

Decentralized Auction-based Pricing with PeerMart

David Hausheer¹, Burkhard Stiller^{2,1}

¹*Computer Engineering and Networks Laboratory TIK
Swiss Federal Institute of Technology ETH Zurich*

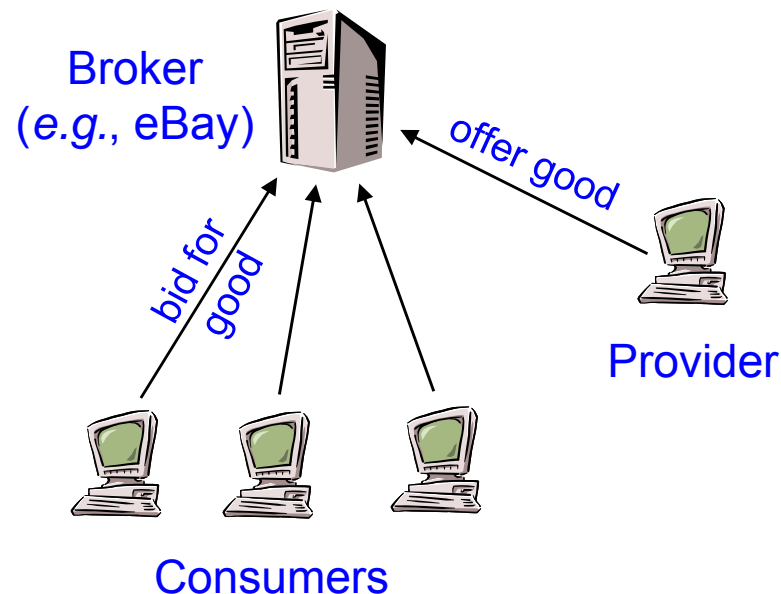
²*Department of Informatics IFI, University of Zurich
E-Mail: [hausheer, stiller]@tik.ee.ethz.ch*

Motivation and Goals
PeerMart Requirements and Design
Analytical and Experimental Results
Conclusions

Motivation

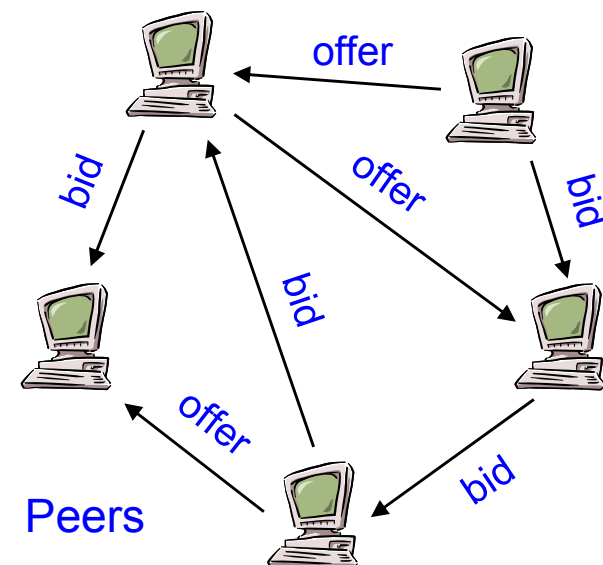
□ Centralized auctions

- Dedicated (centralized) broker
- + High reliability
- + High technical efficiency
- Not scalable
- Single Point of Failure
- Vulnerable against attacks



□ Decentralized (P2P) auctions

- Peers act as brokers, providers, and consumers **at the same time**
- + Scalable
- + No Single Point of Failure
- Vulnerable against selfish or malicious peers
- Technical feasibility?



Goals

□ Key goals

- To maintain the **economic efficiency** of auctions
- To exploit the **scalability** and **resilience** of P2P networks
- To achieve a **high reliability** even in the presence of **malicious** peers
- To provide a **technically efficient** solution

▶ Re-use of **economic efficiency** characteristics of double auctions

▶ Focus on **technical feasibility** of implementing a P2P double auction in a P2P environment



