SEVENTH FRAMEWORK PROGRAMME
Research Infrastructures

INFRA-2007-2.2.2.1 - Preparatory phase for 'Computer and Data Treatment' research infrastructures in the 2006 ESFRI Roadmap

PRACE

Partnership for Advanced Computing in Europe

Grant Agreement Number: RI-211528

D6.2.2
Final Report on Application Requirements

Final

Version:	1.0
Author(s):	Mark Bull and Alan Simpson, EPCC Harald Klimach, HLRS
Date:	26.09.08

Project and Deliverable Information Sheet

PRACE Project	**Project Ref. №: RI-211528**	
	Project Title: Partnership for Advanced Computing in Europe	
	Project Web Site: http://www.prace-project.eu/	
	Deliverable ID: D6.2.2	
	Deliverable Nature: Report	
	Deliverable Level: PU	**Contractual Date of Delivery:** 30 / Sept / 2008
		Actual Date of Delivery: 30 / Sept/ 2008
	EC Project Officer: Maria Ramalho-Natario	

* - The dissemination level are indicated as follows: **PU** – Public, **PP** – Restricted to other participants (including the Commission Services), **RE** – Restricted to a group specified by the consortium (including the Commission Services). **CO** – Confidential, only for members of the consortium (including the Commission Services).

Document Control Sheet

Document	**Title: Final Report on Application Requirements**	
	ID: D6.2.2	
	Version: 1.0	**Status:** Final
	Available at: http://www.prace-project.eu/	
	Software Tool: Microsoft Word 2003	
	File(s): PraceDeliverableTemplate.doc	
Authorship	**Written by:**	Mark Bull and Alan Simpson, EPCC Harald Klimach, HLRS
	Contributors:	
	Reviewed by:	Rolf Rabenseifner, HLRS Thomas Eickermann, FZJ
	Approved by:	Technical Board

Document Status Sheet

Version	**Date**	**Status**	**Comments**
0.1	26/Aug/2008	Draft	Initial Draft
0.2	9/Sep/2008		Chapter 2 drafted
0.3	10/Sep/2008		Included Chapter 3
0.4	12/Sep/2008	Final draft	Draft for internal review
0.5	23/Sep/2008		Response to reviewers comments
1.0	26/Sep/2008	Final	For EC

Document Keywords and Abstract

Keywords:	PRACE, HPC, Research Infrastructure, Applications, Benchmark Suite, User Survey
Abstract:	This is a detailed report on application requirements for European petaflop/s systems. It updates the previous report (D6.2.1) in two ways: 1. The applications analysed are based on those identified as representative in D6.1. 2. The analysis of applications is based on actual performance data, rather than a questionnaire. 3. This data is complemented by the results of a survey of many of the major users in Europe. The applications codes analysed are those selected by PRACE as representative of the likely load on European petaflop/s systems and which are candidates for inclusion in a benchmark suite to help select such systems. These applications span a broad range of scientific areas and are used in many European countries. Each of the applications was run on one of the classes of architectures identified by WP7 as suitable prototypes for future petascale systems. We selected appropriate data sets for each application and then collected profiling data on production-scale runs so that we could identify the requirements of each application. A user survey was also sent to the Top 10 users in each PRACE country, as identified by WP3. This survey included questions about the user, usage patterns, HPC infrastructure, upcoming algorithms and general comments about future petascale systems. Almost 70 responses were received and analysed for this report.

Table of Contents

List of Figures viii
List of Tables ix
References and Applicable Documents xii
List of Acronyms and Abbreviations xiii
Executive Summary 1
1 Introduction 2
1.1 Structure of the Report 3
2 Analysis of Applications 4
2.1 Methodology 4
2.1.1 *Execution times* 4
2.1.2 *Memory usage* 4
2.1.3 *Profiling* 4
2.1.4 *I/O* 4
2.1.5 *Communication* 5
2.1.6 *CPU and cache* 5
2.1.7 *Derived metrics* 5
2.2 NAMD 6
2.2.1 *Summary* 6
2.2.2 *Benchmark dataset* 6
2.2.3 *Hardware platforms* 6
2.2.4 *Execution times* 6
2.2.5 *Memory Usage* 7
2.2.6 *Profiling* 7
2.2.7 *Communication* 8
2.2.8 *I/O* 9
2.2.9 *CPU and cache Cray XT4* 9
2.2.10 *Derived metrics* 9
2.2.11 *Analysis* 10
2.3 CPMD 11
2.3.1 *Summary* 11
2.3.2 *Benchmark dataset* 11
2.3.3 *Hardware platforms* 11
2.3.4 *Execution times* 11
2.3.5 *Memory Usage* 11
2.3.6 *Profiling* 11
2.3.7 *Communication* 12
2.3.8 *I/O* 12
2.3.9 *CPU and cache* 12
2.3.10 *Derived metrics* 13
2.3.11 *Analysis* 13
2.4 VASP 14
2.4.1 *Summary* 14
2.4.2 *Benchmark dataset* 14
2.4.3 *Hardware platforms* 14
2.4.4 *Execution times* 14
2.4.5 *Memory Usage* 14

