OSF Research Institute

Concurrent Remote Task Creation

Dejan S. Milojičić, David Black and Steve Sears

The OS Advisory Board Meeting

Cambridge, MA, Feb. 2-3 1995

Problem statement

- poor performance of Mach remote task address space creation, common denominator for the support of:
    - new address space creation (*rexec* and *rspawn*)
    - parallel applications (*rspawn-multi*, *rfork-multi*)
    - process migration
- multiple task creation has increased performance, fault-tolerance and scalability requirements

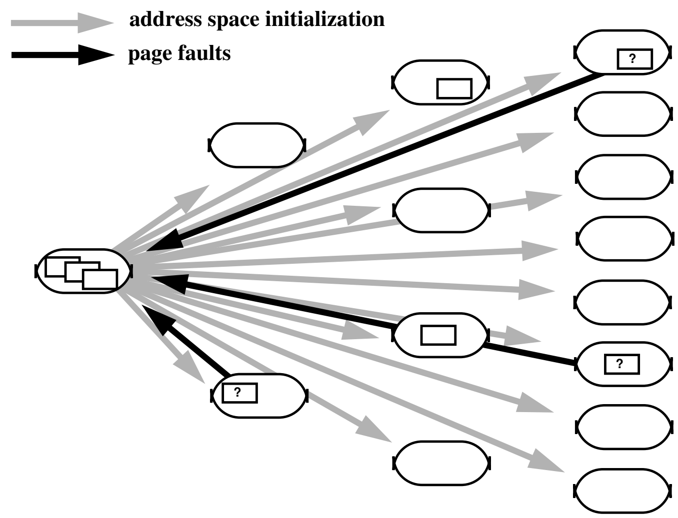
Sources of Problem

- network IPC, addressed by new DIPC ✔
- distributed VM, will be addressed by new EMMI ✔
- unoptimized algorithm for remote task creation ↵
- no support for multiple remote task creation ↵

Do we really need optimized CRTC?

- creation/termination phase
- paging from all nodes (large address spaces)
- performing I/O initialization at one site

Internode Traffic for Centralized CRTC



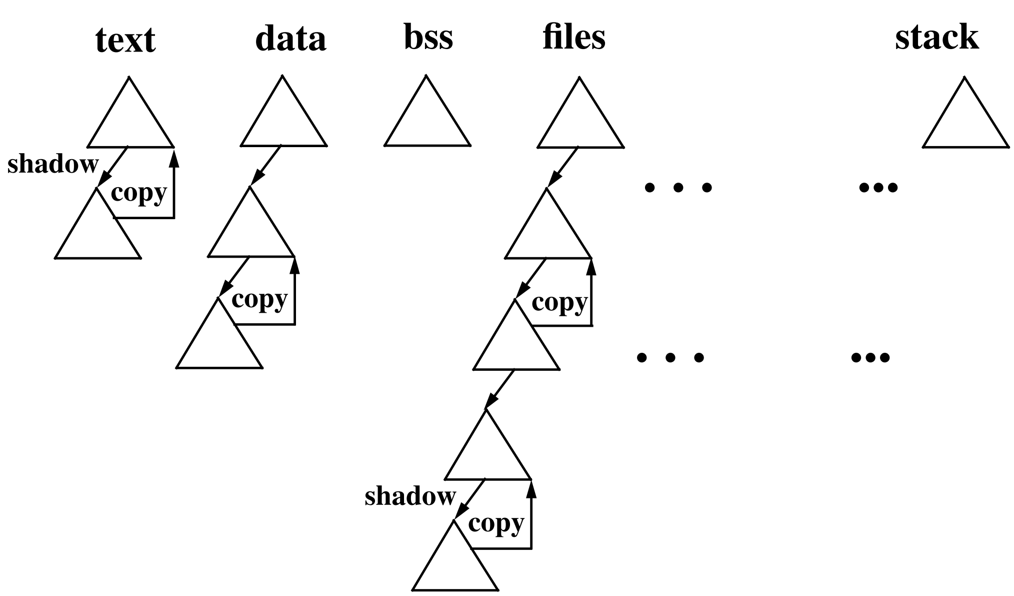
Presentation Outline

- overview of Mach VM (local and distributed)
- improvements to single remote task creation
- hierarchical remote task creation
- handling (avoiding) COW over network
- implementation, status, performance
- future work
- related work
- conclusion

