

# Contents

<b>List of Figures</b>	<b>10</b>
<b>1 Introduction</b>	<b>2</b>
<b>2 Active Objects</b>	<b>6</b>
2.1 Active Objects . . . . .	7
2.2 Reflection . . . . .	8
2.2.1 Reflective Architecture . . . . .	9
2.3 ProActive . . . . .	10
2.3.1 Distribution model . . . . .	11
2.3.2 Active Objects implementation for ProActive . . . . .	12
2.3.3 Message Passing for Actives Objects in ProActive . . . . .	13

2.3.4	Synchronisation: Wait-by-necessity . . . . .	15
2.3.5	ProActive: Environment and implementation . . . . .	16
2.3.6	ProActive Meta-Object Protocol . . . . .	19
<b>3</b>	<b>Networks for parallelism</b>	<b>24</b>
3.1	History of parallel computing . . . . .	25
3.1.1	Cluster of computers . . . . .	25
3.1.2	Computer Grids . . . . .	27
3.1.3	A model overview for Project Grids . . . . .	28
3.2	Peer-to-Peer Infrastructure of ProActive . . . . .	29
3.2.1	Bootstrapping: First Contact . . . . .	31
3.2.2	Discovering and Self-Organising . . . . .	31
3.3	Theory of Networks . . . . .	33
3.3.1	Generating random graphs . . . . .	34
3.3.2	Natural Networks . . . . .	35
<b>4</b>	<b>State of the Art on Load-Balancing</b>	<b>39</b>
4.1	Static Load-Balancing . . . . .	40

4.2	Dynamic Load-Balancing . . . . .	41
4.3	Components of a Load-Balancing Algorithm . . . . .	44
4.3.1	Load Index . . . . .	45
4.3.2	Information-Sharing Policy . . . . .	46
4.3.3	Transfer Policy . . . . .	47
4.3.4	Location Policy . . . . .	49
4.4	Related Work . . . . .	49
4.4.1	Condor . . . . .	49
4.4.2	Legion . . . . .	52
4.4.3	Cilk . . . . .	55
4.4.4	Satin . . . . .	58
<b>5</b>	<b>Setting foundations for Load-Balancing of Active-Objects</b>	<b>61</b>
5.1	Active-Objects and Processing Idleness . . . . .	62
5.2	Location policy for load-balancing of active-objects . . . . .	64
5.3	Information and transfer policies for load-balancing of active-objects . . . . .	65
5.3.1	Modelling ProActive behaviour to test algorithm policies . . . . .	65

5.3.2	Implementing the Information-Sharing Policies . . . . .	66
5.3.3	Hardware and Software . . . . .	69
5.3.4	Results Analysis . . . . .	70
5.3.5	Testing the impact of Information-Sharing Policies . . . . .	74
5.4	Exploiting the Peer-to-Peer infrastructure: Information on-demand . . . . .	75
5.4.1	Robin-Hood Load-Balancing Algorithm . . . . .	76
5.4.2	Robin-Hood over ProActive's Peer-to-Peer Infrastructure . . . . .	77
5.5	Robin-Hood and the Nottingham Sheriff . . . . .	79
5.6	Testing algorithms in a real environment . . . . .	80
<b>6</b>	<b>Models, Simulations and Deployment on Large-Scale Networks</b>	<b>83</b>
6.1	Simulating Desktop Grids . . . . .	84
6.1.1	Characterising nodes of Desktop Grids . . . . .	84
6.1.2	Modelling Desktop Grids . . . . .	85
6.1.3	Finding the best processor . . . . .	87
6.1.4	Scaling towards the "infinite network" . . . . .	95
6.2	Simulating Project Grids . . . . .	103

6.2.1	Characterising a Project Grid . . . . .	106
6.2.2	Modelling a Project Grid . . . . .	108
6.2.3	Environment-aware Algorithms . . . . .	110
6.2.4	Experimental Setup . . . . .	111
6.2.5	Simulation Results . . . . .	112
6.2.6	Results Confidence . . . . .	114
6.3	Where to run parallel applications? . . . . .	118
6.3.1	Problematic of Applications and Descriptors . . . . .	118
6.3.2	Clauses in ProActive Descriptors . . . . .	119
6.3.3	Clauses in ProActive Applications . . . . .	121
6.3.4	Constraints . . . . .	121
6.4	The real world . . . . .	124
<b>7</b>	<b>Conclusions and Future Work</b>	<b>129</b>
<b>A</b>	<b>Matrices for Robin-Hood algorithm working alone</b>	<b>133</b>
<b>B</b>	<b>Matrices for Robin-Hood + Nottingham-Sheriff algorithm</b>	<b>138</b>

<b>C Expected values for Kolmogorov-Smirnov test statistics</b>	<b>143</b>
<b>Bibliography</b>	<b>146</b>

# List of Figures

- 2.1 The reflection process, featuring levels of data, reification and reflection. . . . . 9
- 2.2 Parallelisation and distribution with active objects . . . . . 11
- 2.3 Execution of an asynchronous and remote method call . . . . . 14
- 2.4 Base-level and meta-level of an active object . . . . . 19
- 2.5 Migration and tensioning . . . . . 22
  
- 3.1 Grids divided by objective . . . . . 29
- 3.2 (a) step two of Watts and Strogatz model with  $n = 12$  and  $k = 2$ ; (b) step three with small  $p_e$  . . . . . 36
  
- 4.1 A supermarket . . . . . 40
- 4.2 Examples of information-sharing policies . . . . . 47
- 4.3 Matchmaking process of Condor . . . . . 50