2.4.6 *Profiling* ... 14
2.4.7 *Communication* ... 14
2.4.8 *I/O* ... 15
2.4.9 *CPU and cache* ... 15
2.4.10 *Derived metrics* ... 16
2.4.11 *Analysis* ... 16
2.5 GADGET ... 16
2.5.1 *Summary* ... 16
2.5.2 *Benchmark dataset* ... 17
2.5.3 *Hardware platforms* ... 17
2.5.4 *Execution times* ... 17
2.5.5 *Memory Usage* ... 17
2.5.6 *Profiling* ... 17
2.5.7 *Communication* ... 17
2.5.8 *I/O* ... 18
2.5.9 *CPU and cache* ... 18
2.5.10 *Derived metrics* ... 19
2.5.11 *Analysis* ... 19
2.6 CODE_SATURNE ... 20
2.6.1 *Summary* ... 20
2.6.2 *Benchmark dataset* ... 20
2.6.3 *Hardware platforms* ... 20
2.6.4 *Execution times* ... 20
2.6.5 *Memory Usage* ... 20
2.6.6 *Profiling* ... 20
2.6.7 *Communication* ... 21
2.6.8 *I/O* ... 21
2.6.9 *CPU and cache* ... 21
2.6.10 *Derived metrics* ... 22
2.6.11 *Analysis* ... 22
2.7 TORB ... 23
2.7.1 *Summary* ... 23
2.7.2 *Benchmark dataset* ... 23
2.7.3 *Hardware platforms* ... 23
2.7.4 *Execution times* ... 23
2.7.5 *Memory Usage* ... 23
2.7.6 *Profiling* ... 23
2.7.7 *Communication* ... 23
2.7.8 *I/O* ... 24
2.7.9 *CPU and cache* ... 24
2.7.10 *Derived metrics* ... 25
2.7.11 *Analysis* ... 25
2.8 NEMO ... 25
2.8.1 *Summary* ... 25
2.8.2 *Benchmark dataset* ... 26
2.8.3 *Hardware platforms* ... 26
2.8.4 *Execution times* ... 26
2.8.5 *Memory Usage* ... 26
2.8.6 *Profiling* ... 26
2.8.7 *Communication* ... 27
2.8.8 *I/O* ... 28
2.8.9 *CPU and cache* ... 28

2.8.10 *Derived metrics* ... *29*
2.8.11 *Analysis* ... *29*
2.9 ECHAM5 ... 29
2.9.1 *Summary* ... *29*
2.9.2 *Benchmark dataset* ... *30*
2.9.3 *Hardware platforms* ... *30*
2.9.4 *Execution times* ... *30*
2.9.5 *Memory Usage* ... *30*
2.9.6 *Profiling* ... *30*
2.9.7 *Communication* ... *31*
2.9.8 *I/O* ... *32*
2.9.9 *CPU and cache* ... *32*
2.9.10 *Derived metrics* ... *32*
2.9.11 *Analysis* ... *33*
2.10 CP2K ... 33
2.10.1 *Summary* ... *33*
2.10.2 *Benchmark dataset* ... *34*
2.10.3 *Hardware platforms* ... *34*
2.10.4 *Execution times* ... *34*
2.10.5 *Memory Usage* ... *34*
2.10.6 *Profiling* ... *34*
2.10.7 *Communication* ... *35*
2.10.8 *I/O* ... *35*
2.10.9 *CPU and cache* ... *35*
2.10.10 *Derived metrics* ... *36*
2.10.11 *Analysis* ... *36*
2.11 GROMACS ... 37
2.11.1 *Summary* ... *37*
2.11.2 *Benchmark dataset* ... *37*
2.11.3 *Hardware platforms* ... *37*
2.11.4 *Execution times* ... *37*
2.11.5 *Memory Usage* ... *38*
2.11.6 *Profiling* ... *38*
2.11.7 *Communication* ... *39*
2.11.8 *I/O* ... *39*
2.11.9 *CPU and cache* ... *40*
2.11.10 *Derived metrics* ... *40*
2.11.11 *Analysis* ... *41*
2.12 N3D ... 41
2.12.1 *Summary* ... *41*
2.12.2 *Benchmark dataset* ... *41*
2.12.3 *Hardware platforms* ... *41*
2.12.4 *Execution times* ... *41*
2.12.5 *Memory Usage* ... *41*
2.12.6 *Profiling* ... *42*
2.12.7 *Communication* ... *42*
2.12.8 *I/O* ... *42*
2.12.9 *CPU and cache* ... *42*
2.12.10 *Derived metrics* ... *43*
2.12.11 *Analysis* ... *43*
2.13 HELIUM ... 44
2.13.1 *Summary* ... *44*

2.13.2 *Benchmark dataset* ... *44*
2.13.3 *Hardware platforms* ... *44*
2.13.4 *Execution times* ... *44*
2.13.5 *Memory Usage* ... *44*
2.13.6 *Profiling* ... *44*
2.13.7 *Communication* ... *45*
2.13.8 *I/O* ... *46*
2.13.9 *CPU and cache* ... *46*
2.13.10 *Derived metrics* ... *46*
2.13.11 *Analysis* ... *47*

2.14 GPAW ... 47

2.14.1 *Summary* ... *47*
2.14.2 *Benchmark dataset* ... *47*
2.14.3 *Hardware platforms* ... *47*
2.14.4 *Execution times* ... *47*
2.14.5 *Memory Usage* ... *47*
2.14.6 *Profiling* ... *48*
2.14.7 *Communication* ... *48*
2.14.8 *I/O* ... *48*
2.14.9 *CPU and cache* ... *48*
2.14.10 *Derived metrics* ... *49*
2.14.11 *Analysis* ... *49*

2.15 ALYA ... 50

2.15.1 *Summary* ... *50*
2.15.2 *Benchmark dataset* ... *50*
2.15.3 *Hardware platforms* ... *50*
2.15.4 *Execution times* ... *50*
2.15.5 *Memory Usage* ... *50*
2.15.6 *Profiling* ... *50*
2.15.7 *Communication* ... *50*
2.15.8 *I/O* ... *51*
2.15.9 *CPU and cache* ... *51*
2.15.10 *Derived metrics* ... *51*

2.16 PEPC ... 52

2.16.1 *Summary* ... *52*
2.16.2 *Benchmark dataset* ... *52*
2.16.3 *Hardware platforms* ... *52*
2.16.4 *Execution times* ... *53*
2.16.5 *Memory Usage* ... *53*
2.16.6 *Profiling* ... *53*
2.16.7 *Communication* ... *53*
2.16.8 *I/O* ... *54*
2.16.9 *CPU and cache* ... *54*
2.16.10 *Derived metrics* ... *54*
2.16.11 *Analysis* ... *55*

