University of Duisburg-Essen
Bismarckstr. 90
47057 Duisburg
Germany

# Towards Peer-to-Peer-based Cryptanalysis

Matthäus Wander, Arno Wacker, Torben Weis

Oct 14th, 2010

*6th LCN Workshop on Security in Communications Networks*

# Outline

- Introduction, Requirements
- System Model, Example Application
- Self-organizing job management
- Result verification
- Summary and Outlook

# Introduction

- **Goal**
  - Run computationally intensive cryptanalytic job
- **Stakeholder**
  - Users with interest in cryptanalysis
  - But without distributed computing infrastructure
- **Job distribution**
  - Invite friends
  - Share CPU cycles with foreigners

# Requirements

R1) CPU cycle sharing

R2) Ad-hoc setup

R3) Open & decentralized (participant-driven)

R4) Correctness

R5) Offline support

R6) Scalability & Efficiency

# Distributed Computing Paradigms

| | Sharing (R1) | Ad-hoc (R2) | Open (R3) | Correct (R4) | Offline (R5) | Scales (R6) |
|---|---|---|---|---|---|---|
| Client/Server Computing | X | | | X | X | |
| Cloud Computing | | X | | X | X | X |
| Cluster Computing | X | | | X | X | X |
| Grid Computing | X | (x) | (x) | X | X | X |
| Peer-to-Peer Computing | X | X | X | X | X | X |

# Related Work

- **CoDiP2P [Castella2008]**
  - Manager hierarchy with job master
- **Organic Grid [Chakravarti2006]**
  - Manager hierarchy with job master
- **Jalapeno [Therning2005]**
  - Manager/worker groups
- **JNGI [Verbeke2002]**
  - Monitor/dispatcher/worker groups

**→Require trusted job managers**

# Job Management

- **Management work**
  - Divide job into tasks
  - Allocate tasks
  - Monitor status, track progress
  - Collect, verify and merge results
- **Who manages the job?**
  - Job Submitter
  - Elected workers
  - All workers (self-organization)

# Requirements

R1) Cycle-sharing

R2) Ad-hoc setup

R3) Open & decentralized (participant-driven)

R4) Correctness

R5) Offline support

R6) Scalability & efficiency

Submitter manages job

# Requirements

R1) Cycle-sharing

R2) Ad-hoc setup

R3) Open & decentralized (participant-driven)

R4) Correctness

R5) Offline support

R6) Scalability & efficiency

Elected workers manage job

# Requirements

R1) Cycle-sharing

R2) Ad-hoc setup

R3) Open & decentralized (participant-driven)

R4) Correctness

R5) Offline support

R6) Scalability & efficiency

All workers manage job

# Idea

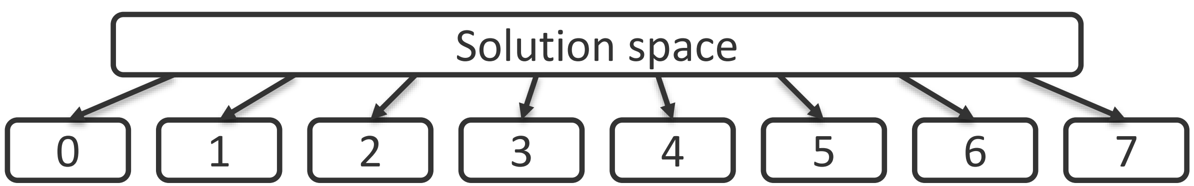
- **Share management burden among peers**
- **Do not use designated manager role**
- **Peers write job status into a distributed storage**
- **Resulting challenges**
  - Organize data structure for efficient access
  - Ensure correctness of computation
  - Ensure correctness of stored data

# System Model and Assumptions

- Scalable peer-to-peer overlay (Chord, Pastry)
- Unique participant ID [Wacker2008b]
- Secure message transport [Wacker2008b]
- NAT traversal [Wacker2008a]
- Accounting of work performed [Garcia2005] [Turner2004]
- Distributed storage (Distributed Hashtable)

