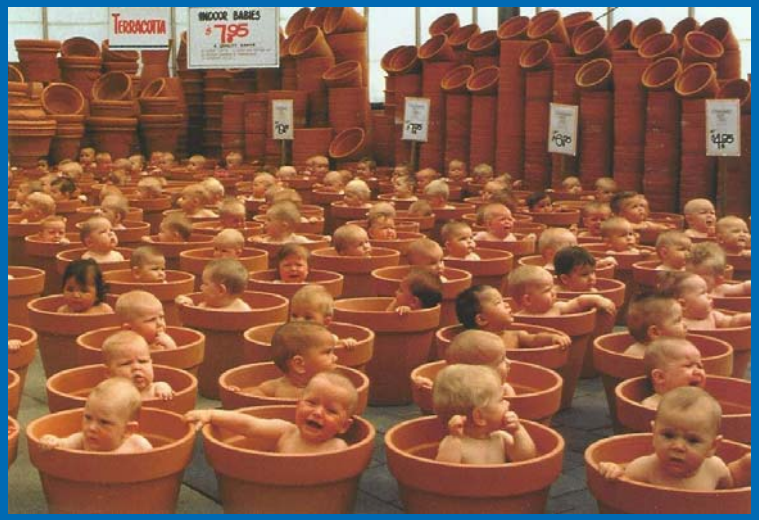
Making TripCom scalable



Manfred Hauswirth, DERI Galway, NUIG

The problem rephrased

- TripCom is supposed to provide a distribute data store for triples with support for reasoning
- This requires:
  - A distribution strategy (organization of the data stores)
  - A distributed indexing strategy
  - Query support (expressivity of queries)
  - A consistency model (updates, on-/offline behaviour of nodes)
  - A model for the operational environment
- Support for reasoning is directly impacted by all of these decisions

Organization of data stores

Sixth Framework Programme Priority 2
Information Society Technologies (IST)
Specific Targeted Research Project

Organization of data stores

Three essential options:

- Centralized (Google)
- Semi-decentralized (Napster, Kaaza)
- Decentralized (Gnutella, distributed hash tables)

Centralized organization

- Strengths
  - Global view and ranking
  - Expressive query predicates
  - Fast response time
- Weaknesses
  - Infrastructure, administration, cost
  - A new company for every global application?
  - Bottleneck, single-point-of-failure
- Problem: If we go for this, what is the added value of TripCom?
  - @ review: “What does TripCom offer that I cannot do with a standard database?”

(Semi-) decentralized organization

- Organization:
  - Index is centralized
  - Data storage is decentralized with direct access
- Strengths: Resource Sharing
  - Every node “pays” its participation by providing access to its resources
  - Physical resources (disk, network), knowledge (annotations), ownership
  - Every participating node acts as both a client and a server => P2P
  - Global information system without huge investment
  - Decentralization of cost and administration = avoiding resource bottlenecks
- Weaknesses: Centralization
  - Server is single point of failure
  - Unique entity required for controlling the system = design bottleneck

Decentralized organization

- Organization:
  - Both index and data are decentralized with direct access
  - No node has a global view or a special role
- Strengths:
  - Good response time, scalable
  - No infrastructure, no administration
  - No single point of failure
- Weaknesses:
  - Higher network traffic
  - Higher costs for maintaining the integrity of the infrastructure and data
  - Design choices have a significant impact on the level of functionality the system can provide (type of queries, robustness, etc.)

Indexing

Sixth Framework Programme Priority 2
Information Society Technologies (IST)
Specific Targeted Research Project