2.17 Summary of Application Requirements ... 55

3 Survey of Users ... 61

3.1 Information about the users ... 62
3.2 Usage patterns for HPC systems ... 65
3.3 Feedback on the desired infrastructure ... 67
3.4 Future algorithms ... 72

3.4.1 *Computational Chemistry (total 16 working groups)* ... *72*

3.4.2 *Computational Fluid Dynamics (total 12 working groups)* *72*
3.4.3 *Condensed Matter Physics (total 12 working groups)* *72*
3.4.4 *Astronomy and Cosmology (total 7 working groups)* *72*
3.4.5 *Life Sciences (total 6 working groups)* *72*
3.4.6 *Particle Physics (total 6 working groups)* *73*
3.4.7 *Earth and Climate Sciences (total 3 working groups)* *73*
3.4.8 *Other scientific fields* *73*
3.4.9 *Combined Overview* *73*
4 Summary and Future Work 74
Appendix A: User Survey 76

List of Figures

Figure 1. Application efficiency between the two core counts used 56
Figure 2. Percentage of peak flop rate attained 56
Figure 3. Percentage of instructions executed that are floating point 57
Figure 4. Ratio of floating point operations to loads and stores 57
Figure 5. Cycles per L2 cache reference 59
Figure 6. Cycles per memory reference 59
Figure 7. Cycles per byte communicated per core 60
Figure 8. Cycles per byte of total I/O 60
Figure 9. Share of responses by scientific fields. 63
Figure 10. Working group sizes and their correlation with open source licenses. 65
Figure 11. Typical job sizes and their execution lengths. 66
Figure 12. Shares of Capability and Capacity computing. 67
Figure 13. Required memory space by typical jobs. Six responses not captured in this figure, as they stated a size per process of 1 GB. 68
Figure 14. Disk Usage: Amount of needed disk space for typical jobs colour coded with the needed accessibility of the stored data. 69
Figure 15. Need for data transfer of produced data back to the home site of the working group, split into 5 categories. 70

List of Tables

Table 1. Execution times for NAMD on Cray XT4 6
Table 2. Execution times for NAMD on IBM p575 cluster 7
Table 3. Memory usage for NAMD on Cray XT4 7
Table 4. Memory usage for NAMD on IBM p575 cluster 7
Table 5. Profile of NAMD on 256 cores of Cray XT4 7
Table 6. Profile of NAMD on 512 cores of Cray XT4 7
Table 7. Profile of NAMD on 256 cores of IBM p575 cluster 8
Table 8. Profile of NAMD on 512 cores of IBM p575 cluster 8
Table 9. Data transfer for NAMD on Cray XT4 8
Table 10. Communication profile of NAMD on 256 cores of Cray XT4 8
Table 11. Communication profile of NAMD on 512 cores of Cray XT4 8
Table 12. Communication profile of NAMD on 256 cores of IBM p575 cluster 8
Table 13. Communication profile of NAMD on 512 cores of IBM p575 cluster 8
Table 14. I/O usage of NAMD on Cray XT4 9
Table 15. Hardware counter data for NAMD on Cray XT4 9
Table 16. Hardware counter data for NAMD on IBM p575 cluster 9
Table 17. Derived metrics for NAMD on Cray XT4 10
Table 18. Derived metrics for NAMD on IBM p575 cluster 10
Table 19. Execution times for CPMD on IBM JS21 cluster 11
Table 20. Data transfer for CPMD on IBM JS21 cluster 12
Table 21. Communication profile of CPMD on 128 cores of IBM JS21 cluster 12
Table 22. Communication profile of CPMD on 256 cores of IBM JS21 cluster 12
Table 23. I/O usage of CPMD on 128 cores of IBM JS21 cluster 12
Table 24. I/O usage of CPMD on 256 cores of IBM JS21 cluster 12
Table 25. Hardware counter data for CPMD on IBM p575 cluster 13
Table 26. Derived metrics for CPMD on IBM JS21 cluster 13
Table 27. Execution times for VASP on IBM JS21 cluster 14
Table 28. Data transfer for VASP on IBM JS21 cluster 14
Table 29. Communication profile of VASP on 128 cores of BM JS21 cluster 15
Table 30. Communication profile of VASP on 256 cores of IBM JS21 cluster 15
Table 31. I/O usage of VASP on 128 cores of IBM JS21 cluster 15
Table 32. I/O usage of VASP on 256 cores of IBM JS21 cluster 15
Table 33. Hardware counter data for VASP on IBM p575 cluster 15
Table 34. Derived metrics for VASP on IBM JS21 cluster 16
Table 35. Execution times for GADGET on SGI Altix 17
Table 36. Memory usage for GADGET on SGI Altix 17
Table 37. Data transfer for GADGET on SGI Altix 17
Table 38. Communication profile of GADGET on 256 cores of SGI Altix 18
Table 39. Communication profile of GADGET on 512 cores of SGI Altix 18
Table 40. I/O usage of GADGET on 256 cores of SGI Altix 18
Table 41. I/O usage of GADGET on 512 cores of SGI Altix 18
Table 42. Hardware counter data for GADGET on SGI Altix 19
Table 43. Derived metrics for GADGET on SGI Altix 19
Table 44. Execution times for CODE_SATURNE on IBM p575 cluster 20
Table 45. Memory usage for CODE_SATURNE on IBM p575 cluster 20
Table 46. Data transfer for CODE_SATURNE on IBM p575 cluster 21
Table 47. Communication profile of CODE_SATURNE on 256 cores of IBM p575 cluster .. 21
Table 48. Communication profile of CODE_SATURNE on 512 cores of IBM p575 cluster .. 21
Table 49. I/O usage of CODE_SATURNE on 256 cores of IBM p575 cluster 21
Table 50. I/O usage of CODE_SATURNE on 512 cores of IBM p575 cluster 21
Table 51. Hardware counter data for CODE_SATURNE on IBM p575 cluster 22
Table 52. Derived metrics for CODE_SATURNE on IBM p575 cluster 22
Table 53. Execution times for TORB on IBM JS21 cluster 23

