustness
- **Comparison with other alternatives**
  - Flood-based protocol (Gnutella)
  - Probabilistic flood
- **Detailed model for flood-based protocols → fair comparison**
  - Barabasi-Albert model for neighbors
  - Preliminary experiments for protocol tuning
- **Simulation based on ns-2**

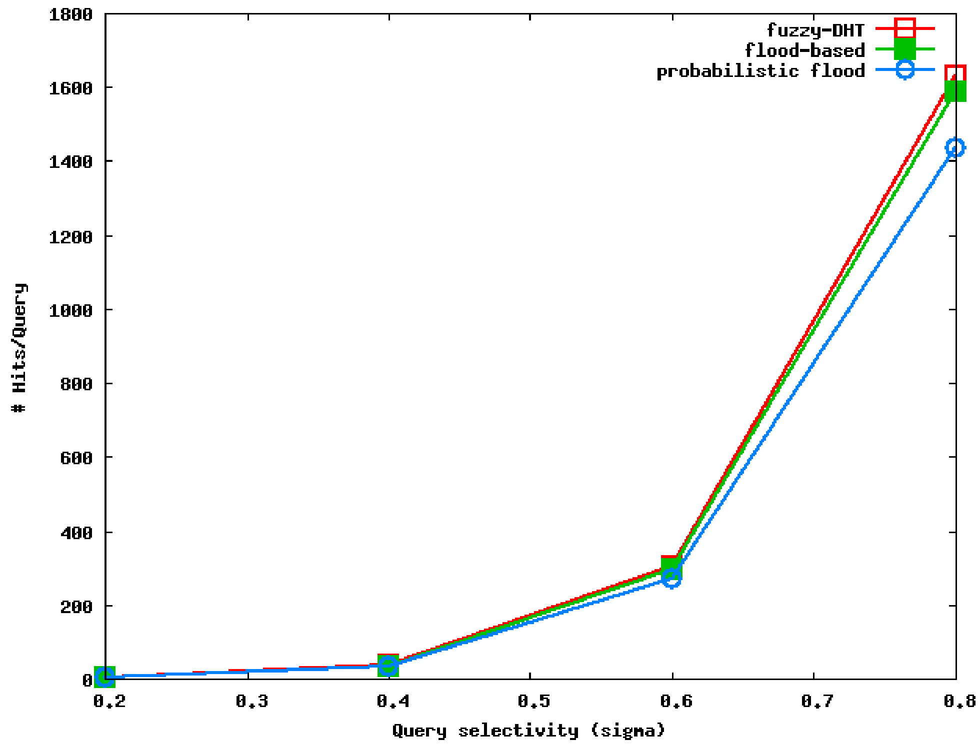
Experimental setup

- **Wide set of scenarios considered**
- **Network size:**
  - 100-1000 nodes (default 500 nodes)
- **Network topology**
  - BRITE network topology generator
  - Real topology University network (not shown)
- **Query selectivity (sigma)**
  - 0.2 – 0.8 (default 0.6)
  - Amount of “0” in the query key
  - Typical value for a 3-4 keyword query: 0.6-0.7
- **Node failure probability**
  - 0 – 0.15 (default 0)

Impact of query selectivity

- **High effectiveness for all protocols**
  - within 5% of theoretical values
  - Probabilistic flood is slightly less effective than other solutions