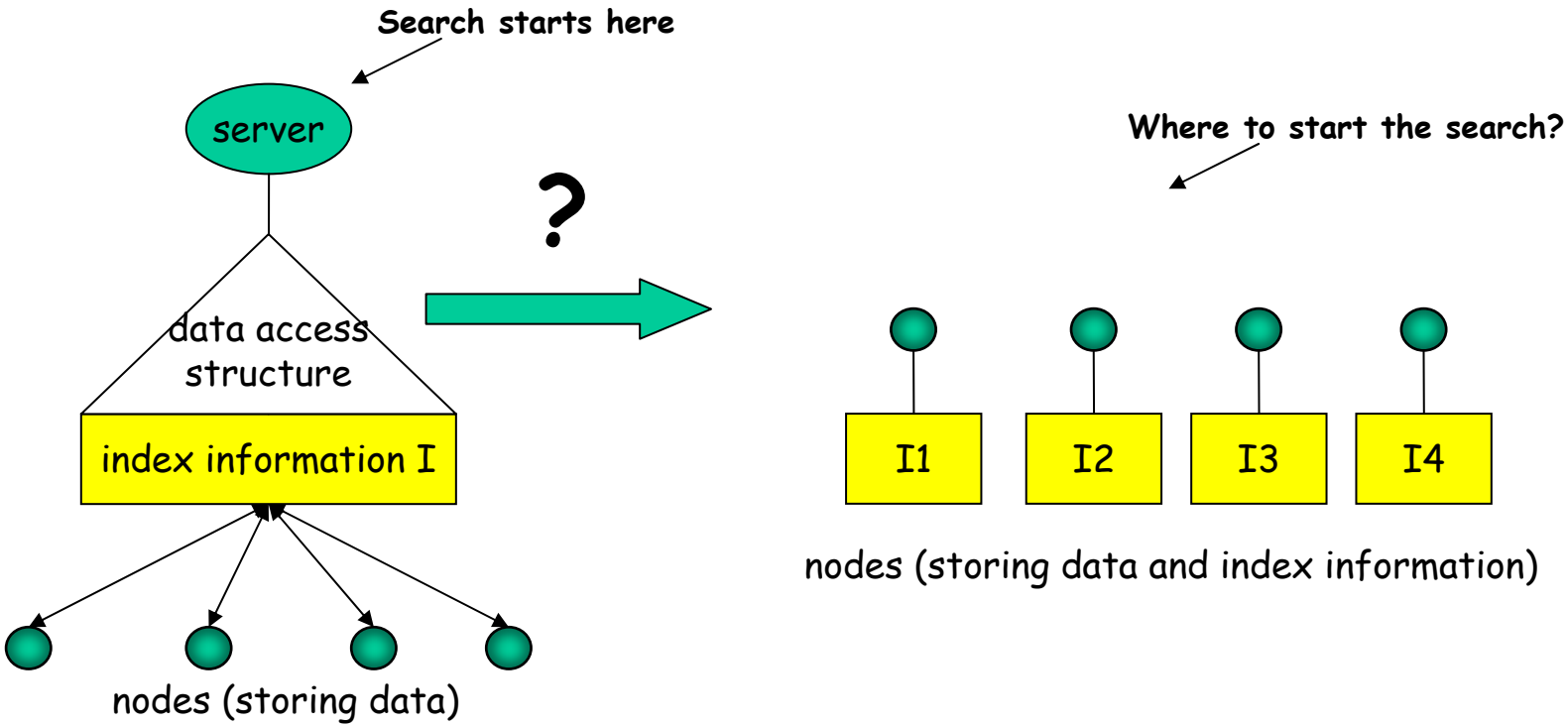
Design choices: Unstructured

Unstructured organization (Gnutella-like):

- Performance
  - Search latency: low (graph properties)
  - Message Bandwidth: high
    - improvements through random walkers, but essentially the whole network needs to be explored
  - Storage cost: low (only local neighborhood)
  - Update and maintenance cost: low (only local updates)
  - Resilience to failures good: multiple paths are explored and data is replicated
- Qualitative Criteria
  - Search predicates: very flexible, any predicate is possible
  - Global knowledge: none required
  - Node autonomy: high

Design choices: Structured

- Goal: provide efficient search using few messages without using designated servers
- Easy: distribution of index information over all peers, i.e. every peer maintains and provides part of the index information
- Difficult: distributing the data access structure to support efficient search