Table 54. Data transfer for TORB on IBM JS21 cluster 23
Table 55. Communication profile of TORB on 128 cores of BM JS21 cluster 24
Table 56. Communication profile of TORB on 256 cores of IBM JS21 cluster 24
Table 57. I/O usage of TORB on 128 cores of IBM JS21 cluster 24
Table 58. I/O usage of TORB on 256 cores of IBM JS21 cluster 24
Table 59. Hardware counter data for TORB on IBM p575 cluster 25
Table 60. Derived metrics for TORB on IBM JS21 cluster 25
Table 61. Execution times for NEMO on IBM Power6 cluster 26
Table 62. Memory usage for NEMO on IBM Power6 cluster 26
Table 63. Profile of NEMO on 256 cores of IBM Power6 cluster 27
Table 64. Profile of NEMO on 512 cores of IBM Power6 cluster 27
Table 65. Data transfer for NEMO on IBM Power6 cluster 27
Table 66. Communication profile of NEMO on 256 cores of IBM Power6 cluster 28
Table 67. Communication profile of NEMO on 512 cores of IBM Power6 cluster 28
Table 68. I/O usage of NEMO on 256 cores of IBM Power6 cluster 28
Table 69. I/O usage of NEMO on 512 cores of IBM Power6 cluster 28
Table 70. Hardware counter data for NEMO on IBM Power6 cluster 28
Table 71. Derived metrics for NEMO on IBM Power6 cluster 29
Table 72. Execution times for ECHAM5 on Cray XT3 30
Table 73. Profile of ECHAM5 on 256 cores of Cray XT3 31
Table 74. Profile of ECHAM5 on 512 cores of Cray XT3 31
Table 75. Communication profile of ECHAM5 on 256 cores of Cray XT3 31
Table 76. Communication profile of ECHAM5 on 512 cores of Cray XT3 32
Table 77. I/O usage of ECHAM5 on 256 cores of Cray XT3 32
Table 78. I/O usage of ECHAM5 on 512 cores of Cray XT3 32
Table 79. Hardware counter data for ECHAM5 on Cray XT3 32
Table 80. Derived metrics for ECHAM5 on Cray XT3 33
Table 81. Execution times for CP2K on Cray XT4 34
Table 82. Memory usage for CP2K on Cray XT4 34
Table 83. Profile of CP2K on 256 cores of Cray XT4 34
Table 84. Communication profile of CP2K on 256 cores of Cray XT4 35
Table 85. Communication profile of CP2K on 512 cores of Cray XT4 35
Table 86. I/O usage of CP2K on 256 cores of Cray XT4 35
Table 87. I/O usage of CP2K on 512 cores of Cray XT4 35
Table 88. Hardware counter data for CP2K on Cray XT4 36
Table 89. Derived metrics for CP2K on Cray XT4 36
Table 90. Execution times for GROMACS on Cray XT4 37
Table 91. Memory usage for GROMACS on Cray XT4 38
Table 92. Profile of GROMACS on 256 cores of Cray XT4 38
Table 93. Profile of GROMACS on 512 cores of Cray XT4 38
Table 94. Data transfer for GROMACS on Cray XT4 39
Table 95. Communication profile of GROMACS on 256 cores of Cray XT4 39
Table 96. Communication profile of GROMACS on 512 cores of Cray XT4 39
Table 97. I/O usage of GROMACS on 256 cores of Cray XT4 39
Table 98. I/O usage of GROMACS on 512 cores of Cray XT4 39
Table 99. Hardware counter data for GROMACS on Cray XT4 40
Table 100. Derived metrics for GROMACS on Cray XT4 40
Table 101. Execution times for N3D on NEC SX-8 41
Table 102. Memory usage for N3D on NEC SX-8 41
Table 103. Profile of N3D on 256 cores of NEC SX-8 42
Table 104. Data transfer for N3D on NEC SX-8 42
Table 105. I/O usage of N3D on 256 cores of NEC SX-8 42
Table 106. I/O usage of N3D on 256 cores of NEC SX-8 42
Table 107. Hardware counter data for N3D on NEC SX-8 43
Table 108. Derived metrics for N3D on NEC SX-8 43
Table 109. Execution times for HELIUM on Cray XT4 44

Table 110. Memory usage for HELIUM on Cray XT4 44
Table 111. Profile of HELIUM on 253 cores of Cray XT4 45
Table 112. Profile of HELIUM on 496 cores of Cray XT4 45
Table 113. Data transfer for HELIUM on Cray XT4 45
Table 114. Communication profile of HELIUM on 253 cores of Cray XT4 45
Table 115. Communication profile of HELIUM on 496 cores of Cray XT4 45
Table 116. I/O usage of HELIUM on 253 and 496 cores of Cray XT4 46
Table 117. Hardware counter data for HELIUM on Cray XT4 46
Table 118. Derived metrics for HELIUM on Cray XT4 46
Table 119. Execution times for GPAW on Cray XT4 47
Table 120. Profile of GPAW on 256 cores of Cray XT4 48
Table 121. Profile of GPAW on 512 cores of Cray XT4 48
Table 122. Data transfer for GPAW on Cray XT4 48
Table 123. Communication profile of GPAW on 256 cores of Cray XT4 48
Table 124. Communication profile of GPAW on 512 cores of Cray XT4 48
Table 125. Hardware counter data for GPAW on Cray XT4 49
Table 126. Derived metrics for GPAW on Cray XT4 49
Table 127. Execution times for ALYA on IBM JS21 cluster 50
Table 128. Data transfer for ALYA on IBM JS21 cluster 50
Table 129. Communication profile of ALYA on 256 cores of IBM JS21 cluster 51
Table 130. Communication profile of ALYA on 512 cores of IBM JS21 cluster 51
Table 131. Hardware counter data for ALYA on IBM JS21 cluster 51
Table 132. Derived metrics for ALYA on IBM JS21 cluster 52
Table 133. Execution times for PEPC in normal mode on IBM Power6 cluster 53
Table 134. Execution times for PEPC in SMT mode on IBM Power6 cluster 53
Table 135. Memory usage for PEPC in normal mode on IBM Power6 cluster 53
Table 136. Profile of PEPC in normal mode on 256 cores of IBM Power6 cluster 53
Table 137. Communication profile of PEPC in normal mode on 256 cores of IBM Power6 cluster 54
Table 138. Hardware counter data for PEPC in normal mode on IBM Power6 cluster 54
Table 139. Derived metrics for PEPC on IBM Power6 cluster 55
Table 140. Comments on possible future petaflop/s systems. 62
Table 141. Applications used by responding groups on the HPC systems in the specified scientific fields. Multiple answers were allowed. 64
Table 142. Motivations for smaller jobs than the complete machine and the number of groups stating this reason (multiple answers allowed). 66
Table 143. Further comments regarding the usage of HPC systems. 67
Table 144. The conditions for the *No If* category of transferring data back to the home site of the user. 68
Table 145. Is pre/post processing possible to be done at the HPC center? 70
Table 146. Preferred ways of access to the super computing facility (multiple answers allowed). 71
Table 147. Additional useful tools. 71
Table 148. Comments regarding the HPC infrastructure. 71