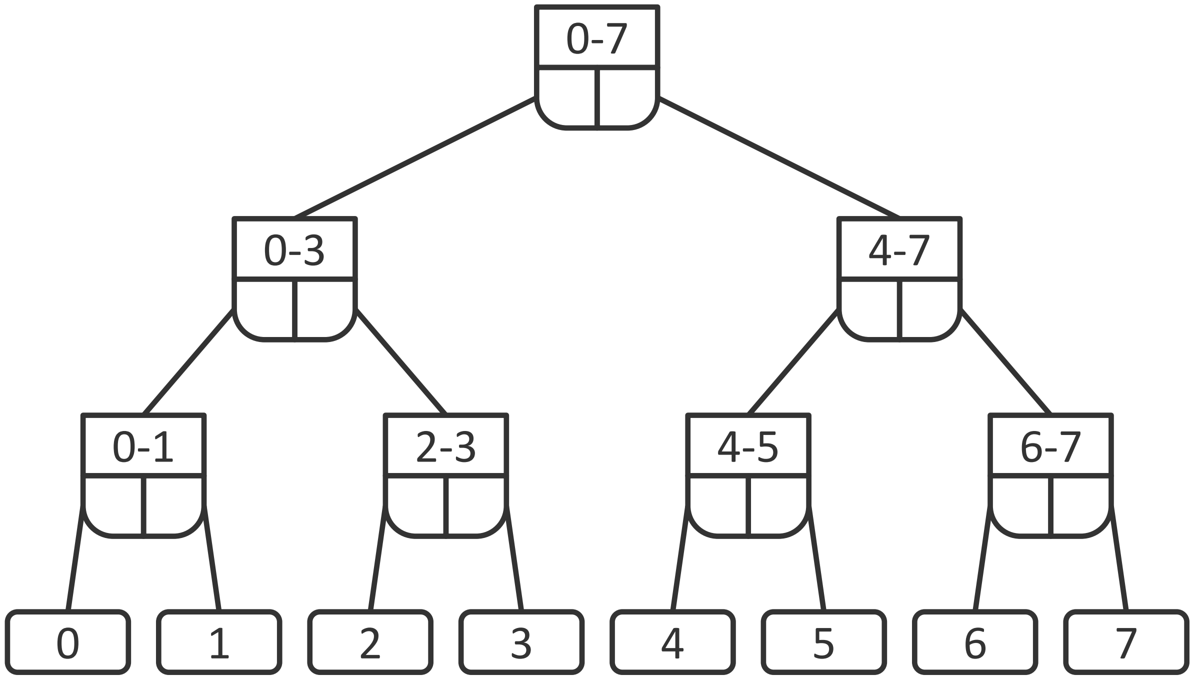
# Example Application

- **Brute-force attack on symmetric-key cipher**
- **Represents class of search problems**
- **Input: ciphertext, cipher being used**
- **Solution space: all possible keys**
  - Decrypt ciphertext
  - Rate result with score function
- **Divide solution space into task blocks**



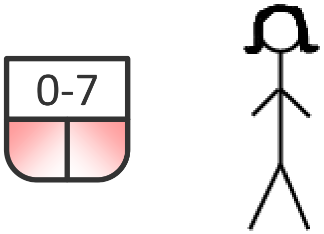
# Self-organizing Job Management

- Structure task list as tree
- Each **object** stored on different peer



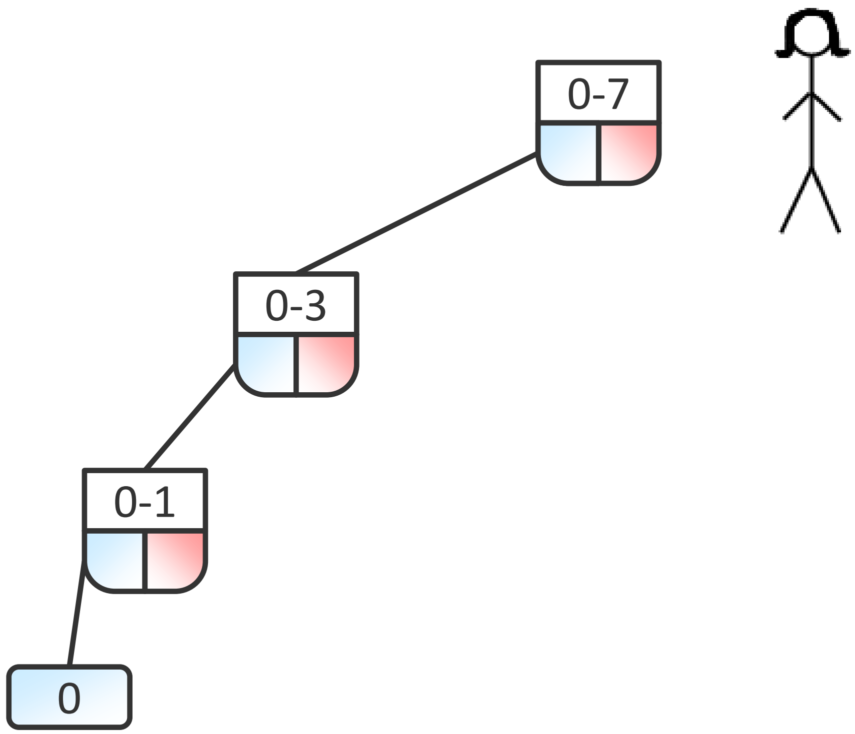
# Self-organizing Job Management (2)