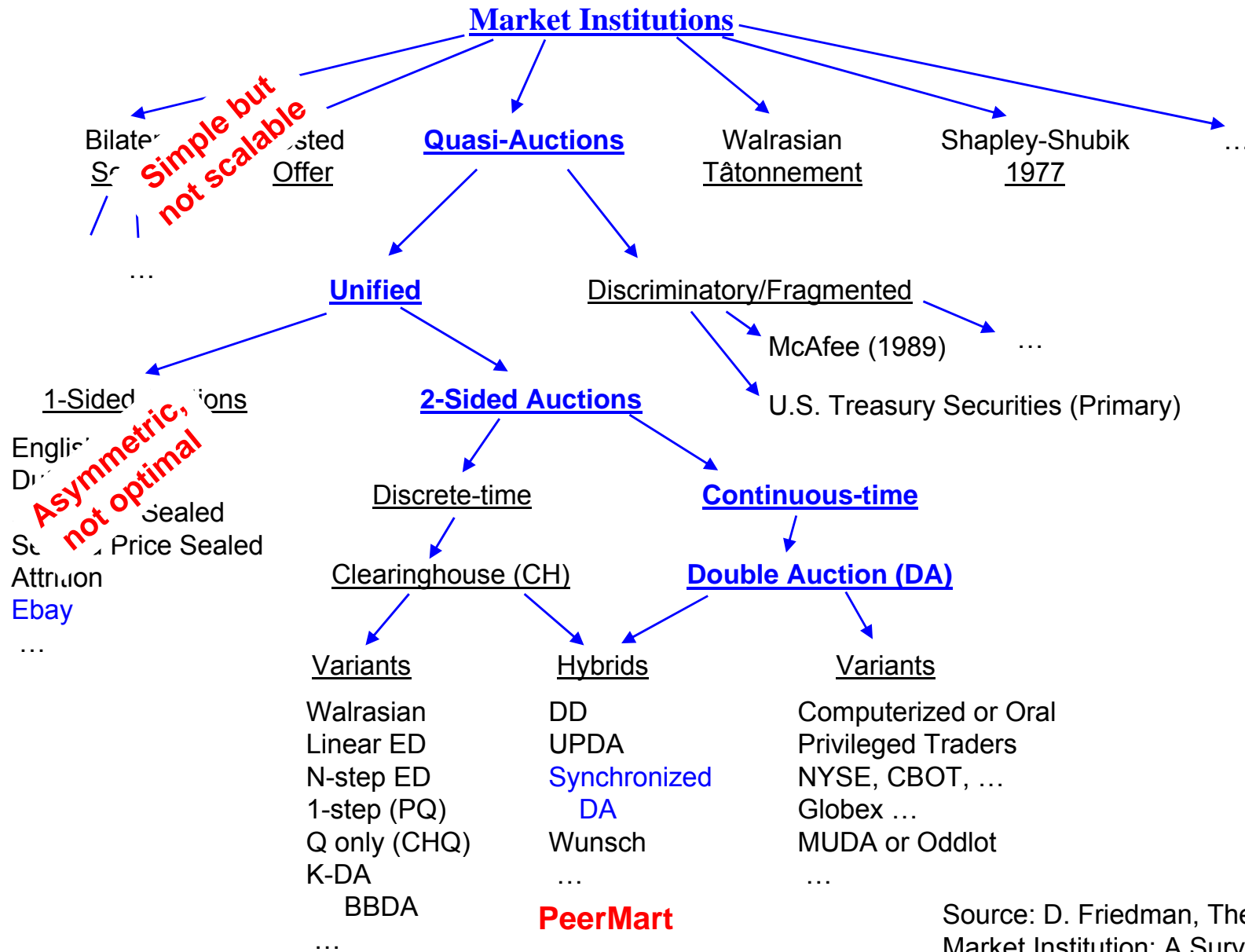
PeerMart Requirements

- Economic efficiency
 - Double auctions close to the ideal case (maximization of benefits)
- Technical performance
 - Efficient use of resources (capacity, storage, processing, messages)
- Scalability
 - Any number of services under any type of load being traded
- Reliability
 - Continuous availability, correctness, secure, DoS-safe, selfish-safe
- Accountability
- Privacy
- Incentive compatibility

- Determine suitable trade-offs for contradicting requirements



Design Space for Auctions



Source: D. Friedman, The Double Auction Market Institution: A Survey, 1993



Related Work on P2P Auctions

- Z. Despotovic, J. Usunier, K. Aberer: [Towards Peer-To-Peer Double Auctioning](#); 37th HICSS Conference, January 2004.
 - Offers/Bids are broadcasted in a Gnutella-like fashion
 - Any peer can answer with a counter offer
 - Not scalable, not strategy-proof