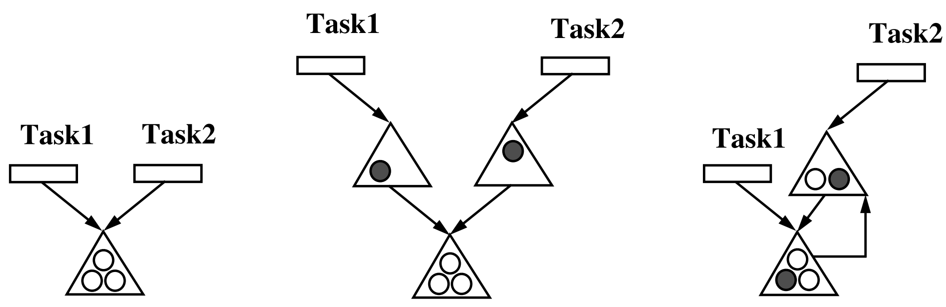
Mach VM

- built around VM object which is kernel cache for pages
- external memory manager
- VM copy optimizations, “Copy-On-Write” (COW)
- expanded to distributed, “Copy-On-Reference” (COR)

Address Space of a Mach Task

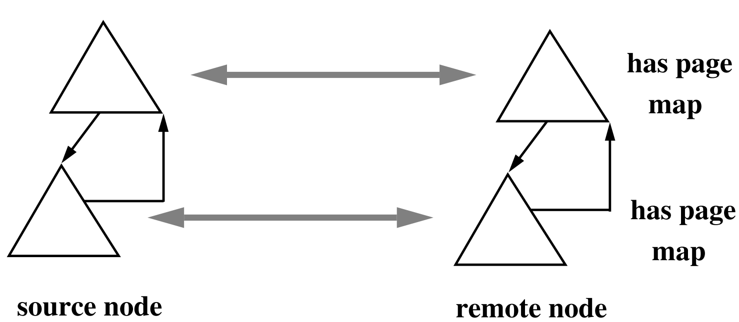


COW Optimization



(a) Read-only sharing (b) Symmetrical COW (c) Asymmetrical COW

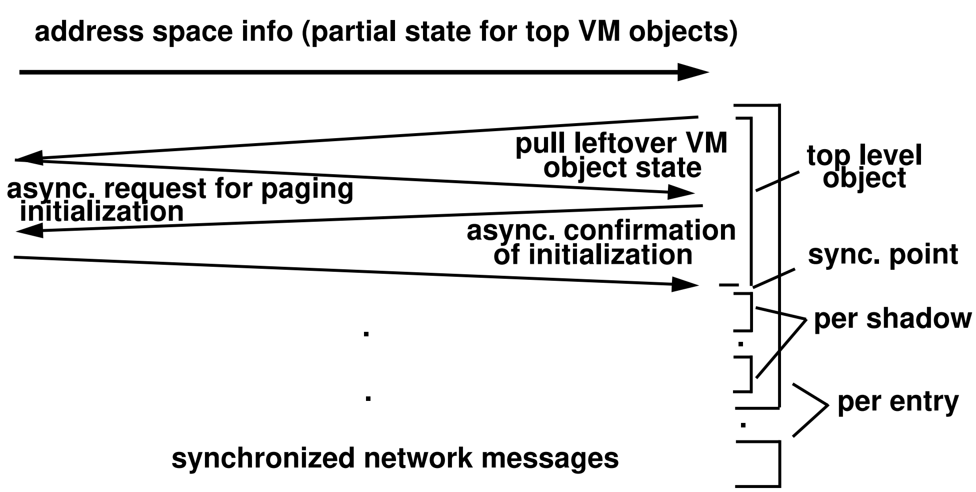
COR Optimization



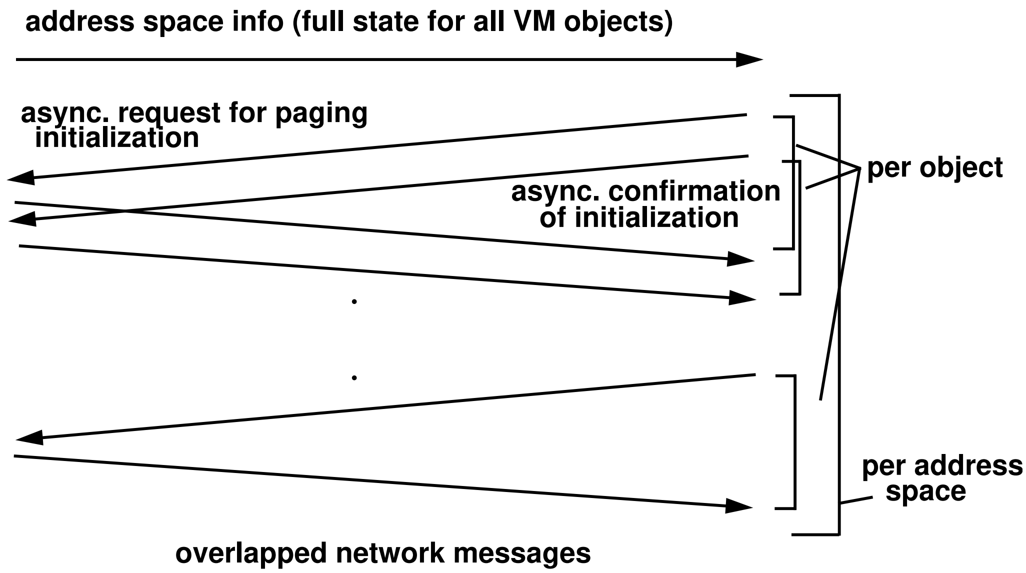
Improving Single Rem. Task Creation

- any delay is multiplied in concurrent case
- improvements consists of:
  - sending more state per each object eagerly
  - sending the whole shadow chain eagerly
  - eliminating all synchronization points
- hard to batch object initialization, so overlap them

