

Generalized Optional Locking in Distributed Systems

Thomas Schöbel-Theuer, University of Stuttgart,
Germany

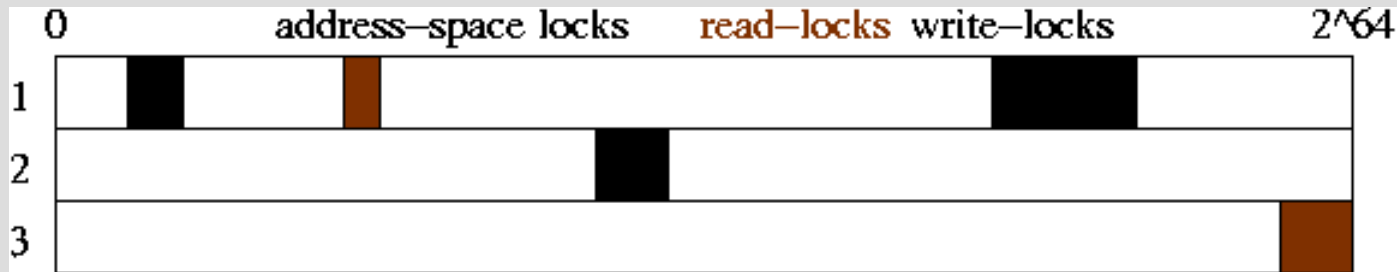
schoebel@informatik.uni-stuttgart.de

www.athomux.net

Outline

- Problem: mutual exclusion
 - *very slow* in distributed systems (esp. fine-grained)
 - most distributed systems try to avoid / circumvent it
 - efficient solution => uniform programming models
- Idea: exploit *spatial locality* of locks
 - 2 kinds of locks: obligatory / optional
 - *negotiate* the *size* of optional locks dynamically
(in difference to *hierarchical* locking)
- Performance Study => high speedups possible
- Further Result: communication paradigm is special case of optional locking
- Future work / Conclusions

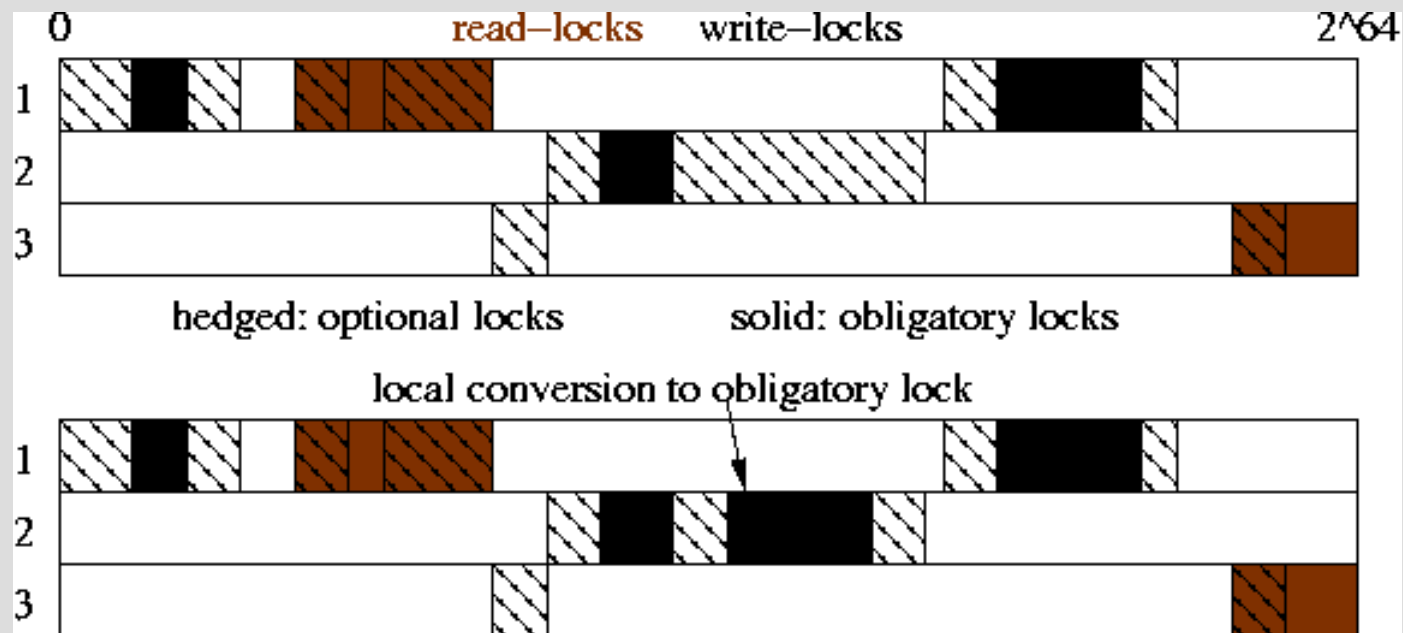
Preparation



- Don't use substitute objects, e.g. semaphore
- Issue lock requests *directly* on the memory region occupied by a data object
 - similar to Unix `lockf()` or `fcntl()` locking
 - characterized by (startaddress,length,locktype)
- => **locality of access** behavior translates to locking *directly*