Design choices: Structured approaches

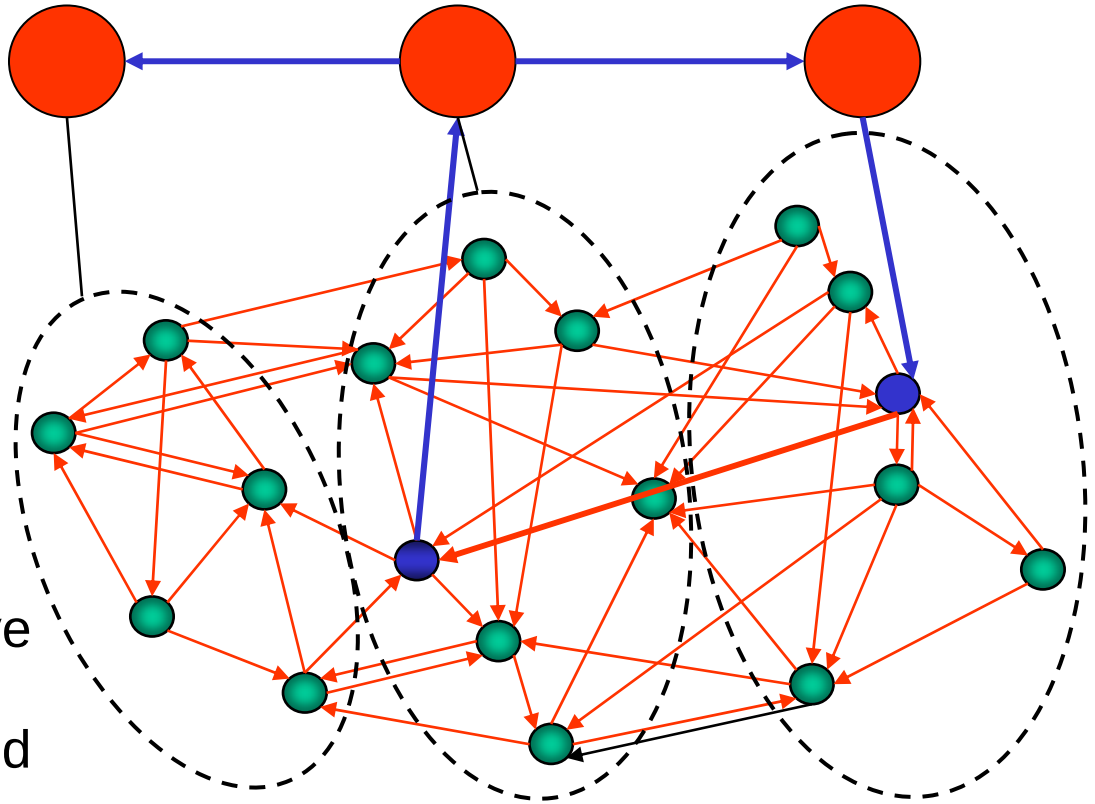
- Different strategies
  - P-Grid: distributing a binary search tree
  - Chord: constructing a distributed hash table
  - CAN: routing in a d-dimensional space
  - Freenet: caching index information along search paths
- Commonalities
  - Each peer maintains a small part of the index information (routing table)
  - Searches performed by directed message forwarding
- Differences
  - Performance and qualitative criteria

Comparison

	Paradigm	Search Type	Search Cost (messages)
Gnutella	Breadth-first search on graph	String comparison	$2 * \sum_{i=0}^{TTL} C * (C-1)^i$
Freenet	Depth-first search on graph	Equality	O(Log n) ?
Chord	Implicit binary search trees	Equality	O(Log n)
CAN	d-dimensional space	Equality	O(d n^(1/d))
P-Grid	Binary prefix trees	Prefix	O(Log n)

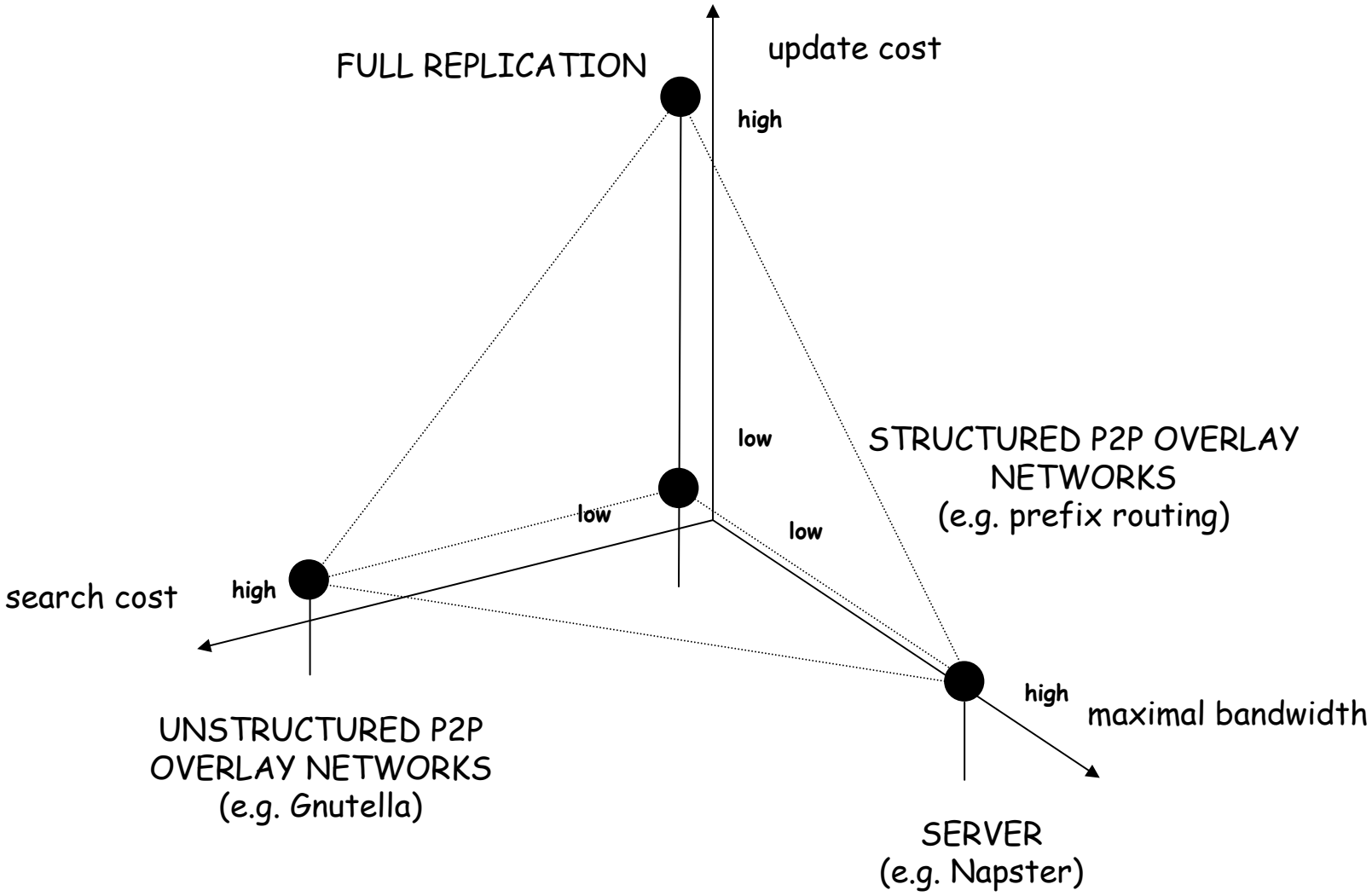
Design choices: Hybrid (Super-Peer)