- **Participating peers traverse tree to get task**
- **Construct tree on demand**
  - Divide
  - Compute
  - Merge



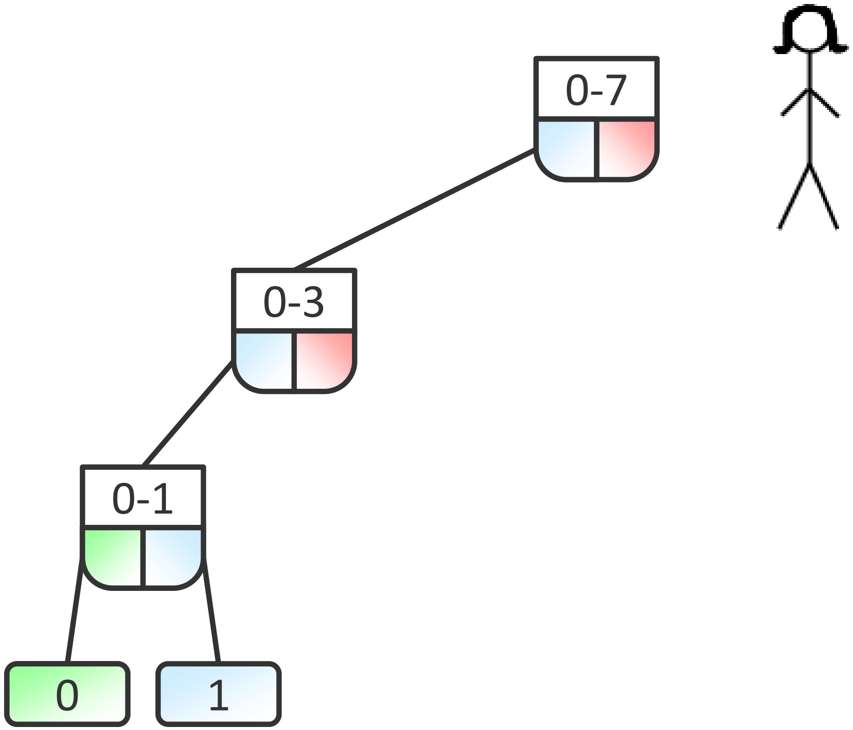
# Self-organizing Job Management (2)

- **Participating peers traverse tree to get task**
- **Construct tree on demand**
  - Divide
  - Compute
  - Merge



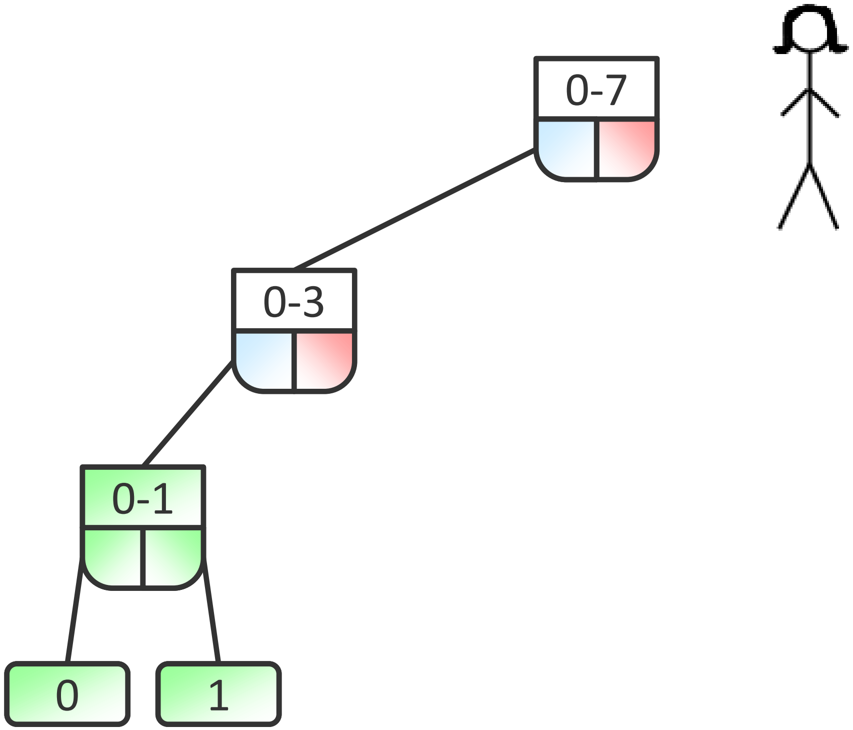
# Self-organizing Job Management (2)