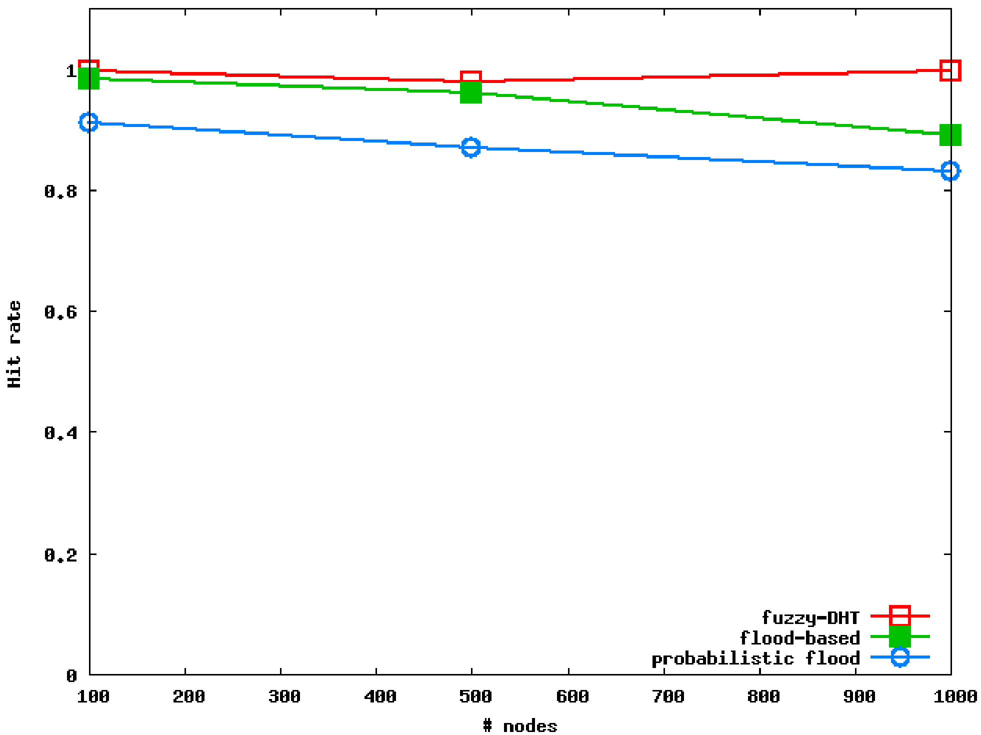


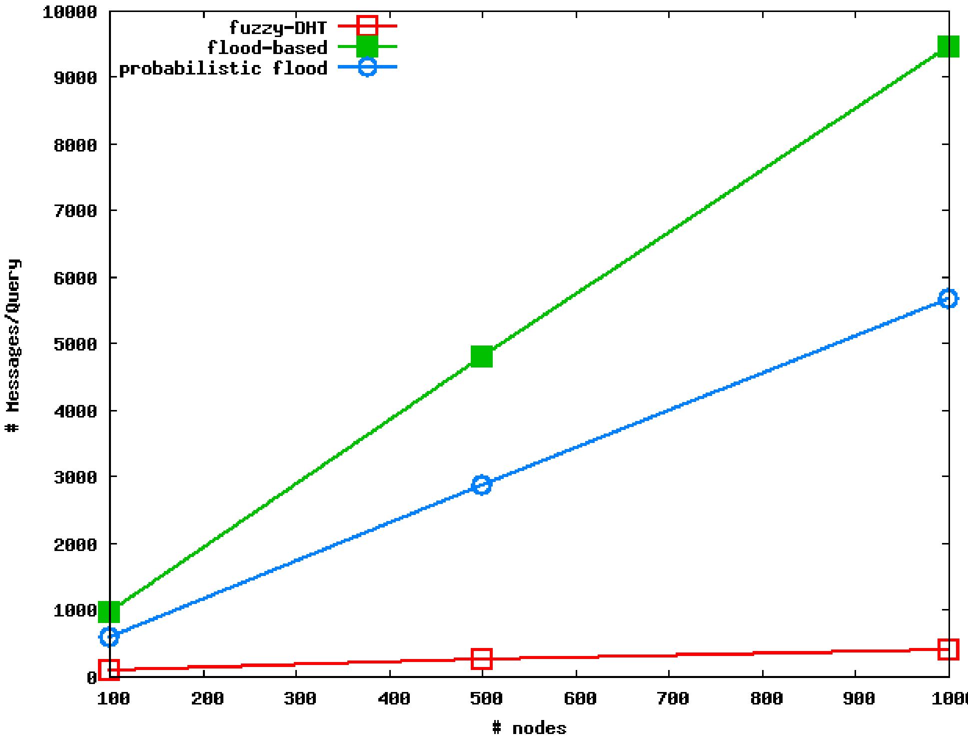


- **Fuzzy-DHT → high efficiency**
  - significant reduction of overhead
  - Fuzzy-DHT overhead at least 1 order of magnitude lower