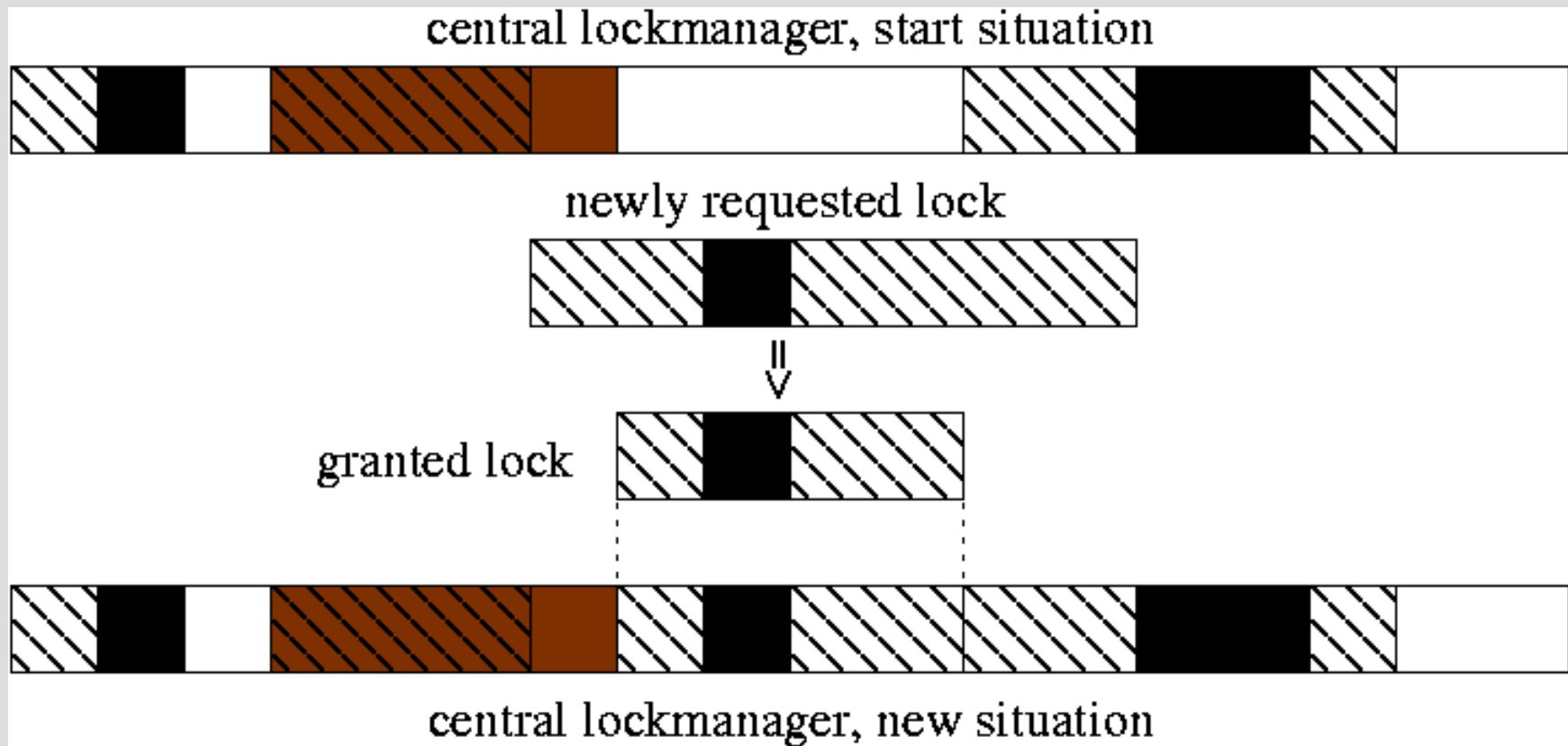
Optional Locks

- 2 types: obligatory / optional
- Optional lock is *locally convertible* to obligatory lock at any time, no network traffic