- **Participating peers traverse tree to get task**
- **Construct tree on demand**
  - Divide
  - Compute
  - Merge



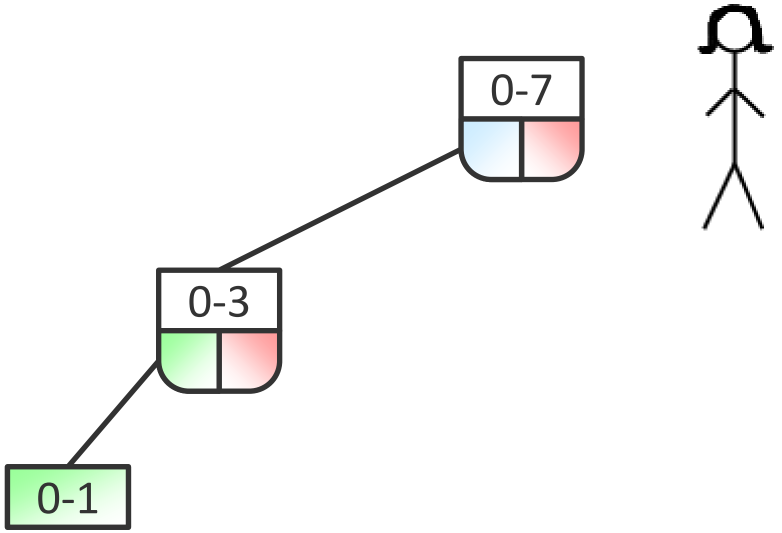
# Self-organizing Job Management (2)

- **Participating peers traverse tree to get task**
- **Construct tree on demand**
  - Divide
  - Compute
  - Merge



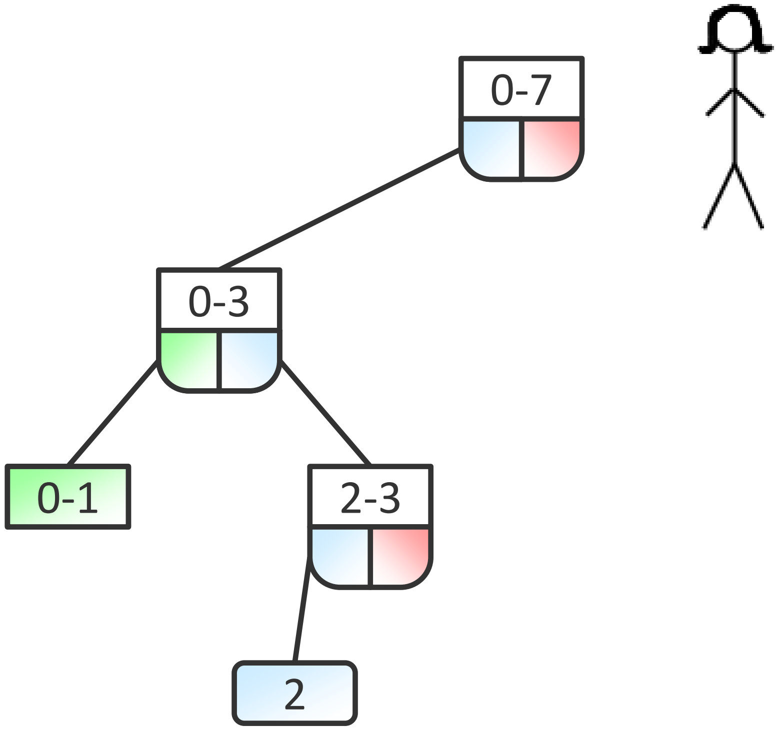
# Self-organizing Job Management (2)

