Bibliography


<table>
<thead>
<tr>
<th>Reference</th>
<th>Author(s)</th>
<th>Title</th>
<th>Venue</th>
</tr>
</thead>
</table>
BIBLIOGRAPHY


[Cond93a] CONDICT, M. Personal communication 1993a.


Index

Accent, 13, 16, 32, 58
  migration performance, 67
accounting, 6, 20, 75, 77, 79, 118
  on network IPC, 79, 80
  on VM and DSM, 80
ALT, 63, 64, 66, 94, 95, 97, 99, 102
Amoeba, 2
applications, 75
  ALT, see ALT
classification, 85
Dump benchmark, 86
for "worm" program, 15
granularity, 11
integrity, 49, 55, 61, 120
Jigsaw, 67, 86, see Jigsaw
Linda, 3
Linpack, 64
matrix multiplication, 63, 70
on UNIX, 40, 41, 46, 118
p4, 3
parallel make, see parallel make
Parform, 3
PCN, 3
profiling, 7, 77, 85–87, 118
PVM, 3, 94, see PVM
traveling salesman problem, 70
WPI benchmark suite, see WPI benchmark suite

Arcade, 19

BirliX, 49
capabilities, 32, 33, 49
  accounting on messages, 80, 85
distributed space, 38
  for kernel objects, 51, 54, 56, 57
migration of, 51, 53, 57, 59, 60, 64, 66, 70, 110
Cedar, 28
Charlotte, 13, 16, 22, 47, 67, 115
  migration performance, 67
checkpointing, 10, 17, 71
Chorus, 2, 34, 35, 46, 70
custers, 2, 5, 69, 70, 103, 110, 114, 117
Condor, 17, 67, 71, 72
data transfer policy, 13
  copy-on-reference, 14, 57, 59, 61, 64, 66, 72
  copy-on-reference-extend, 59
eager copy, 13, 16, 50, 114, 115
flushing, 13, 14, 16, 61, 71, 72
precopy, 13, 18, 61, 72
dependency
  home, 13, 109
  residual, 13, 16, 50, 55, 58, 114, 118, 120
Dhrystone benchmark, 86, 109
distributed scheduling, 1, 3, 4, 6, 7, 9, 20, 21, 42, 89, 117, 118, 120
  on Mach, 89–105
overview and examples, 23–29
distributed shared memory, 6, 7, 37, 39, 41, 47, 52, 75, 77, 87, 90, 104, 115, 117, 118
  server, 37
DSM, see distributed shared memory
emulator, 36, 40, 41, 53, 59, 69, 70, 73, 108, 109
eviction, 12, 15, 18
initial placement, 3, 4, 9, 15, 21, 26–28, 90, 96, 102–104, 107, 111, 116
instability, 21, 25, 27, 79, 83
insularity, 14, 47, 73
interposing kernel ports, 49, 51–57, 61, 73, 86, 120
IPC profiler, 7, 40, 77, 85, 86, 108
load balancing, 1, 5, 9, 16, 19, 21, 22, 26–28, 70
load distribution, 1, 3–6, 16, 21, 23, 26, 27, 29, 72, 90, 103, 104, 107, 108, 117–120
Mach support, 39–41
on top of Mach, overview, 42
load information management, 3, 5–7
for Mach, 75–88
information collection, 20, 21, 78–81
information dissemination, 21, 81–83
overview and examples, 19–23
load sharing, 1, 5, 9, 18, 19, 27, 28, 103
Locus, 15, 19, 52, 70
Lynx, 16
Mach
load distribution support, see load distribution, Mach support
NORMA version, 37–39
overview, architecture and history, 31–37
message based
interface, 34, 71, 108, 113, 114
systems, 75, 108, 112
message forwarding, 50, 51
μkernels, 2–5, 14, 16, 18, 29, 31, 34–36, 40, 41, 45–48, 70–72, 75, 87–90, 107, 111, 117–120
requirements for LD, 78
requirements for task migration, 48
suitability for LD, 112
UNIX emulation, 2, 4, 36, 40, 41, 46, 47, 49, 50, 65, 69, 71, 72, 110, 113
migration
candidate, 22, 49, 59, 65, 84, 86, 93, 100
characteristics
extensibility, 5, 14, 34, 72, 111, 112, 116, 117
flexibility, 5, 117
functionality, 5, 12, 13, 15, 45, 111, 116–118
maintainability, 5, 14, 72
performance, 2, 40, 45, 62–70, 111, 117
scalability, 111–113, 116
transparency, 2, 12–18, 31, 40, 41, 49, 54, 71–73, 111, 112, 116, 117
course grained, 10
fine grained, 10
heterogeneous, 14
of processes, see process migration
of tasks, see task migration
overview and examples, 9–19
Mirage, 15
modularity, 33, 34
monolithic kernels, 2, 14–16, 29, 34–36, 46, 47, 49, 70, 71, 111–113, 116
MS DOS, 2, 37
Netmsg, 38
Netmsg server, 37
accounting on, see accounting, on network IPC
experiences with, 113–114
network paging, see distributed shared memory
OMS, 56, 57, 59–63, 66
migration performance, 67
Optimized Migration Server, see OMS
INDEX