Scalability evaluation

- **Effectiveness of protocol does not change with network size**

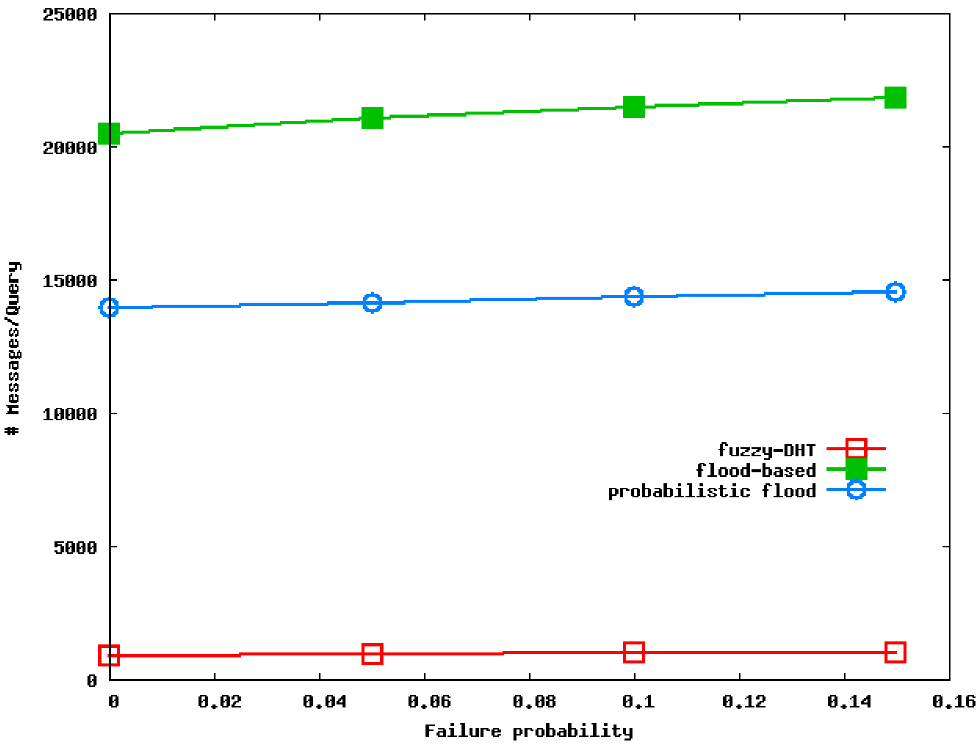


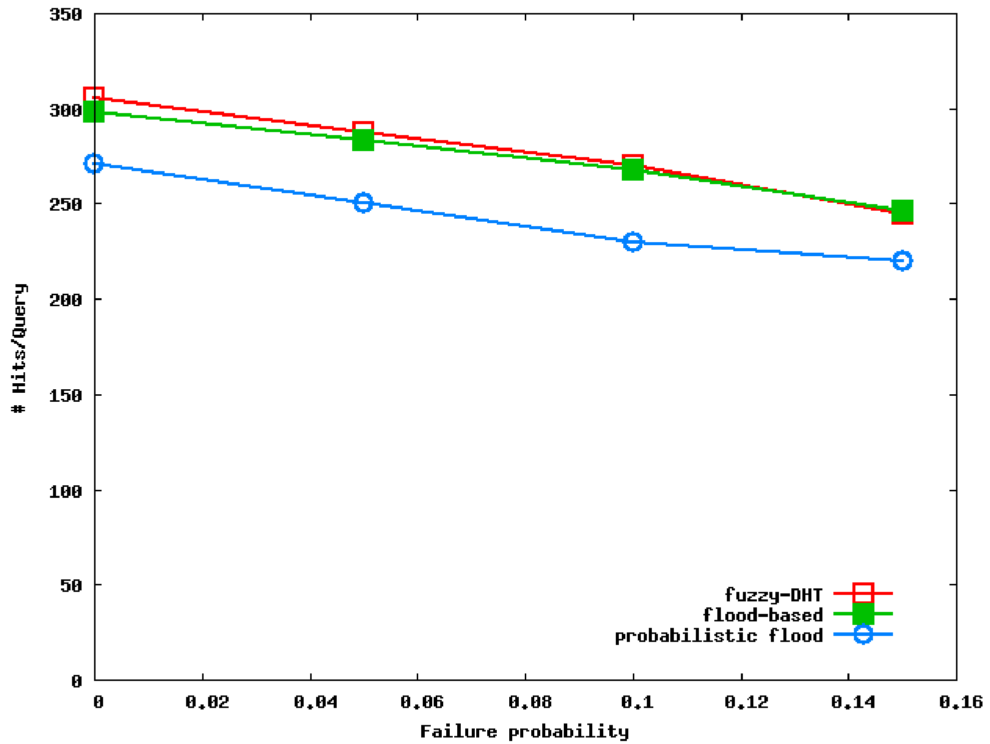


- **Overhead grows linearly with number of nodes**
  - Fuzzy-DHT preserves a low overhead in large networks
  - Fuzzy-DHT improves lookup scalability

Robustness evaluation

- **Presence of failure does not change the results of the analysis**
- **Fuzzy-DHT is a robust algorithm**





- **Fuzzy-DHT algorithm ensures**
  - High effectiveness
  - Low overhead

Conclusions

- **Analysis of P2P requirements for lookup algorithms**
- **Trade-off between flexibility and efficiency**
  - Flood-based vs. DHT

- **Proposal of Fuzzy-DHT**
  - Flexibility → Fuzzy-DHT supports multiple keywords
  - Effectiveness → Fuzzy-DHT has hit rate close to 1
  - Efficiency → Query overhead at least one order of magnitude lower than alternatives
  - Robustness → Small performance degradation even with 15% of faulty nodes

- **Fuzzy-DHT can be easily implemented with little modifications over existing DHTs**

A flexible and robust lookup algorithm for P2P systems

Mauro Andreolini, Riccardo Lancellotti

University of Modena and Reggio Emilia