Unoptimized Address Space Transfer



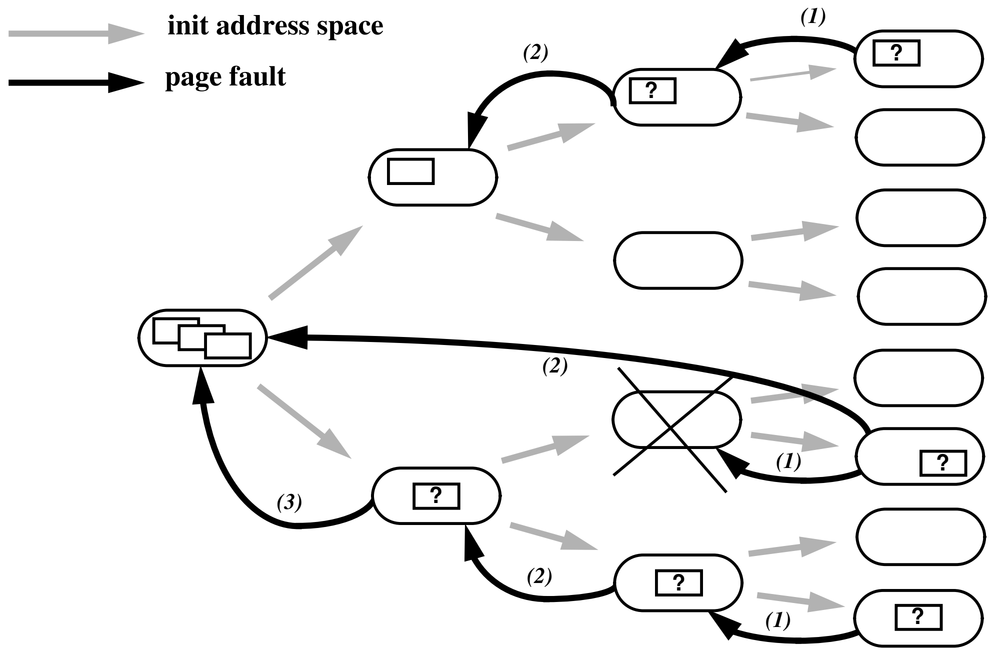
Optimized Address Space Transfer



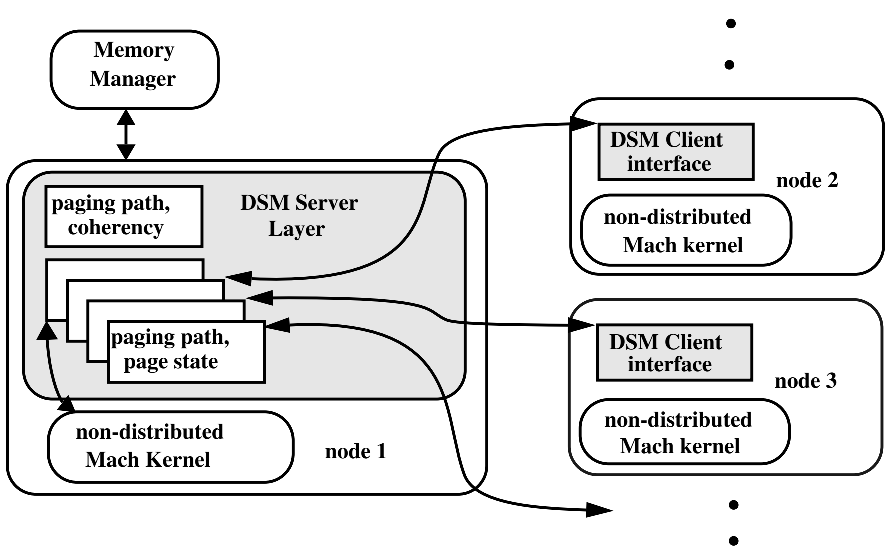
Hierarchical Remote Task Creation

- performance
- scalability
- hooks for fault-tolerance

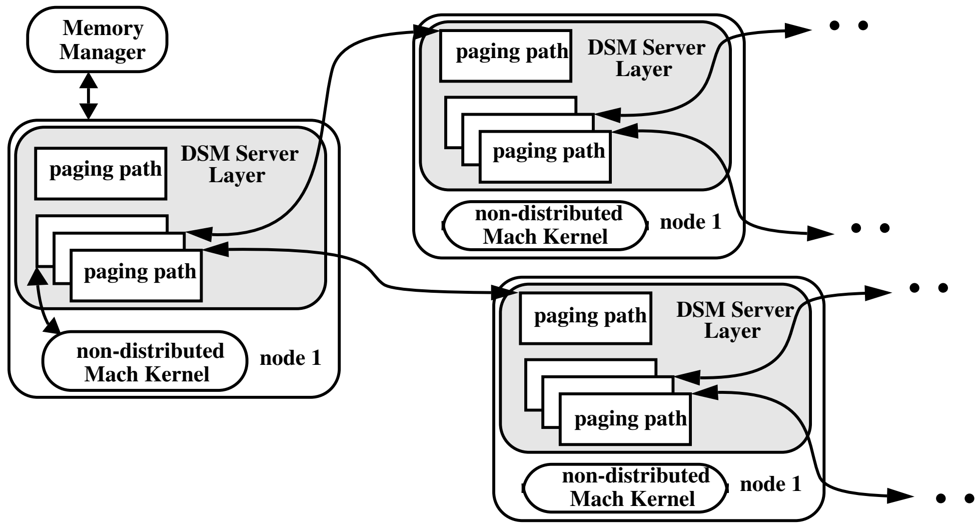
Hierarchical XMM



Centralized VM