- E. Ogston, S. Vassiliadis: [A Peer-to-Peer Agent Auction](#); AAMAS Conference, July 2002.
 - Agents are connected in a random network
 - Single agent is assigned as cluster center
 - Cluster size is limited => not scalable
 - No message delay is assumed => not realistic

- M. Fontoura, M. Ionesu, N. Minsky: [Decentralized Peer-to-Peer Auctions](#); Journal of Electronic Commerce Research, January 2005.
 - Based on Law Governed Interaction (LGI) paradigm
 - Only the auction process itself is decentralized
 - Communication overhead due to routing messages via controllers



PeerMart Approach: Distributed Double Auctions

□ Key Idea

- **Providers** and **consumers** offer prices for services
- Offers are **optimally matched** by **broker peers**

□ Basic Algorithm

- Providers **publish services** they wish to provide
 - Broker peers reply with a **bid price** (current highest pay price)
- Consumers **request services** they wish to use
 - Broker peers reply by an **ask price** (current lowest sell price)
- Brokers run a **matching strategy** at regular intervals:
 - Upon every price offer, if offer is higher (lower) than current bid price (ask price) => no match, store offer in a table
 - Otherwise, forward offer to the peer with the best counteroffer (highest bid/lowest ask)



Technical Design (1)

- PeerMart uses a **structured P2P overlay network** to distribute the broker load
 - **Pastry** is used as underlying infrastructure
 - Provides unique **128-bit nodeIDs**
 - Stores n closest nodeIDs (**leaf-set**) in routing table

- Each peer has a **public/private key pair** to sign messages
 - Assumption: Public keys certified offline and bound to nodeIDs

- Services are mapped onto a **redundant set of n broker peers** (broker-set)
 - Assumption: Services have **unique IDs** (e.g., hash value of a file)



Technical Design (2)

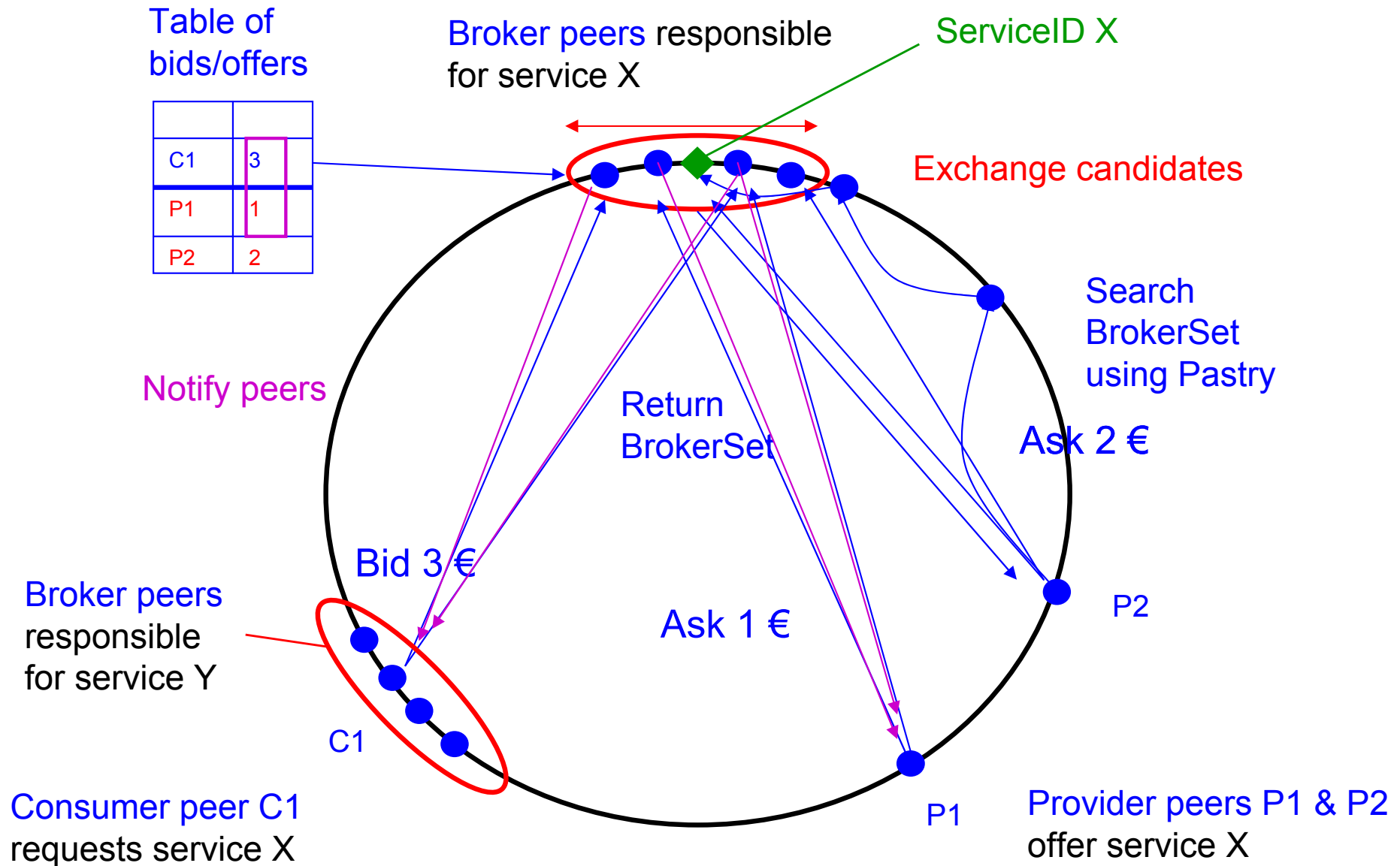
- Consumers/providers **randomly select f** brokers out of the broker set
 - => Load is uniformly distributed on all broker peers