References and Applicable Documents

[1] European Strategy Forum on Research Infrastructures (ESFRI) Annual Report, 2006. Available from: ftp://ftp.cordis.europa.eu/pub/esfri/docs/esfri-annual-report-2006_en.pdf
[2] Requirements of HPC Applications – Initial Analysis. PRACE Deliverable 6.2.1
[3] Identification and Categorisation of Applications and Initial Benchmarks Suite. PRACE Deliverable 6.1
[4] PAPI Homepage. http://icl.cs.utk.edu/papi/index.html

List of Acronyms and Abbreviations

CFD	Computational Fluid Dynamics
DEISA	Distributed European Infrastructure for Supercomputing Applications. EU project by leading national HPC centres.
DFT	Density Functional Theory
DGMEM	BLAS routine for double precision matrix-matrix operations
DMTAP	dimyristoyltrimethylammonium propane
DMPC	dimyristoylphosphatidylcholine
flop	Floating point operation.
FFT	Fast Fourier Transform.
FNC	Fat Node Cluster. A cluster architecture with at least 16 cores per node.
HPC	High Performance Computing; Computing at a high performance level at any given time; often used synonym of Supercomputing.
L1, L2, L3, etc	Levels of cache memory for a processor
LQCD	Lattice quantum chromodynamics.
LDA	Local Density Approximation
MD	Molecular dynamics.
MPI	Message Passing Interface. A library for message-passing programming.
MPP	Massively parallel processor. A distributed memory architecture with an integrated interconnect.
OpenMP	Open Multi-Processing. An API for shared-memory parallel programming.
PAPI	Performance Application Programming Interface, a consistent interface and methodology for use of the performance counter hardware found in most major microprocessors
PME	Particle-Mesh Ewald method for calculating interaction energies of periodic systems.
PRACE	Partnership for Advanced Computing in Europe; Project Acronym.
QCD	Quantum chromodynamics.
SPC	Simple Point Charge
SPH	Smoothed Particle Hydrodynamics
TNC	Thin Node Cluster. A cluster architecture with fewer than 16 cores per node.

Executive Summary

PRACE WP6 investigates the applications and other software that will run on future European petaflop/s systems. Many of the other workpackages (WPs) within PRACE need to understand the requirements from the key applications, so that they can plan the infrastructure and systems appropriately. This report analyses the requirements of a representative list of key applications spanning most scientific disciplines and that are broadly used across Europe.

The applications analysed in this report were previously identified in D6.1 as representative of the current and future usage of major European HPC systems. D6.1 focussed on surveys of the current HPC systems and their major applications and surveyed more than 20 systems including the major systems in most PRACE countries. This identified some 70 heavily-used applications which were investigated. From these applications, we identified representative subsets that were representative of the use of key algorithm classes and scientific areas. Later tasks in WP6 will involve optimising and petascaling these applications and packaging them into a benchmark suite to be used in future petaflop/s procurements.

This report builds on a previous analysis of applications requirements in D6.2.1. There are at least three significant enhancements to that report:

1. The applications analysed are based on those identified by PRACE (in D6.1) as representative of the likely usage of future systems, rather than on the DEISA benchmark suite. The list of applications used in this report was specifically chosen to be representative of the likely algorithms used and to span a broad range of scientific areas and countries.
2. The analysis of applications is based on actual performance data, rather than a questionnaire. For practical reasons, D6.2.1 focussed on an applications survey completed by expert users. This report uses performance and profiling data collected from architectures that were similar to those identified by WP7 as appropriate for prototypes of future petascale systems. From this data, we derive metrics which allow us to assess the different requirements of each application in terms of architectural features.
3. This data is complemented by the results of a survey of many of the major users in Europe. WP3 has identified a list of major users in most PRACE countries. A survey was sent to these users asking for their input on future applications requirements. This survey included questions about the user, usage patterns, HPC infrastructure, upcoming algorithms and general comments about future petascale systems. Almost 70 responses were received and analysed for this report.

So far, WP6 has therefore investigated:

- the usage of most major HPC systems across Europe;
- the key applications and algorithms used;
- the performance of these applications on key architectures;
- the views of major users on emerging applications requirements.

This should provide a full picture of the requirements of current applications and should identify how these would translate to future petaflop/s systems. This information should be useful for other WPs (such as WP5, 7 and 8) in helping to evaluate prototype systems and to prepare for future production systems.