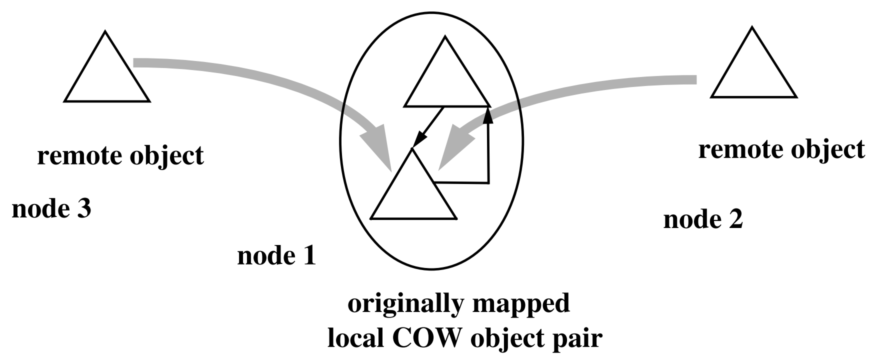
Hierarchical VM



Handling COW in CRTC

- prevent the case of pushing a page into a copy object over network
- prevent creation of new copy objects over network
- introduction of COR objects:
    - they are read-only, no need for coherence
    - do not page-out, pages brought from nodes higher in the tree hierarchy

