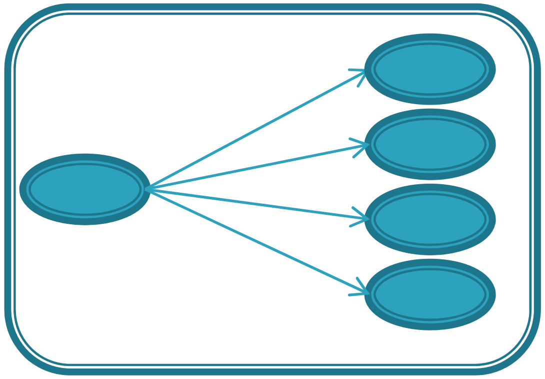
Process-Oriented Collective Operations

John Markus Bjørndalen
Adam T. Sampson

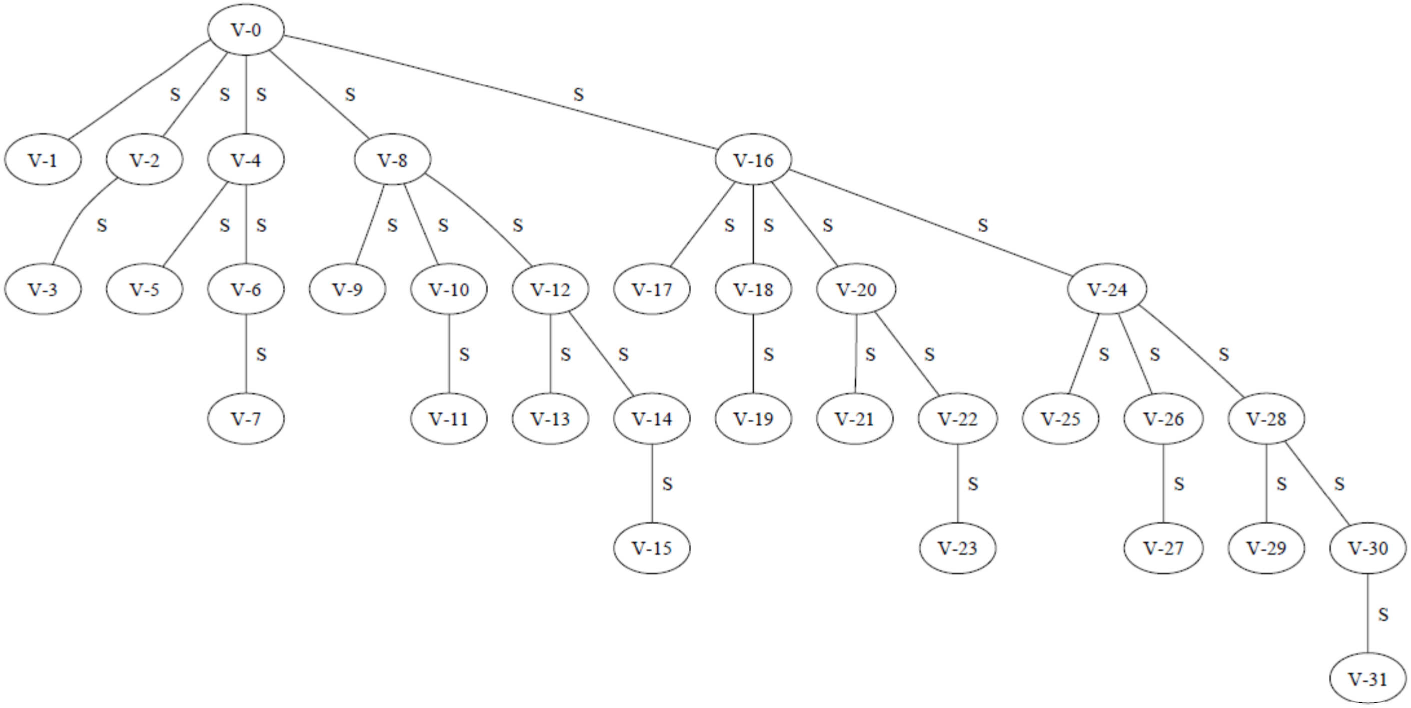
Collective Operations

- Operations involving a specified group of processes
- Example: broadcast



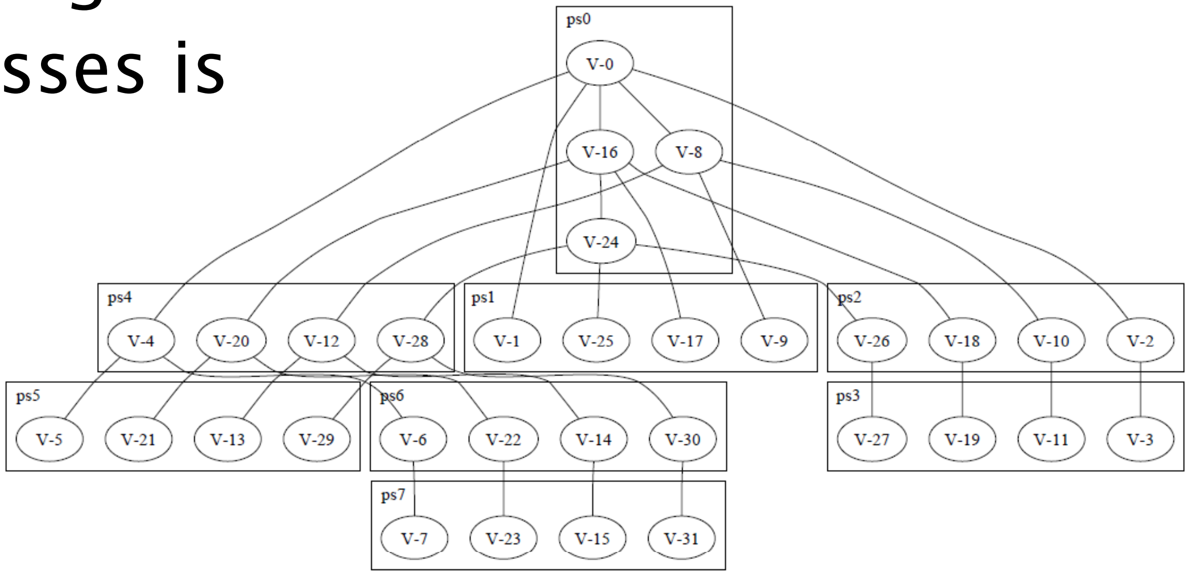
LAM-MPI reduction tree

- To improve scaling and reduce latency, LAM uses a binomial tree:



Reduction tree mapped to cluster nodes

- 8 SMP computers with 4 processors each
- Default mapping
- Moving processes is only a partial solution



Solution

- Configuration system:
  - Map trees/algorithms to given cluster and application (PATHS and CoMPI)
  - Minimizing network messages not always the best performing configuration!
    - Can get non-intuitive results due to overlooking factors in theoretical models

Process-oriented Collective Operations

- Applications for CoSMoS project
- Learn from MPI (OpenMPI)
  - First approximation for cluster-wide process oriented applications
  - MPI algorithms a good first approximation

Process-oriented Collective Operations

- CSP-based configuration system
  - CSP-based language for algorithms and mapping
  - Improve configurations compared to OpenMPI (tune application, cluster and configuration)
  - Improve specification of parallel properties (run-time knows more)
  - May be useful for configuration of MPI implementations