1 Introduction

The Partnership for Advanced Computing in Europe (PRACE) has the overall objective to prepare for the creation of a persistent pan-European HPC service. PRACE is divided into a number of inter-linked work packages and WP6 focuses on the software for petascale systems.

PRACE is working towards a pan-European HPC infrastructure with a number of Tier-0 petascale systems. A need for such systems has been identified by the European Strategy Forum on Research Infrastructures (ESFRI) [1]. These systems would run from 2009/2010 onwards and would provide the computational resources for the major Grand Challenges in capability computing. Analysing the application requirements for these systems is a major challenge but is vital to ensure that the systems do meet the needs of European researchers.

The primary goal of PRACE WP6 is: to identify and understand the software libraries, tools, benchmarks and skills required by users to ensure that their application can use a Petaflop/s system productively and efficiently. WP6 is the largest of the PRACE technical work packages and involves all of the PRACE partners.

Task T6.1 has investigated the applications usage of the major European HPC systems and has, subsequently, identified lists of between 10 and 20 applications which:

- are representative of the current usage of HPC systems in Europe;
- span the scientific areas which exploit HPC;
- include examples of the various classes of algorithms;
- and which are widely used across Europe.

Task 6.2 is responsible for capturing and analysing the requirements of application codes across Europe. This work feeds directly into WP7 (Petaflop/s systems for 2009) which will translate this into architectural requirements for prototypes and future production systems. It will also feed into WP5 (Deployment of prototype systems) via the benchmark suite from Task 6.3. As information on the applications requirements was needed as early as possible in PRACE, the initial deliverable of this task (D6.2.1 [2]) chose an existing list of applications codes to investigate - those previously selected by DEISA, another European HPC project, to form part of their benchmark suite. Experts on each of the applications were sent a questionnaire and asked to estimate the importance for their code of a range of architectural characteristics. This data was analysed and, perhaps surprisingly, peak flop rate was the mostly highly rated factor, with communication latency following closely behind.

This deliverable (D6.2.2) is a significant enhancement on D6.2.1 as the list of applications used is based on the representative list from D6.1 [3]. Moreover, the data collected is more in-depth as it comes from runs of the applications using realistic data sets on appropriate HPC architectures. For each application, we report a set of performance data, from which are derived metrics which are as architecture-independent as possible. The metrics are than compared across the applications to assess the range of requirements in terms of key architectural features. The applications analysed in this report are intended to form a benchmark suite that will be packaged up by Task 6.3. The initial profiling of the applications performed for

D6.2.2 will also develop into more detailed work in petascaling (T6.4) and optimisation (T6.5) for these applications.

In addition, to complement the previous surveys of HPC systems and applications, D6.2.2 also includes a survey of the major HPC users across Europe to help build up a complete picture of the requirements for future production systems. As well as general comments about future petascale systems, the survey had questions on:

1. The user;
2. Usage patterns;
3. HPC infrastructure;
4. Upcoming algorithms.

This survey was sent to the Top 10 users in each PRACE country, as identified by WP3, and we have analysed almost 70 responses from these major users.

1.1 Structure of the Report

This report has two main sections: the first section is the analysis of the applications performance and profiling data; and the second section is the analysis of the data from the user survey. Within each section, we discuss how the data was collected and then this data collected is presented, described and analysed. The final chapter summarises this work and indicates how this will be further built upon.

2 Analysis of Applications

2.1 Methodology

In this Chapter, a performance analysis for each application in the current PRACE application set is presented. For each application, we give a brief summary of the code, then describe the benchmark dataset that was used for this study, and the hardware platform the application was executed on (N.B. in one case there is more than one platform). Due to the significant effort required in porting applications, it was decided to run all the applications on one platform each, with the choice of platform being dictated largely by convenience for the person responsible: i.e. a platform available at their local site.

For each application, a suitable data set and input parameters were chosen for a profiling run which was executed on 256 and 512 cores (where this was not possible 128 and 256 cores were used). The data size used was chosen to represent a realistic problem. The runtime was chosen to be long enough that startup effects were not significant, but not so long as to consume excessive resources. Where possible, the same problem was solved on both processor counts, so that strong scaling can be assessed. Each run was executed on a dedicated set of nodes: however, the interconnect and I/O systems were not guaranteed to be dedicated.

Note that in the timescales permitted by this exercise, and due to the unavailability of relevant tools on some systems, it has not been possible to collect all the required data for all the applications. Data is presented for 15 out of the 20 applications in the current PRACE application set. Of the other five applications, two (a set of QCD kernels, and BSIT, seismic migration code) are still under construction. Data was not available for AVBP, TRIPOLI_4 and SIESTA.

The performance analysis results for each application are divided into the following sections:

2.1.1 *Execution times*

The code was run on both core counts and the execution time recorded.

2.1.2 *Memory usage*

The amount of memory used per task (including the mean, minimum and maximum over tasks) was recorded.

2.1.3 *Profiling*

Each code was profiled on both processor counts and the most heavily used routines (which together account for 95% of the execution time) were identified.

2.1.4 *I/O*

The total amount of I/O performed (separated into reads and writes) was recorded.

2.1.5 *Communication*

The communications usage for both processor counts was recorded, including the following:

- Time spent in each communication routine used
- Total number of bytes transmitted

2.1.6 *CPU and cache*

Hardware counter data for both processor counts was obtained, recording as many of the following as possible (Note: some of these metrics have been derived from others, rather than measured directly). Where one exists, the relevant PAPI [4] event name is given in brackets. All data presented are averages over the number of cores.