- Each broker peer keeps table of **m highest bids/lowest asks**
 - Lower bids and higher asks are **rejected**

- **Matching** is performed in a **decentralized way**
 - Candidates for a match are **forwarded** to other brokers
 - Candidates: **local matches** and **next best offer**
 - Final matches are determined using **majority decisions**
 - Winners are notified by the f brokers that received the offer



Double Auction in PeerMart



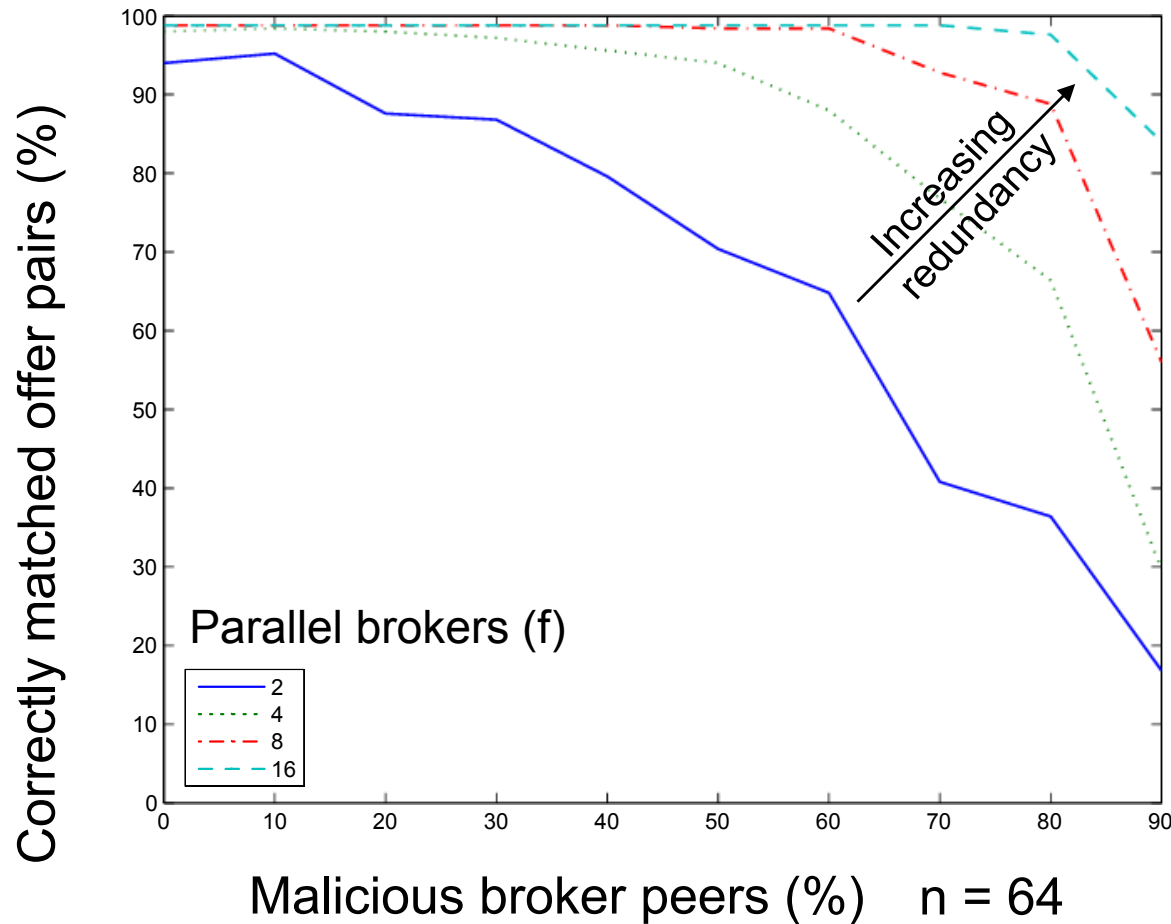
Analytical Results (1)