- **Participating peers traverse tree to get task**
- **Construct tree on demand**
  - Divide
  - Compute
  - Merge



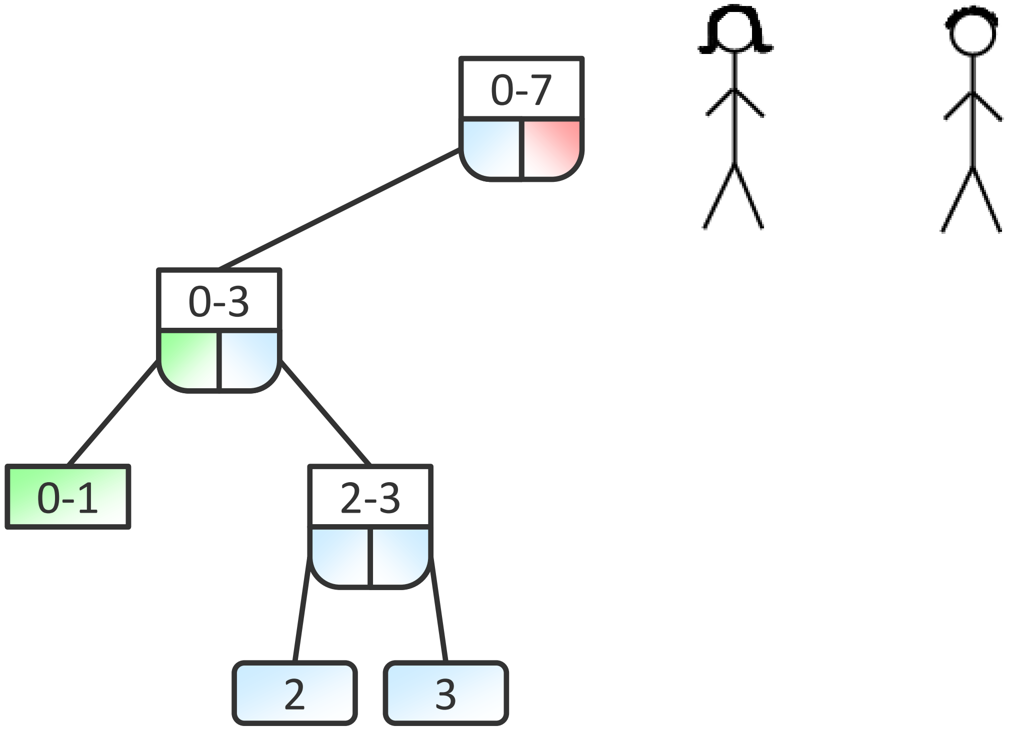
# Self-organizing Job Management (2)

- **Participating peers traverse tree to get task**
- **Construct tree on demand**
  - Divide
  - Compute
  - Merge



# Self-organizing Job Management (2)

- **Participating peers traverse tree to get task**
- **Construct tree on demand**
  - Divide
  - Compute
  - Merge

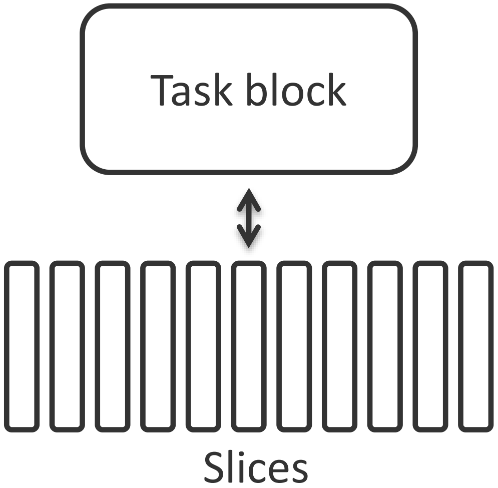


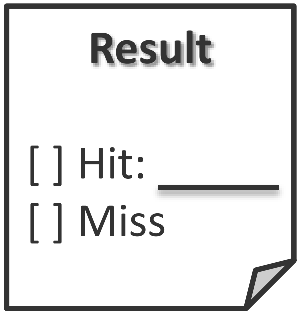
# Self-organizing Job Management (3)