First experiences

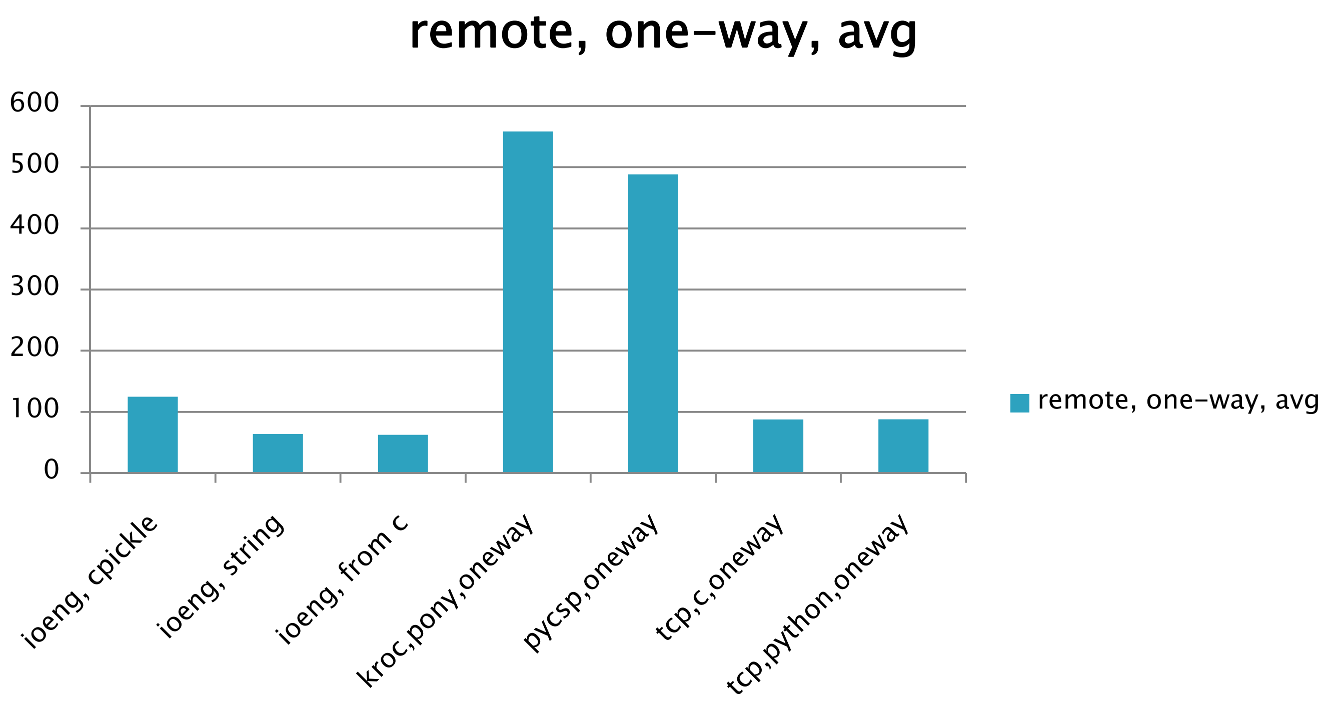
- Sequential operations send/receives
  - trivial
- Nonblocking code encountered so far
  - easily expressed using PAR
- PyCSP code more concise than OpenMPI
- Opportunity for improved parallelism:
  - Not easy to do (wait+do || wait+do) in OpenMPI

Implementation

- Prototyped collective operations in PyCSP
- Network communication library for PyCSP and occam-π (trap)
  - Nodes and Ports
    - Send/receive – similar to buffered channels
  - Buffered, asynchronous communication
  - Supports thousands (millions?) of channels between nodes
  - Only one kernel thread for message transfer
  - Serialization only if needed
    - Raw byte strings at the transport level