- Basic effort for **maintaining Pastry** is $O(\log_b N)$
 - N: Number of peers in the network
- **Finding** responsible **broker-set** is $O(\log_b N)$
- All subsequent communication happens **directly**
 - **Sending offers** is $O(f)$
 - **Exchange/forward** of matching **candidates** is $O(f \cdot n)$
 - **Notify peers** about **final matches** is $O(f)$
 - f: Number of parallel active brokers within a single set
 - n: Average broker set size

} Total costs
- Average number of **offers stored** per peer: $s \cdot n \cdot m$
 - s: Average number of service involvements per peer (naturally limited)
 - m: Maximum number of offers and bids per service



Analytical Results (2)



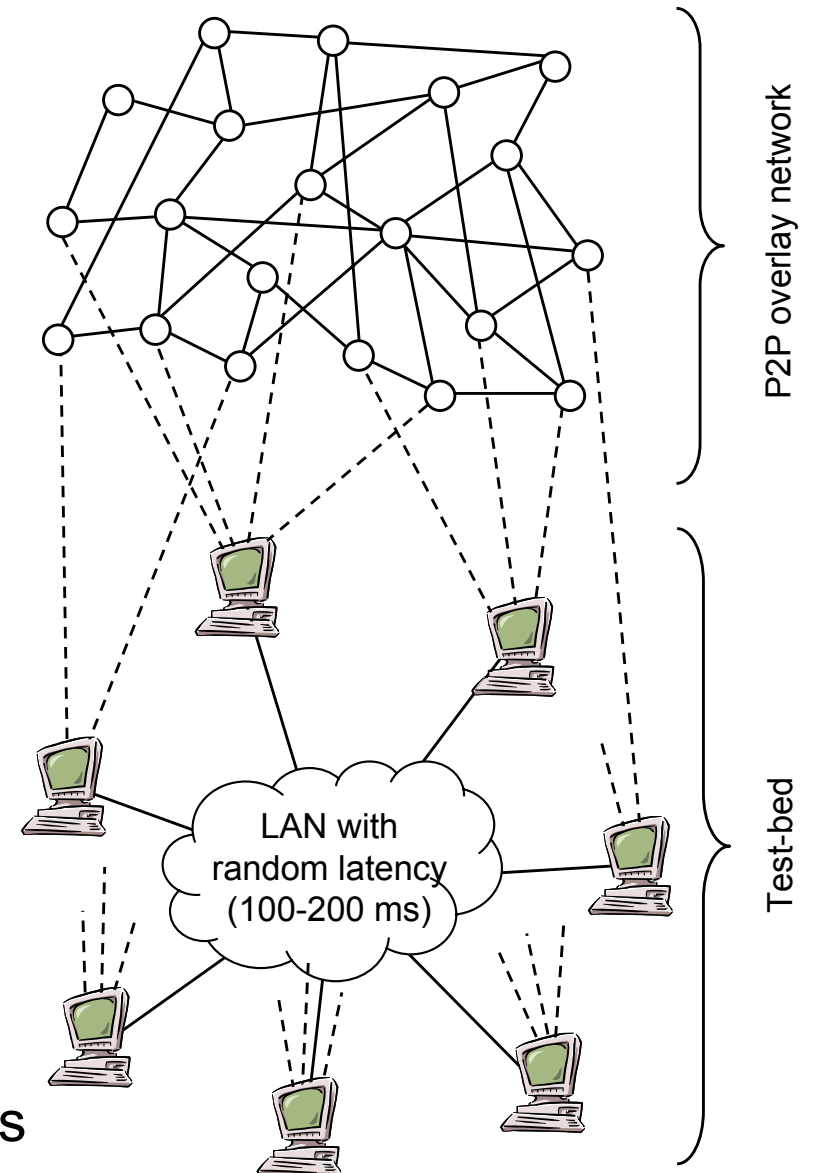
Reliability