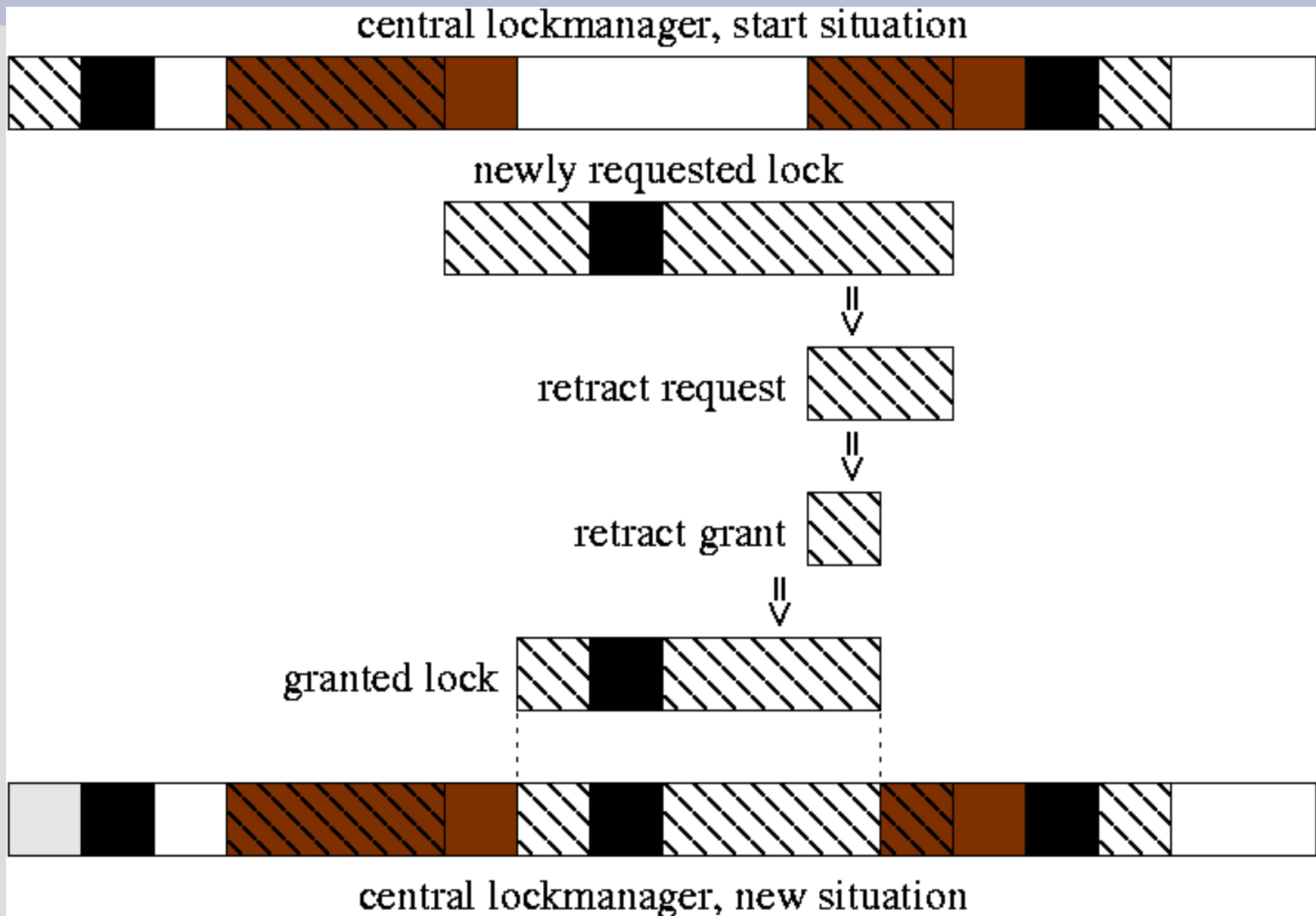


Negotiation of Optional Locks

Scenario: central lockmanager



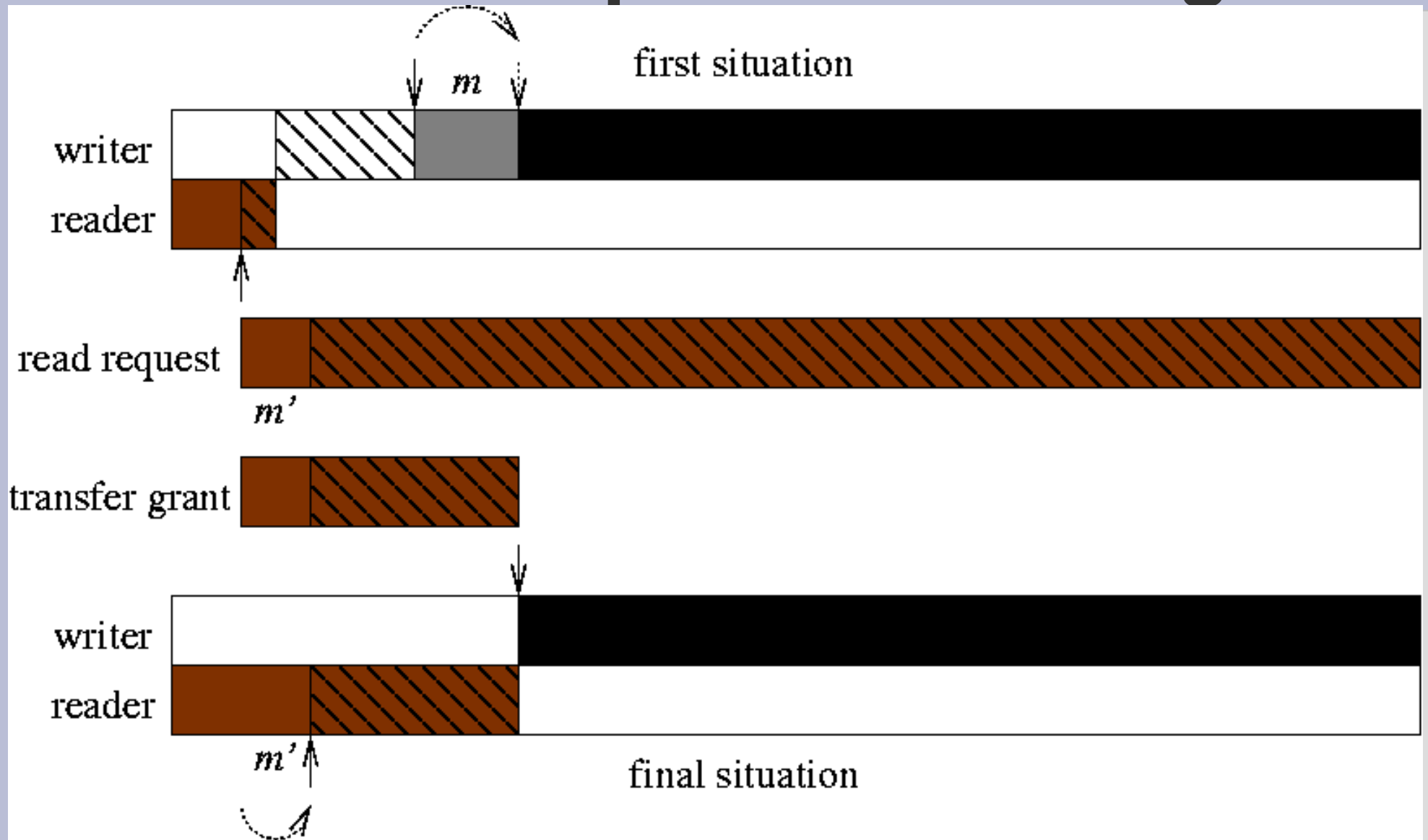
Retraction of Optional Locks



Performance Study

- Experiments: TPC-like database benchmarks on PostgreSQL => observed locking patterns
- Simulator: distribute n server threads to $m \leq n$ virtual network sites, count # of lock/retract requests for different negotiation strategies
- Results: speedup factors from ~ 30 to ~ 180 (speedups relative to known obligatory lock prefetching/caching: from ~ 0.93 to ~ 5.4)
- More details => paper

Message Passing as Special Case of Optional Locking



Consequences

- Message Passing paradigm is a special case of optional locking
- Merged optional locks correspond to coalesced messages
- => efficient solution of both mutual exclusion and message passing is *possible* in uniform way
- Bridging the bottleneck of Distributed Systems no longer “special”?
- Distributed Shared Memory (DSM) should be *reconsidered* (when combined with optional locking)

Future Work

- *Symmetric* optional locking (no central server)
- Reliability, failure resilience, security, ...
- New applications, formerly unsuitable for distributed computing?
- Practical experience ... (a lot missing)
 - Distributed databases?
 - Distributed operating systems / middleware:
Athomux prototype => meta-middleware based on
LEGO principle ==> www.athomux.net
 - High-performance / cluster computing
 - Other ideas => contact me

Contribution / Conclusions

- Automated negotiation of locking *granularities*
- High speedups for mutual exclusion
- Emulation of message passing, probably of other synchronization scenarios
 - => uniform programming models possible
- Details, client-server algorithm + proof
 - => paper, www.athomux.net
- Further research necessary
 - e.g. negotiation strategies, thrashing prevention, practices...