Avoiding Distributed COW



Implementation

- developed on a cluster of PCs connected through ethernet
- measurements performed on a 64-node Intel Paragon computer
- software base is OSF/1 AD 1.0 and Mach nmk17.2
- result of one-man effort during previous year

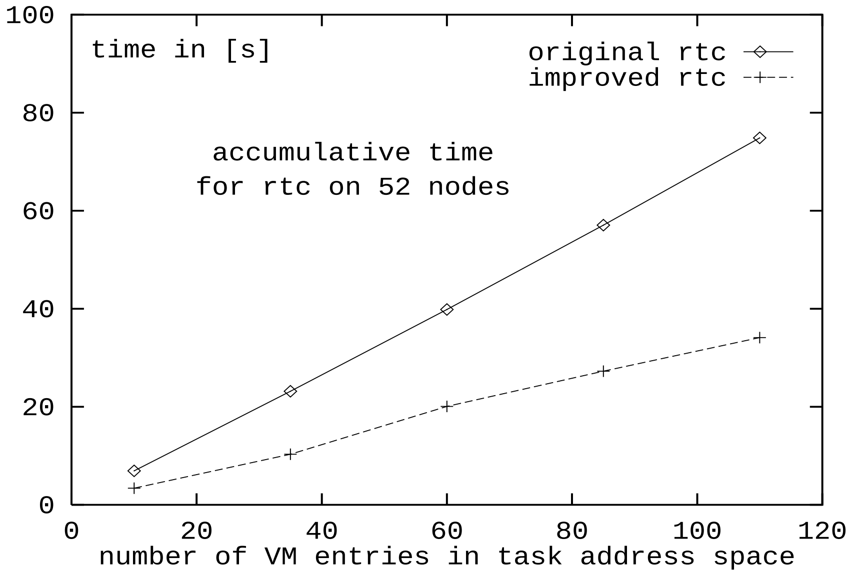
Status

- creation of remote tasks and tree hierarchies
- testing of hierarchical paging performed on PCs, in progress on Paragon
- profiling of distributed and local VM activity (COW/COR)
- work in progress on merging COW and COR objects

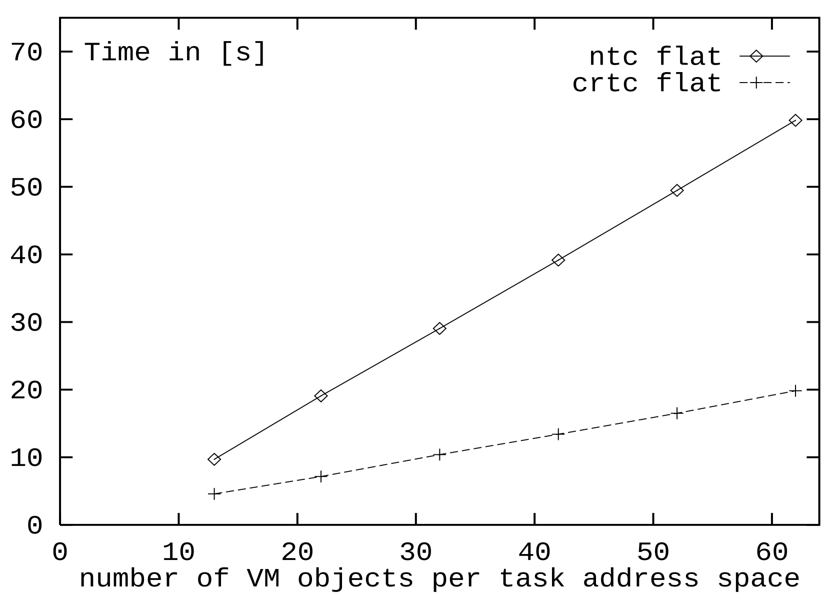
Performance, single RTC case

- 30% performance improvement for eager copy of XMM objects
- 14% performance improvement for eager copy of one additional shadow in the chain
- the rest is due to the overlap of object initialization