- PeerMart correctly matches offer pairs for **< 50% malicious peers** using, e.g., **8 brokers** in parallel (out of **64 brokers** in total)
- If fraction of malicious peers is **< 25%**, even 4 parallel brokers provide good reliability

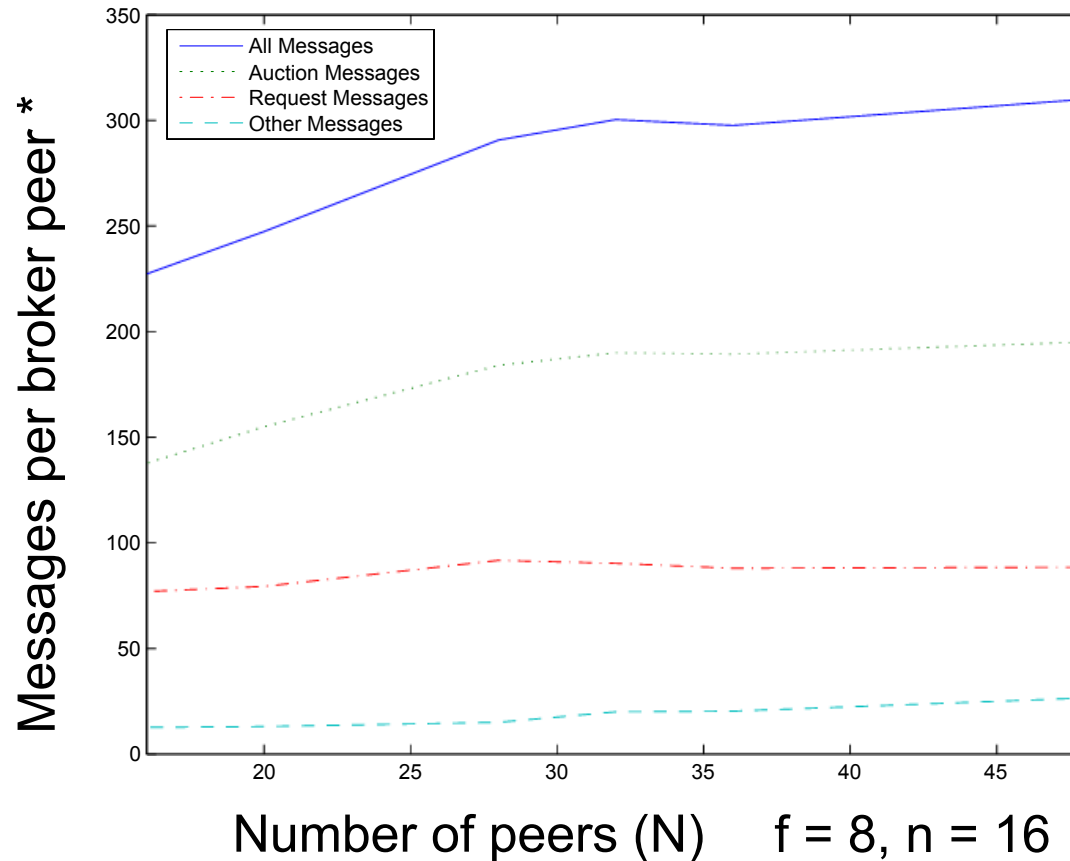


Experimental Setup

- Prototype implemented in Java on top of **FreePastry**
- Assumptions for experiments:
 - Each peer assigned to $s = 2$ services
 - Peers follow the ZIP **bidding strategy**:
 - Consumer's price offer:
$$p_{bid} = \min(p_{ask} + \alpha(p_{max} - p_{ask}), p_{max})$$
 - Provider's price offer:
$$p_{ask} = \max(p_{bid} - \alpha(p_{bid} - p_{min}), p_{min})$$
 - p_{max} , p_{min} are reservation prices
→ normally distributed such that 50% of offers match, α set to 0.1
 - **Malicious** peers:
 - Uniformly distributed
 - Do not forward offers nor notify peers