- Total number of cycles (PAPI_TOT_CYC)
- Number of instructions completed (PAPI_TOT_INS)
- Number of floating point operations (PAPI_FP_OPS)
- Number of loads (PAPI_LD_INS)
- Number of stores (PAPI_SR_INS)
- Number of floating point loads (no PAPI standard event)
- Number of floating point stores (no PAPI standard event)
- Number of accesses to each level of data cache (PAPI_L1_DCA, PAPI_L2_DCA, PAPI_L3_DCA)
- Number of misses in each level of data cache (PAPI_L1_DCM, PAPI_L2_DCM, PAPI_L3_DCM)
- Number of accesses to main memory (PAPI_L*_DCM where * =2 or 3)
- Number of cycles stalled waiting for memory accesses (PAPI_MEM_SCY)

2.1.7 *Derived metrics*

From the above data, we have derived the following metrics for each application, where possible. These metrics have been chosen to be as machine independent and runtime independent as possible, to allow some meaningful comparison of applications run on different systems.

- Efficiency (set efficiency = 1 on the smaller core count)
- Achieved flop rate (total flops per cycle for all cores)
- Achieved flop rate (Gflops per second per core)
- Percentage of peak flop rate
- Percentage of instructions which are floating point
- Number of flops per load/store
- Ratio of stores to loads
- Number of cycles per L2 cache reference
- Number of cycles per L3 cache reference
- Number of cycles per main memory reference
- L1, L2 and L3 cache miss rates (as percentages)
- Number of cycles (per core) per byte communicated (per core)
- Number of cycles (per core) per byte I/O (all cores)

2.2 NAMD

2.2.1 *Summary*

NAMD is a parallel molecular dynamics code for high-performance simulation of large biomolecular systems.

Code author(s): J. C. Phillips and others	
Application areas: Computational Chemistry, Condensed Matter Physics, Life Sciences	
Language: C++	**Estimated lines of code**: 62,000
Parallelisation technique(s): charm++	
URL: http://www.ks.uiuc.edu/Research/namd/	

2.2.2 *Benchmark dataset*

The benchmark run is a simulation of F1-atpase, with 327,506 atoms, a cutoff of 11 Å, and with PME every fourth step. The simulation was run for 11,000 timesteps.

2.2.3 *Hardware platforms*

NAMD was run on two platforms: a Cray XT4 MPP and an IBM p575 FNC.

The EPSRC Cray XT4 system has 5664 AMD Opteron Dual Core 2.8 GHz chips (11328 cores in total). The peak flop rate is 5.6 GFlop/s per core. Each core has separate level 1 caches for data and instructions of 64 kB each. The L1 data cache is 2-way set associative. There is a combined data and instruction L2 cache of 1 MB for each core, which is 16-way associative. The L1 data and the L2 cache use 64 byte cache lines. The L2 cache acts as a victim cache for the L1 cache. Data evicted from the L1 cache gets established on the L2 cache. Each dual core chip has 3GB of main memory. The interconnect is a Cray SeaStar2 3D torus, and the I/O subsytem consists of 12 I/O nodes connected to 576 TB of RAID disks running Lustre.

The EPSRC IBM p575 cluster has 160 nodes, each containing 8 dual core Power 5 1.5GHz chips (2560 cores in total). The peak flop rate is 6.0 Gflop/s per core. Each core has a 32 Kbyte data cache and a 64 Kbyte instruction cache. The level 1 data cache has 128-byte lines, is 2-way set associative and write-through. The level 2 cache is on-chip, shared between the cores. It is a 1.9 Mbyte combined data and instruction cache, with 128 byte lines and is 10-way set associative and write-back. The level 3 cache is 36 Mbytes, off-chip and is shared between the 2 cores. It has 256 byte lines, and is 12-way set associative and write-back. Each node has 32GB of main memory. The interconnect is an IBM High Performance Switch (HPS). Each p575 node has two network adapters and there are two links per adapter, making a total of four links between each of the frames and the switch network. The I/O subsystem contains 72 Tbytes of disk running GPFS, connected to computes nodes via the HPS.

2.2.4 *Execution times*

No. of cores	**256**	**512**
Execution time (s)	347.5	193.4

Table 1. Execution times for NAMD on Cray XT4

No. of cores	256	512
Execution time (s)	487.7	285.1

Table 2. Execution times for NAMD on IBM p575 cluster

2.2.5 *Memory Usage*

No. of cores	256	512
Mean (MB)	87.96	86.69
Max (MB)	150.62	145.47
Min (MB)	86.48	85.11

Table 3. Memory usage for NAMD on Cray XT4

No. of cores	256	512
Mean (MB)	89.95	93.09
Max (MB)	197.00	203.07
Min (MB)	86.93	90.34

Table 4. Memory usage for NAMD on IBM p575 cluster

2.2.6 *Profiling*

Note that NAMD is a C++ code which relies heavily on inlining to achieve good performance. As a result, profilers may attribute time to parent routines.

Routine	% of execution time
_ZN20ComputeNonbondedUtil16calc_pair_energyEP9nonbonded	31.1
_ZN20ComputeNonbondedUtil26calc_pair_energy_fullelectEP9nonbonded	15.7
_ZN20ComputeNonbondedUtil16calc_self_energyEP9nonbonded	14.8
_ZN9Sequencer17runComputeObjectsEii	12.6
_ZN20ComputeNonbondedUtil26calc_self_energy_fullelectEP9nonbonded	6.7
_ZN7BackEnd4initEiPPc	4.3
_ZN14LdbCoordinator9rebalanceEP9Sequenceri	1.9
_ZN9HomePatch15doAtomMigrationEv	1.4
_ZN5Patch14positionsReadyEi	1.3

Table 5. Profile of NAMD on 256 cores of Cray XT4

Routine	% of execution time
_ZN20ComputeNonbondedUtil16calc_pair_energyEP9nonbonded	27.9
_ZN9Sequencer17runComputeObjectsEii	17.9
_ZN20ComputeNonbondedUtil26calc_pair_energy_fullelectEP9nonbonded	14.2
_ZN20ComputeNonbondedUtil16calc_self_energyEP9nonbonded	13.3
_ZN7BackEnd4initEiPPc	10.2
_ZN20ComputeNonbondedUtil26calc_self_energy_fullelectEP9nonbonded	6.2
_ZN14LdbCoordinator9rebalanceEP9Sequenceri	3.7
_ZN5Patch14positionsReadyEi	1.4
_ZN9HomePatch15doAtomMigrationEv	1.1

