Table of Contents

Biographical Sketch iii
Acknowledgements v
Table of Contents vii
List of Tables xi
List of Figures xii

1 Introduction 1

1.1 Thesis Contributions 4

1.1.1 Jbufs: Safe and Explicit Management of Buffers 4
1.1.2 Jstreams: Optimizing Serialization for Cluster Applications 5
1.1.3 Overview of Related Approaches 6

1.2 Thesis Overview 7

2 Interfacing Java with Network Interfaces 9

2.1 Background 10

2.1.1 The Virtual Interface Architecture 10
2.1.2 Giganet cLAN™ GNN-1000 Cluster 13
2.1.3 Explicit Buffer Mapping: A Case for Buffer Re-Use 13
2.1.4 Java: A Safe Language 16
2.1.5 Separation between Garbage-Collected and Native Heaps 19

2.2 Java-Native Interfaces 21
List of Tables

2.1 Marmot, J/Direct, and JNI’s GC-related features. 24
2.2 Cost of Java-to-C downcalls. 25
2.3 Cost of C-to-Java upcalls. 25
2.4 Cost of accessing Java fields from C. 25
2.5 Cost of crossing the GC/Native separation. 26
2.6 Javia-I 4-byte round-trip latencies and per-byte overhead. 32
3.1 Jbufs overheads in Marmot. 45
3.2 Javia-II 4-byte round-trip latencies and per-byte overhead. 51
4.1 Impact of Marmot optimizations in serialization. 76
4.2 Impact of Marmot optimizations in de-serialization. 76
4.3 RMI 4-byte round-trip latencies. 81
4.4 Summary of RMI benchmark suite. 82
4.5 Communication profile of structured RMI applications. 88
4.6 Estimated impact of serialization on application performance. 88
5.1 Measured impact of jstreams on application performance. 109
List of Figures

2.1 Virtual Interface data structures. ................................. 11
2.2 Typical in-memory representation of a Buffer object. .... 19
2.3 The hard separation between garbage-collected and native heaps. 20
2.4 JaviaI per-endpoint data structures. ......................... 29
2.5 Javia-I round-trip latencies. .................................... 33
2.6 Javia-I effective bandwidth. .................................... 34
3.1 Typical lifetime of a jbuf with a copying garbage collector. 39
3.2 Jbufs state diagram for runtime safety checks. ........... 42
3.3 Javia-II per-endpoint data structures. ....................... 49
3.4 Javia-II round-trip latencies ................................. 51
3.5 Javia-II effective bandwidth ................................ 52
3.6 Performance of MM on a single 450Mhz Pentium-II. .... 55
3.7 Impact of safety checks on MM. ................................. 55
3.8 Communication time in pMM (64x64 matrices, 8 processors). 58
3.9 Communication time in pMM (256x256 matrices, 8 processors). 58
3.10 Overall performance of pMM (64x64 matrices, 8 processors). 59
3.11 Overall performance of pMM (256x256 matrices, 8 processors) 59
3.12 Jam round-trip latencies. ....................................... 64
3.13 Jam effective bandwidth. ....................................... 65
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Object Serialization and De-serialization.</td>
<td>72</td>
</tr>
<tr>
<td>4.2 Serialization costs in three implementations of JOS.</td>
<td>74</td>
</tr>
<tr>
<td>4.3 De-serialization costs in three implementations of JOS.</td>
<td>75</td>
</tr>
<tr>
<td>4.4 RMI round-trip latencies.</td>
<td>80</td>
</tr>
<tr>
<td>4.5 RMI effective bandwidth.</td>
<td>80</td>
</tr>
<tr>
<td>4.6 Speedups of TSP and IDA.</td>
<td>86</td>
</tr>
<tr>
<td>4.7 Speedup of SOR.</td>
<td>86</td>
</tr>
<tr>
<td>4.8 Performance of EM3D on 8 processors.</td>
<td>87</td>
</tr>
<tr>
<td>4.9 Performance of FFT on 8 processors.</td>
<td>87</td>
</tr>
<tr>
<td>4.10 Performance of pMM on 8 processors.</td>
<td>88</td>
</tr>
<tr>
<td>5.1 Serialization with jstreams.</td>
<td>96</td>
</tr>
<tr>
<td>5.2 De-serialization with jstreams.</td>
<td>97</td>
</tr>
<tr>
<td>5.3 Jstreams state diagram for runtime safety checks.</td>
<td>98</td>
</tr>
<tr>
<td>5.4 Jstreams wire protocol in Marmot.</td>
<td>100</td>
</tr>
<tr>
<td>5.5 Serialization overheads of jstreams in Marmot.</td>
<td>103</td>
</tr>
<tr>
<td>5.6 De-serialization overheads of jstreams in Marmot.</td>
<td>103</td>
</tr>
<tr>
<td>5.7 RMI round-trip latencies with jstreams.</td>
<td>107</td>
</tr>
<tr>
<td>5.8 RMI effective bandwidth with jstreams.</td>
<td>108</td>
</tr>
</tbody>
</table>