Experimental Results (1)



**Scalability
(Increasing
Network
Size, Message
Overhead)**

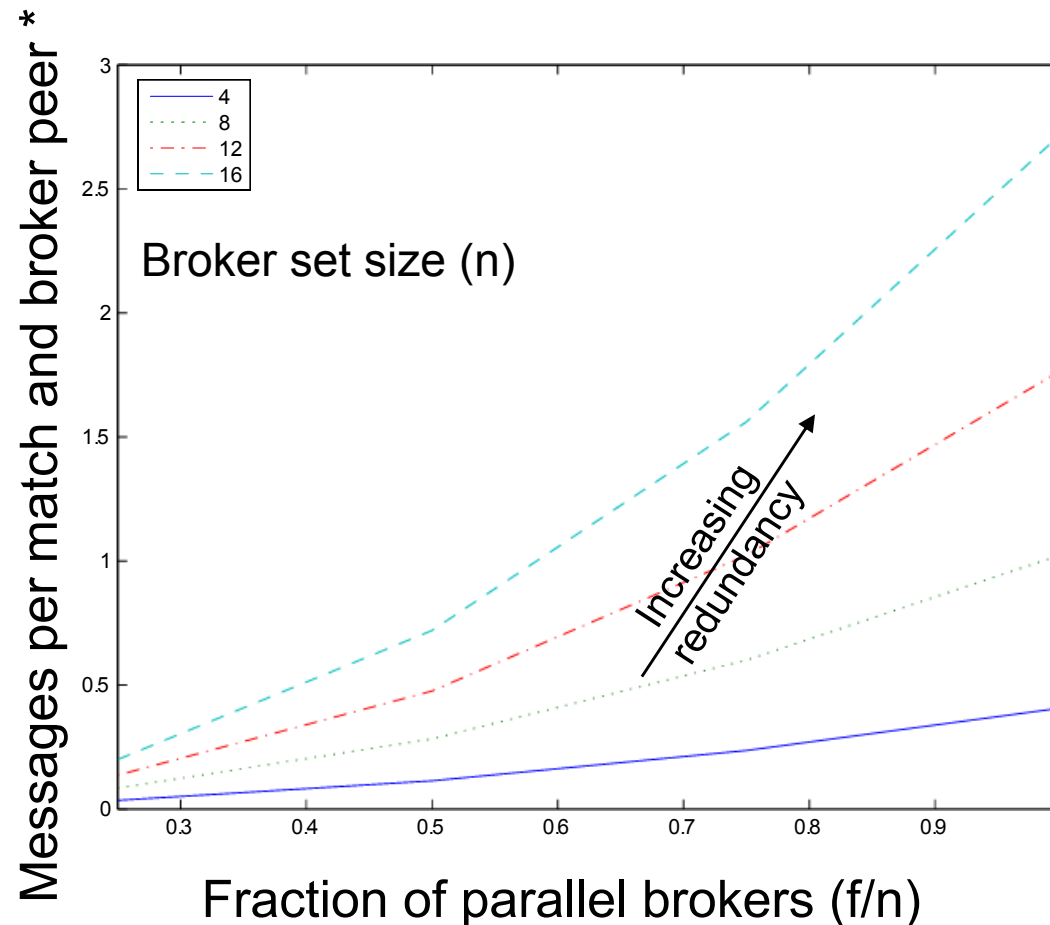
Measured security overhead:

- Overhead for SHA1 with DSA signature is ~15% of message size
- Signing / verifying messages takes 10 - 20 ms (~10% of the average RTT)
on Pentium 4 CPU 2.4 GHz, 512 MB RAM, Java VM 1.4.2

* Message size is ~1 kB



Experimental Results (2)