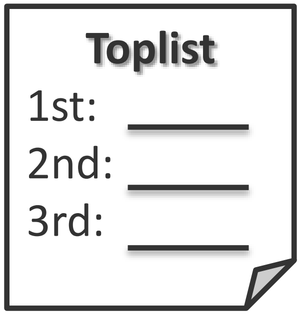
- **Task allocation**
  - Peers mark tasks as allocated
  - But allocations are not exclusive locks
  - Other peers may ignore allocations
- **Tree traversal**
  - Avoid querying occupied objects
  - Avoid unnecessary large trees
  - Depth-first traversal with frequent left turns

# Result Verification

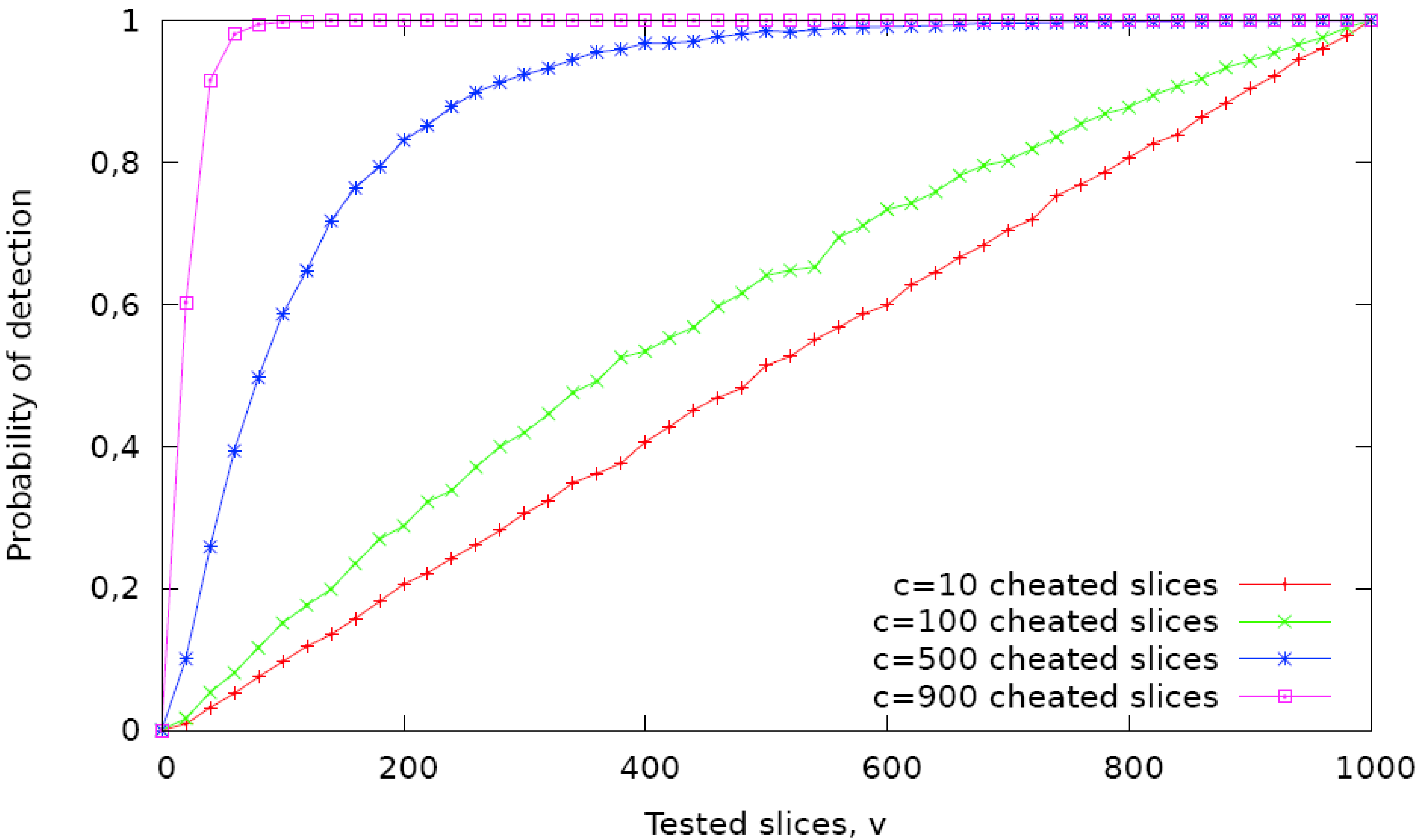
- **Cheat attempts for search problems**
  - Peer claims to have found solution (false positive)
  - Peer claims subspace does not contain solution (false negative)
- **Find opportunistic cheaters efficiently**