4.4	Parallel problems solved by <i>Condor</i> . . . . .	51
4.5	Main classes of Legion infrastructure . . . . .	53
4.6	Legion Resource Management Infrastructure . . . . .	55
4.7	<i>Cilk</i> model: each thread is a circle, grouped in procedures. Each downward arrow is a spawned child, and each horizontal arrow is a spawned successor. Dashed arrows represent data dependency (synchronisations). Also, spawn-levels from the original thread are presented. . . . .	56
5.1	Different behaviours for active-objects request (Q) and reply (P): (a) B starts in wait-for-request (WfR) and A made a wait-by-necessity (WfN). (b) Bad utilisation of the active-object pattern: asynchronous calls become almost synchronous. (c) C has a long waiting time because B delayed the answer. . . . .	62
5.2	The supermarket abstraction for load-balancing of enqueued tasks. . . . .	63
5.3	The supermarket abstraction for load-balancing of Active Objects. . . . .	63
5.4	Migration time from the point of view of latency and object' size . . . . .	64
5.5	Mean response time for all policies . . . . .	71
5.6	Bandwidth usage of coordination policies during the information-sharing phase . . . . .	72
5.7	Bandwidth usage of coordination policies during all the load-balancing . . . . .	73
5.8	Impact of load-balancing algorithms over Jacobi calculus . . . . .	82
6.1	Frequency distribution of Mflops for 200,000 processors registered at Seti@home and the normal function which models it. . . . .	86



6.2	Final distribution for the <i>Robin-Hood</i> algorithm only, for $RB = 0.5$ and $T = 0.5$ . . .	89
6.3	Final distribution for the <i>Robin-Hood + Nottingham Sheriff</i> . . . . .	91
6.4	Tuning for RS considering: a) number of active-objects in (9,9) per total of active-objects; and b) Number of total migrations reaching a stable state. . . . .	92
6.5	Tuning for RS considering: a) number of active-objects in (9,9) per total of active-objects; and b) Number of total migrations reaching a stable state. Because the results using 3 to 6 acquaintances were similar, only those for 3 are shown. . . . .	94
6.6	Tuning for RS considering: a) mean number of total migrations until each time-step; and b) mean number of overloaded nodes in each time-step. Using $RB = 0.7$ , acquaintances subset size = 3, $ x - y  \leq 3$ , $\lambda = 0.1, 0.2, 0.3$ and $T = 0.7$ . . . . .	96
6.7	Tuning the value of RS considering: a) mean number of active objects on a node with $\mu \geq 1$ per total number of active objects; and b) mean number of active objects on a node with $\mu > 1 + \frac{1}{3}$ per total number of active objects. Using $RB = 0.7$ , acquaintances subset size = 3, $ x - y  \leq 3$ , $\lambda = 0.1, 0.2, 0.3$ and $T = 0.7$ . . . . .	98
6.8	Scalability for a network using $RS = 0.9, 1.0, 1.1$ , $RB = 0.7$ . . . . .	100
6.9	Scalability in terms of number of processors used, having $RS = 1.0$ . . . . .	102
6.10	Scalability in terms of number of migrations, having $RS = 1.0$ . The plot presents, for an active object, the (mean) number of accumulated migrations performed until a time-step $t \in [0; 1,000]$ . . . . .	103
6.11	Scalability, having the number of active objects proportional to the number of nodes	104
6.12	Latency between nodes from the PlugTest project grid. . . . .	108
6.13	Total number of pending requests in all active-objects using message-size $C = 0.1$ and object size $M = 1$ , without synchronisation. . . . .	114

6.14	Total number of pending requests in all active-objects using message-size $C = 1$ and object size $M = 10$ , without synchronisation. . . . .	115
6.15	Total number of pending requests in all active-objects using message-size $C = 0.1$ services, object size $M = 1$ services and synchronisation each 10 time-steps. . . . .	116
6.16	% of confidence of load-balancing algorithms, increasing object size ( $M$ ) . . . . .	117
6.17	Example of clauses in descriptor. . . . .	120
6.18	Example of clauses in application. . . . .	122
6.19	Integer Constraint Schema Grammar. . . . .	123
6.20	Institutional clusters on Grid5000: Bordeaux, Grenoble, Lille, Lyon, Nancy, Orsay, Rennes, Sophia-Antipolis and Toulouse. . . . .	124
6.21	Speed of Jacobi parallel application in iterations per milliseconds. . . . .	126
6.22	Mean number of cumulated migrations that an active object performs during the experience. . . . .	127
A.1	Final distribution for the <i>Robin-Hood</i> algorithm only, for $RB = 0.5$ and $q = 3$ . . . . .	134
A.2	Final distribution for the <i>Robin-Hood</i> algorithm only, for $RB = 0.5$ and $q = 4$ . . . . .	135
A.3	Final distribution for the <i>Robin-Hood</i> algorithm only, for $RB = 0.5$ and $q = 5$ . . . . .	136
A.4	Final distribution for the <i>Robin-Hood</i> algorithm only, for $RB = 0.7$ and $q = 4$ . . . . .	137
B.1	Final distribution for the <i>Robin-Hood</i> + <i>Nottingham Sheriff</i> algorithm, for $RB = 0.5$ , $RS = 0.5$ and $q = 3$ . . . . .	139

B.2	Final distribution for the <i>Robin-Hood + Nottingham Sheriff</i> algorithm, for $RB = 0.5$ , $RS = 0.5$ and $q = 5$ . . . . .	140
B.3	Final distribution for the <i>Robin-Hood + Nottingham Sheriff</i> algorithm, for $RB = 0.7$ , $RS = 0.7$ and $q = 3$ . . . . .	141
B.4	Final distribution for the <i>Robin-Hood + Nottingham Sheriff</i> algorithm, for $RB = 0.9$ , $RS = 0.9$ and $q = 3$ . . . . .	142