**Efficiency
(Redundancy
Costs, Message
Overhead)**

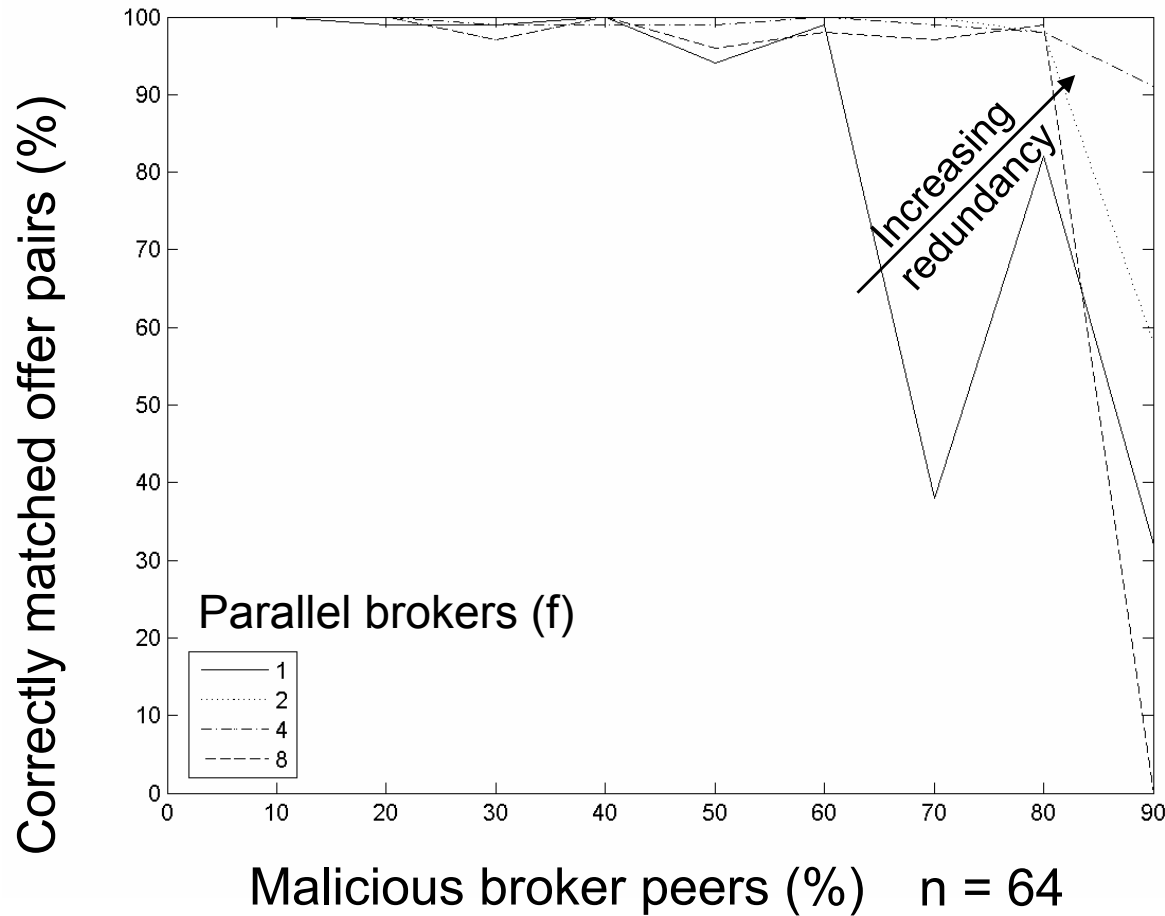
Measured security overhead:

- Overhead for SHA1 with DSA signature is ~15% of message size
- Signing / verifying messages takes 10 - 20 ms (~10% of the average RTT)
on Pentium 4 CPU 2.4 GHz, 512 MB RAM, Java VM 1.4.2

* Message size is ~1 kB



Experimental Results (3)



Reliability

- Matches analytical results quite well



Summary, Conclusions, and Future Work

- PeerMart defines a **fully decentralized double auction**
 - Applicable by any peer to trade **any service**
- Technically designed on top of a **P2P overlay network**
 - Implemented prototype uses **FreePastry** as underlying infrastructure
- PeerMart is both **technically** and **economically efficient**
 - **Scales** well even for a large number of peers trading services
- **High reliability** even in the presence of **malicious** peers
 - Achieved through redundancy

- **Future Work**
 - Punish peers that do not stay to an offered price (=> use **Reputation**)
 - Study other forms of malicious attacks (e.g., **DDoS**)
 - Run prototype in a real-world environment (e.g., **PlanetLab**)



Thank you for your attention!



=> Further information: <http://www.peermart.net/>

