

Contents

Preface	xvii
1 Introduction	1
1.1 Motivation	1
1.2 Load Distribution	3
1.3 Research Contributions	5
1.4 Thesis Outline	6
2 Background and Related Work	9
2.1 Introduction	9
2.2 Migration	9
2.2.1 Design	11
2.2.2 Issues	12
2.2.3 Previous Work	14
2.3 Load Information Management	19
2.3.1 Design	20
2.3.2 Issues	20
2.3.3 Previous Work	21
2.4 Distributed Scheduling	23
2.4.1 Design	23
2.4.2 Issues	24
2.4.3 Previous Work	25
2.5 Summary	29
3 Mach and Load Distribution	31
3.1 Introduction	31
3.2 Mach	31
3.3 Mach NORMA Version	37
3.4 Mach Support for Load Distribution	39
3.5 Load Distribution Architecture, Overview	42
3.6 Summary	43

4	Task Migration	45
4.1	Introduction	45
4.2	General Principles and Architecture	45
4.3	Requirements for Microkernels	48
4.4	Implementation	50
4.4.1	Task Migration Algorithm	50
4.4.2	Necessary Modifications to the Mach Microkernel	55
4.4.3	Simple Migration Server	56
4.4.4	Optimized Migration Server	57
4.4.5	In-Kernel Task Migration	60
4.5	Performance Measurements	62
4.5.1	Migration Server Measurements	64
4.5.2	WPI Benchmarks	67
4.5.3	Parallel Make and Other Applications	70
4.6	Related Work	70
4.7	Summary	72
5	Load Information Management	75
5.1	Introduction	75
5.2	Load Information Collection	78
5.2.1	Information on Processing	79
5.2.2	Information on Network IPC	79
5.2.3	Information on XMM	80
5.3	Information Dissemination and Negotiation	81
5.4	Performance Measurements	83
5.5	Characterization of Mach Applications	85
5.6	Summary	87
6	Distributed Scheduling	89
6.1	Introduction	89
6.2	Distributed Scheduling Algorithms	90
6.3	Artificial Load	94
6.4	Performance Measurements	95
6.4.1	Comparison of Various Strategies	96
6.4.2	Considering Information on Communication	99
6.4.3	Task Migration v. Initial Placement	102
6.5	Summary	104
7	Lessons Learned	107
7.1	Introduction	107
7.2	Task Migration Implementation	107
7.3	Task Migration is not Always Enough	109
7.4	Task Migration v. Initial Placement	111
7.5	Microkernels are the Right Level for LD	111

7.6	Experiences with Network IPC	113
7.7	Fault Tolerance is Hard to Support	114
7.8	Summary	116
8	Conclusions and Future Work	117
8.1	Introduction	117
8.2	Summary of Results	117
8.3	Future Work	118
8.4	Conclusion	120
	Bibliography	121
	Index	146

List of Figures

1.1	Monolithic v. Microkernel Architecture	3
2.1	Load Distribution Elements	10
2.2	Load Information Management	19
2.3	Distributed Scheduling	24
3.1	The History of Mach Development	35
3.2	Supporting the OS Personality Calls on Mach	36
3.3	Extending IPC to Network Environment Using Netmsg Server	38
3.4	NORMA Support for Distributed Shared Memory	39
3.5	Monolithic v. Microkernel Architectures	40
3.6	LD Architecture	42
4.1	Task Migration Design	46
4.2	Interposing Task Kernel Port, External Perspective	52
4.3	The Transferred Task State	53
4.4	Interposing the Task Kernel Port - Internal Perspective	55
4.5	Lazy Copying Address Space with OMS	58
4.6	Task State Transfer for Various Task Migration Versions	62
4.7	Transfer Time v. Memory Size	63
4.8	Transfer Time v. Number of Threads	63
4.9	Transfer Time v. Number of Receive Capabilities	64
4.10	Transfer Time v. Number of Capabilities	64
4.11	Transfer Time v. Memory Contiguity	65
4.12	Execution Time v. Memory Size	65
4.13	Transfer Time v. Number of Receive Capabilities for SMS and OMS	66
4.14	Execution Time v. Remote/Local IPC	66
4.15	Solving Time v. Jigsaw Puzzle Size	68
4.16	Paging v. Jigsaw Puzzle Size	68
4.17	Make Ratio	69
5.1	Common Types of NORMA Applications	76
5.2	IPC-Type Task Execution Time v. Local/Remote Execution.	78
5.3	XMM-Type Task Execution Time v. Local/Remote Execution.	78
5.4	Network and Local IPC Paths and Accounting	81

5.5	Network and Local Paging Paths and Accounting	82
5.6	Average Execution Time as a Function of the IPC and Processing. . .	83
5.7	Average Execution Time as a Function of the XMM and Processing. .	83
5.8	Profiling Application's IPC Activity	85
6.1	Distributed Scheduling Algorithm	92
6.2	Artificial Load for Distributed Scheduling Experiments	95
6.3	Average Execution Time v. Homogeneous Load	96
6.4	Average Execution Time v. Unequal System Load	97
6.5	Average Execution Time for Symmetrical Strategy v. System Load. .	98
6.6	Average Execution Time for Sender-Initiated Strategy v. System Load.	98
6.7	Average Execution Time for Random Strategy v. System Load. . . .	99
6.8	Average Execution Time for Receiver-Initiated Strategy v. System Load.	99
6.9	Average Execution Time for Sender-Initiated Strategy v. XMM Load. .	100
6.10	Average Execution Time for Receiver-Initiated Strategy v. XMM Load.	100
6.11	MPP Architecture	102
6.12	Average Execution Time as a Function of the Amount of IPC for Initial Placement and Task Migration.	103
7.1	Average Execution Time for ALT with Dhrystone.	109
7.2	Average Execution Time for ALT with Simple Processing Code. . . .	109
7.3	Residual Dependency of Migrated Task	115
8.1	Load Distribution and PVM	119

List of Tables

5.1	Applications as a Function of Processing, IPC, and VM	87
6.1	Comparison of Various Strategies	101
7.1	Comparison of the LD Mechanisms on Various Architectures.	113