- Improvement of Central Index Server (Morpheus, Kaaza)
  - Multiple index servers build an overlay network
  - Clients are associated with one (or more) Super-peers
  - Super-peers route search requests



- Experiences
  - Redundant superpeers are good
  - Super-peers should have high outdegree (>20)
  - TTL should be minimized

Scalability trade-offs



Discussion

Sixth Framework Programme Priority 2
Information Society Technologies (IST)
Specific Targeted Research Project

Structural choice

- TripCom so far is centralized with heavy-weight servers
- Heavy servers are required to deliver some of the key functionality promised
- Data needs to be distributed to credibly scale up to Internet size

⇒ Proposal

- Super-peer architecture with servers as super-peers
- Normal users: peers

Indexing - discussion

- What query expressivity is required?
  - Equality, ranges, similarity, joins?
  - Updates?
- Query expressivity is a key determining factor which indexing strategy can be used
- Only minimal requirements on expressivity to avoid complexity, query latency and support required guarantees
- What parts of a triple need to be indexed?
  - Size of the index
- Which query language should be used?

Data stores - discussion

- Query processing and available storage require load-balancing with conflicting requirements
  - We need to find a trade-off here!
- Storage load-balancing requires shipping of data $\Rightarrow$ significant network traffic + delay
- Efficient query processing may require clustering which conflicts with storage load-balancing (same for skewed distributions which will occur in reality)
- What level of consistency has to be supported?
  - Probabilistic guarantees instead of standard DB guarantees
  - What is the impact of probabilistic guarantees on the functionality

Query processing – discussion

- Database problems to be solved
  - Logical query execution plan must be devised without global information
  - Physical query execution plan needs to be derived in a way which minimizes network transfers
  - Network delay is the key limiting resource!
- Distributed transactions are not possible at Internet scale!
  - Complexity to high
  - Seriously inhibit parallel processing (inherent serialization)
- Updates impact index and data storage consistency

Operational environment - discussion

- What is the on-/offline behaviour of the super-peers (churn)?
  - Are they permanently available?
    - Yes ⇒ piece of cake (but is this realistic?)
    - No ⇒ non-negligible maintenance costs and weak consistency
  - Significant churn ⇒ significant replication necessary ⇒ significant efforts for maintenance
- Who becomes a super-peer and when?
  - Adding/deleting a super-peer incurs significant reorganization and costs
  - Should only happen when necessary
  - System should be able to determine this automatically

Operational environment - discussion

- What is the rate of
  - Inserts
  - Updates
- What amount of data will be
  - Inserted
  - Updated

Sixth Framework Programme Priority 2
Information Society Technologies (IST)
Specific Targeted Research Project

Reasoning – discussion

- Reasoning has the requirement to have full access to all data all the time
  - Significant cost in a distributed setting (network delay, consistency)
- Probabilistic guarantees vs. correctness of reasoning?

Sixth Framework Programme Priority 2
Information Society Technologies (IST)
Specific Targeted Research Project