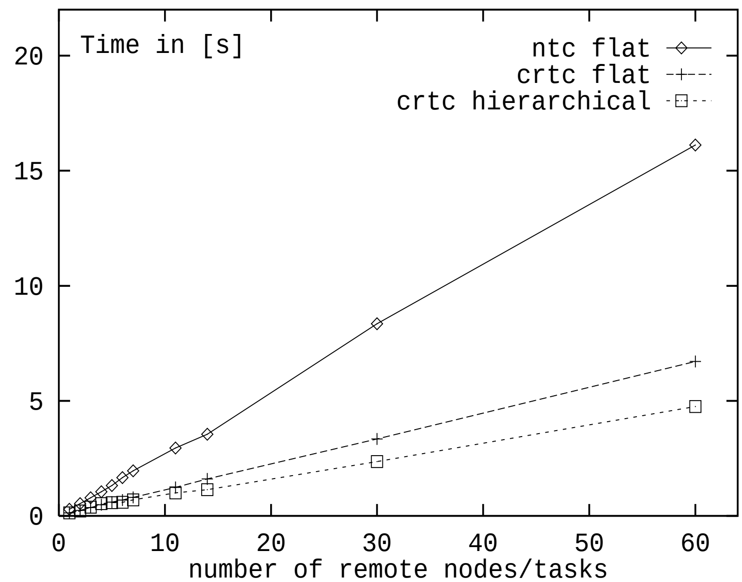
RTC v. Number of VM Entries



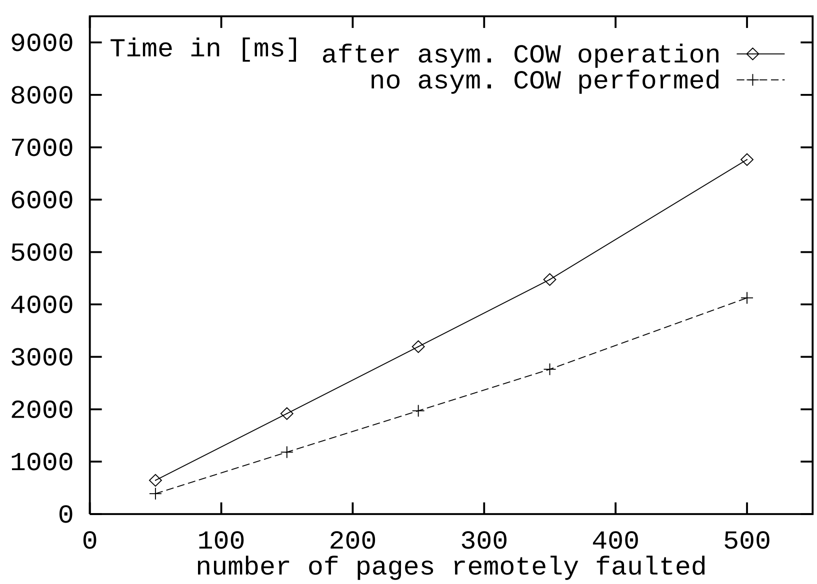
RTC v. Number of VM Objects



Hierarchical Remote Task Creation



Distributed COW



Future Work

- experiments with wider trees, currently size is 2 which leads to tree depth of *e.g.* 5 for 64 nodes, 9 for 1024;
- repeat experiments with new DIPC
- generalize work on relationship between COW and COR
- heuristic optimizations on shadow chain transfer:
  - lazy evaluate shadow chains
  - perform local COW obligations eagerly
- make experiments with real-world applications

Related Work

- process migrations, eager:
  - Chorus, MOS(IX), Locus TCF
- migrations, more:
  - Sprite (flushing), Accent (COR), V Kernel (pre-copy)
- Barrera’s original work
- NX/2
- TNC
- User space task migration

Conclusions

- performance is improved over the earlier versions:
- scalability is improved
- there are hooks for introducing fault-tolerance
- underlying network IPC mechanism plays crucial role in performance

Availability

- this work will hopefully propagate to standard OSF Mach tree, and potentially become freely available
- presentation can be accessed through anonymous ftp at:

  riftp.osf.org: pub/task_migration/advisory.pres.ps

- presented at the Mach/Chorus Workshop at OSDI
- will be presented at the forthcoming IPPS
- for more info, please contact us at:

[dejan,dlb,sjs]@osf.org