OS personality, 6, 7, 34, 35, 37, 45, 47–49, 69, 71, 73, 89, 90, 110, 116
OS/2, 2, 37
OSF/1, 15, 32, 35, 69, 71, 73
OSF/1 AD, 70, 108, 110

pagers, 33, 35, 37, 38, 51, 58, 59, 81, 115
Paragon, 70, 114
parallel make, 70
Peregrine, 77

process
detaching, 11, 14, 18
process migration, 3, 4, 6, 7, 11, 14–19, 22, 26, 28, 29, 37, 46, 47, 67, 70, 71, 90, 94, 103, 107–111, 113, 116, 118, 119

QNX, 2
RIG, 32, 37
rxe, 26

scalability, 23, 24, 27, 34, 84, 85, 112, 113, 120
scheduling heuristics, 25
Schizo, 70, 71, 73
scomp, 68
Simpack, 86
Simple Migration Server, see SMS
SMS, 56, 57, 59–64, 66
migration performance, 67
Sprite, 13–17, 19, 47, 52, 59, 67, 70, 71, 108–110, 115
migration performance, 67
state
extracting, 18, 41, 47, 50–54, 57, 58, 80, 85, 108
freezing, 11, 18, 49, 54, 55
Stealth, 70, 114
stretch factor, 21

task migration, 3–7, 10, 11, 27, 29, 36, 38, 39, 41, 45–73, 75, 89, 90, 94–96, 103, 104, 107–109, 113, 114, 117–120
v. initial placement, 111
v. process migration, 109–111
testbed, 46, 50, 94, 95
thrashing, 68

UNIX, 4, 17, 18, 26, 32, 36, 40, 45, 46, 70
AIX, 17, 72
AOS, 17, 72
AT&T, 2, 22, 26, 37
BSD, 2, 17, 26, 27, 32, 37, 72
emulation on μkernels, see μkernels,
UNIX emulation
SunOS, 17, 72
UNIX-like OS, 12, 14, 15, 29, 35, 71, 108, 110, 112, 113
Version 7, 16
Utopia, 27

V kernel, 13
migration performance, 67
VMS, 2, 4, 37

WPI benchmark suite, 63

XMM, 38, 39, 42, 55, 59, 63, 73, 76, 79, 80, 83, 84, 91, 93, 94, 96, 99, 100, 108, 110, 115, 120, see distributed shared memory
Automatic Parallelization
New Approaches to Code Generation, Data Distribution, and Performance Prediction

Edited by Christoph W. Keßler

1994. xii, 221 pages. (Vieweg Advanced Studies in Computer Science) Softcover
ISBN 3-528-05401-8


Contains 11 contributions which deal true automatic parallelization and the focus on automatic methods. The questions under discussion are: Up to which degree is automatic parallelization for DMS possible today? What are currently the most important problems for automatic parallelization and which new ideas are there? In which cases can knowledge-based methods help? Are there promising methods for automatic data distribution and redistribution? Why is performance prediction problematic?
Modeling of Dynamic Object Systems
A Logic-based Approach
by Ralf Jungclaus
With a foreword by H.-D. Ehrich
1993. xvi, 231 pages. (Vieweg Advanced Studies in Computer Science) Softcover
ISBN 3-528-05386-0


Presents an approach to the formal object-oriented specification of information systems. The approach focuses on the early phases of system development where existing systems have to be described or systems to be developed have to be prescribed (requirements specification or conceptional modeling). Systems are considered to be reactive systems composed from objects that evolve concurrently in a discrete, event-driven way. Based on a temporal logic framework the language TROLL is defined. TROLL supports the integrated specification of structural and behavioral properties of objects as well as abstractions like roles, specifications, and composite objects. Furthermore, concepts for constructing systems specifications from parts are introduced. Finally, the approach is related to similar ones and discussed.