Network latency, node-to-node

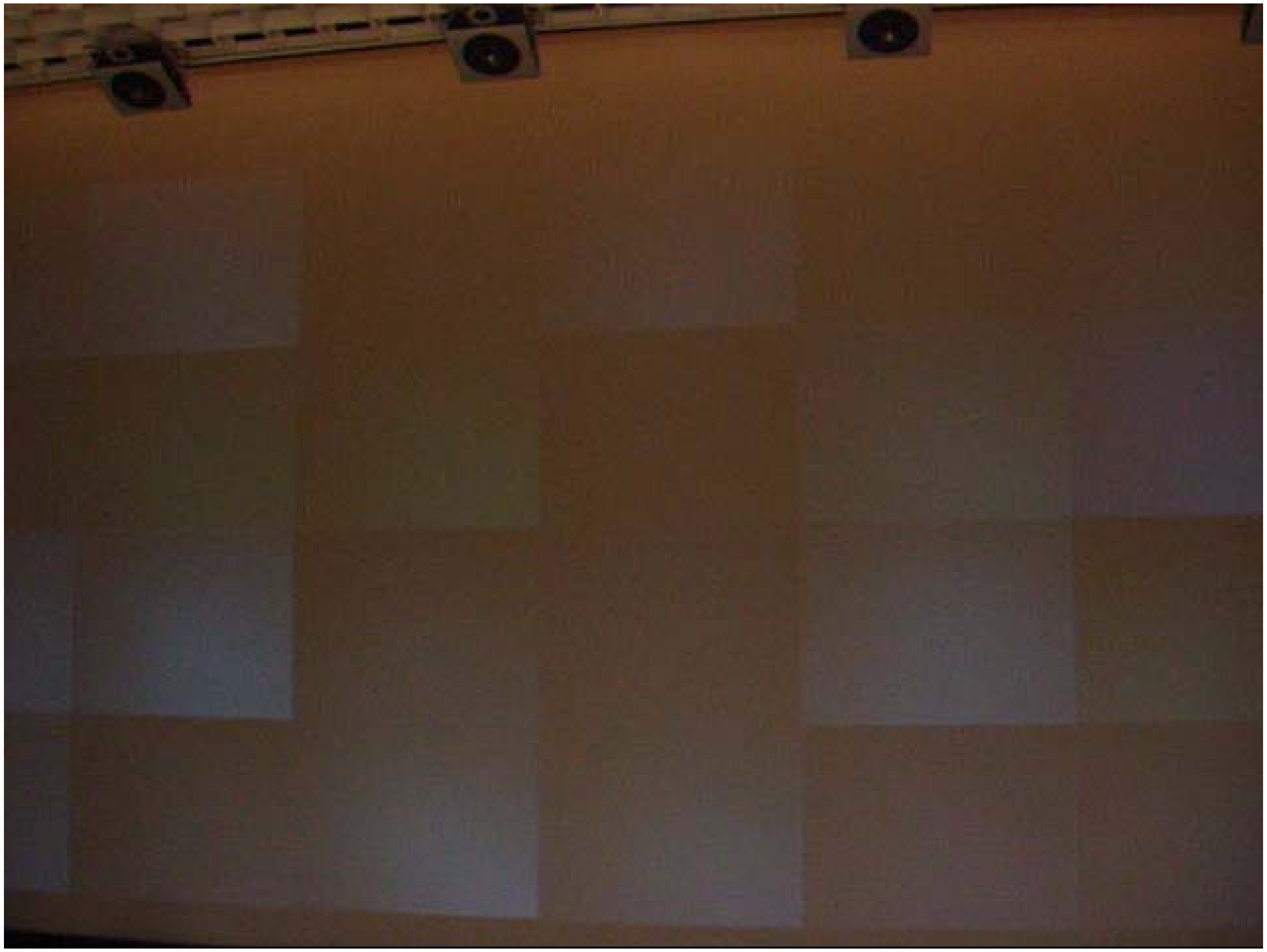
- Informal benchmark



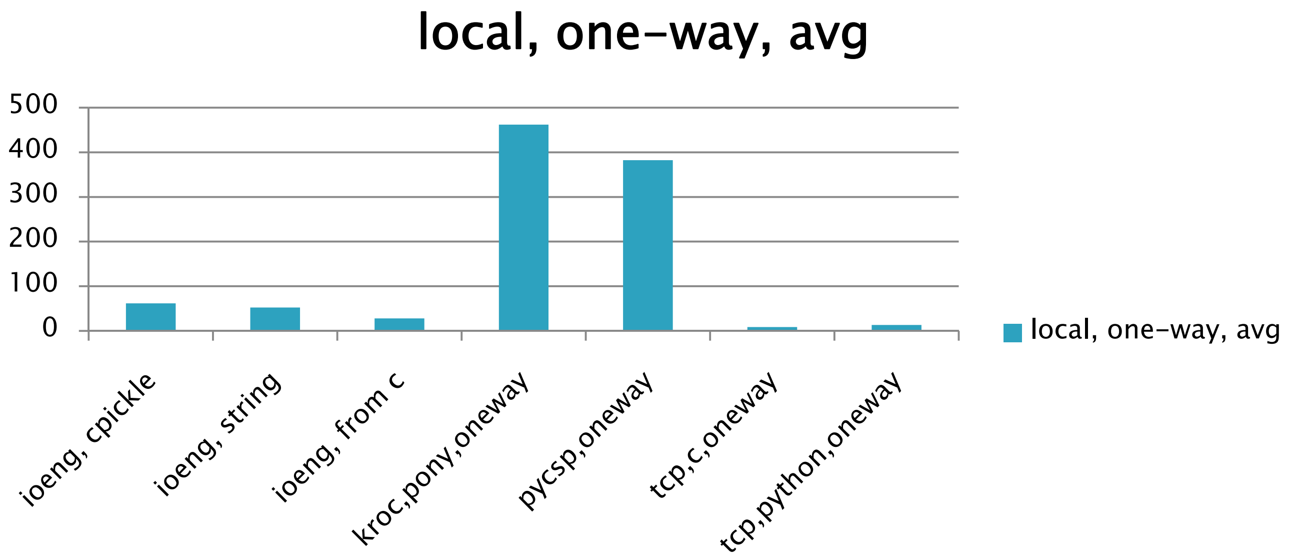
Summary

- Early work
- Expressed MPI group operations as CSP programs with higher level of parallelism than OpenMPI code
  - Techniques in the paper
- Light-weight message transfer layers for occam-π and PyCSP
  - Implementations have been used (Occoids on the display wall in Tromsø)

Occoids on the wall (using trap)



Network latency, localhost


local, one-way, avg
ioeng, cpickle
ioeng, string
ioeng, from c
kroc,pony,oneway
pycsp,oneway
tcp,c,oneway
tcp,python,oneway
🟦 local, one-way, avg