# Result Verification (2)



# Removal of Partial Results

- **Large tree requires bandwidth for maintenance**
- **Tree size would scale with number of peers**
  - … if unneeded subtrees were removed
- **Peers not allowed to remove unneeded subtrees**
  - Too risky to lose progress
  - Even with verification
- **Job submitter removes junk from time to time**
  - If she doesn‘t: some maintenance overhead
  - Still allowed to go offline

# Distributed Storage

- **Special requirements not provided by plain DHT**
- **Adaptive replication**
  - Ensure consistency
  - Scale with number of read operations [Knoll2008]
- **Access model: read all, append all, modify own**
- **Prevent unauthorized modifications**
- **Soft state: remove data if not refreshed**

# Summary and Outlook

- Peer-to-peer computing for CPU **sharing** (R1)
- **Self-organizing** without infrastructure setup (R2)
- Without **provider** or administration (R3)
- Deals with **opportunistic** peers (R4 partly)
- Allows job submitter to go **offline** (R5)
- Considers **scalability** & **efficiency** so far (R6)
- Future work
  - Large-scale evaluation
  - More complex applications

# References


- **[Chakravarti2006]: Organic Grid**
  - Arjav J. Chakravarti, Gerald Baumgartner, Mario Lauria: **Self-Organizing Scheduling on the Organic Grid**. Int. Journal of High Performance Computing Applications, 2006, vol. 20, no. 1
- **[Castella2008]: CoDiP2P**
  - D. Castellà, I. Barri, J. Rius, F. Giné, F. Solsona, F. Guirado: **CoDiP2P: A Peer-to-Peer Architecture for Sharing Computing Resources**. Int. Symposium on Distributed Computing and Artificial Intelligence, 2008

# References

- **[Therning2005]: Jalapeno**
  - Niklas Therning, Lars Bengtsson: **Jalapeno: secentralized grid computing using peer-to-peer technology**. 2nd conference on Computing frontiers, 2005
- **[Verbeke2002]: JNGI**
  - Jerome Verbeke, Neelakanth Nadgir, Greg Ruetsch, Ilya Sharapov: **Framework for Peer-to-Peer Distributed Computing in a Heterogeneous, Decentralized Environment**. 3rd International Workshop on Grid Computing, 2002

# References


- **[Knoll2008]: Replication (Repl1, Repl2)**
  - Mirko Knoll, Haitham Abbadi, Torben Weis: **Replication in Peer-to-Peer Systems**. 3rd Int. Workshop on Self-Organizing Systems (IWSOS), 2008
- **[Wacker2008b]: Authentication (Auth)**
  - Arno Wacker, Gregor Schiele, Sebastian Schuster, Torben Weis: **Towards an Authentication Service for Peer-to-Peer based Massively Multiuser Virtual Environments**. Int. J. Advanced Media and Communications, Inderscience Enterprises Ltd., 2008

# References


- **[Garcia2005]: Currency**
  - Flavio D. Garcia and Jaap-Henk Hoepman: **Off-line Karma: A Decentralized Currency for Peer-to-peer and Grid Applications**. 3rd Applied Cryptography and Network Security (ACNS), 2005
- **[Turner2004]: Currency**
  - David A. Turner and Keith W. Ross: **A Lightweight Currency Paradigm for the P2P Resource Market**. 7th Int. Conference on Electronic Commerce Research, 2004

# References

- **[Wacker2008a]: NAT traversal (NAT)**
  - Arno Wacker, Gregor Schiele, Sebastian Holzapfel, and Torben Weis: (Demo) **A NAT Traversal Mechanism for Peer-To-Peer Networks**. 8th Int. Conference on Peer-to-Peer Computing (P2P), 2008
- **[Knoll2009]: Bootstrapping (IRC)**
  - Mirko Knoll, Matthias Helling, Arno Wacker, Sebastian Holzapfel and Torben Weis: **Bootstrapping Peer-to-Peer Systems Using IRC**. 5th Int. Workshop on Collaborative Peer-to-Peer Systems (COPS), 2009