Table 6. Profile of NAMD on 512 cores of Cray XT4

Routine	% of execution time

.ComputeNonbondedUtil::calc_pair_energy(nonbonded*)	26.5
.ComputeNonbondedUtil::calc_pair_energy_fullelect(nonbonded*)	14.4
.ComputeNonbondedUtil::calc_self_energy(nonbonded*)	12.5
.ComputeNonbondedUtil::calc_self_energy_fullelect(nonbonded*)	5.9
.__divu64	3.1

Table 7. Profile of NAMD on 256 cores of IBM p575 cluster

Routine	% of execution time
.ComputeNonbondedUtil::calc_pair_energy(nonbonded*)	22.6
.ComputeNonbondedUtil::calc_pair_energy_fullelect(nonbonded*)	12.3
.ComputeNonbondedUtil::calc_self_energy(nonbonded*)	10.7
.ComputeNonbondedUtil::calc_self_energy_fullelect(nonbonded*)	5.1
.__divu64	4.8

Table 8. Profile of NAMD on 512 cores of IBM p575 cluster

2.2.7 *Communication*

No. of cores	256	512
Data transmitted per core (MB)	5018	3198

Table 9. Data transfer for NAMD on Cray XT4

MPI routine	% of execution time
MPI_Iprobe	7.44
MPI_Isend	2.16
MPI_Recv	1.23
Total	**10.82**

Table 10. Communication profile of NAMD on 256 cores of Cray XT4

MPI routine	% of execution time
MPI_Iprobe	12.40
MPI_Isend	2.31
MPI_Recv	1.38
Total	**16.09**

Table 11. Communication profile of NAMD on 512 cores of Cray XT4

MPI routine	% of execution time
MPI_Iprobe	10.78
MPI_Isend	2.66
MPI_Recv	0.82
MPI_Test	1.70
Total	**15.96**

Table 12. Communication profile of NAMD on 256 cores of IBM p575 cluster

MPI routine	% of execution time
MPI_Iprobe	16.59
MPI_Isend	1.86
MPI_Recv	1.23
MPI_Test	0.91
Total	**20.60**

Table 13. Communication profile of NAMD on 512 cores of IBM p575 cluster

2.2.8 *I/O*

Operation	Amount of data (MB)
Read	106.87
Write	18.73

Table 14. I/O usage of NAMD on Cray XT4

2.2.9 *CPU and cache Cray XT4*

Event	256 cores	512 cores
Cycles	9.3890E+11	5.2712E+11
Instructions	1.2412E+12	7.3632E+11
Flops	6.2134E+11	3.1129E+11
Loads	N/A	N/A
Stores	N/A	N/A
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	4.9728E+11	2.6194E+11
L2 cache references	6.3957E+09	3.3742E+09
L3 cache references	N/A	N/A
Memory references	9.0070E+08	5.1062E+08
L1 cache misses	N/A	N/A
L2 cache misses	N/A	N/A
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 15. Hardware counter data for NAMD on Cray XT4

Event	256 cores	512 cores
Cycles	7.3470E+11	4.3129E+11
Instructions	8.9638E+11	4.9839E+11
Flops	4.8645E+11	2.4352E+11
Loads	2.6820E+11	1.4915E+11
Stores	9.6437E+10	5.8008E+10
Floating point loads	1.7750E+11	8.9503E+10
Floating point stores	4.6436E+10	2.4020E+10
L1 cache references	3.6464E+11	2.0716E+11
L2 cache references	8.9144E+09	4.8795E+09
L3 cache references	3.0168E+08	1.6537E+08
Memory references	2.0046E+07	1.1207E+07
L1 cache misses	2.4361E+10	1.3992E+10
L2 cache misses	3.2204E+08	1.7677E+08
L3 cache misses	2.0046E+07	1.1207E+07
Memory stall cycles	N/A	N/A

Table 16. Hardware counter data for NAMD on IBM p575 cluster

2.2.10 *Derived metrics*

Metric	256 cores	512 cores
Efficiency	1.00	0.855
Flop rate (total per cycle)	169.4	302.3
Flop rate (Gflop/s per core)	0.93	0.82
% of peak flop rate	16.54	14.76
% of instructions which are floating point	50.06	42.28
Flops per load/store	N/A	N/A
Ratio of stores to loads	N/A	N/A
Cycles per L2 ref.	146.8	156.2
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	1042	1032
L1 cache miss rate	N/A	N/A
L2 cache miss rate	N/A	N/A
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	187.1	164.8
Cycles per byte of I/O	7475	4196

Table 17. Derived metrics for NAMD on Cray XT4

Metric	256 cores	512 cores
Efficiency	1.00	0.899
Flop rate (total per cycle)	169.5	289.1
Flop rate (Gflop/s per core)	0.99	0.85
% of peak flop rate	16.55	14.12
% of instructions which are floating point	54.27	48.86
Flops per load/store	1.33	1.18
Ratio of stores to loads	0.36	0.39
Cycles per L2 ref.	82.42	88.39
Cycles per L3 ref.	2435	2608
Cycles per memory ref.	36650	38483
L1 cache miss rate	6.68	6.75
L2 cache miss rate	3.61	3.62
L3 cache miss rate	6.64	6.78
Cycles per byte communicated	N/A	N/A
Cycles per byte of I/O	5849	3433

Table 18. Derived metrics for NAMD on IBM p575 cluster

2.2.11 *Analysis*

The key features of the data for NAMD are as follows:

- Modest memory usage (less than 100MB per core).
- Most of the computation is concentrated in four or five key routines.
- Communication is intensive. All the communication is point to point and heavy use is made of MPI_Iprobe.

- Reasonably good use of cache, though the L1 miss rate is rather high (on the Power5 system, this may be attributable to the no-cache-on-writes policy.
- High percentage of peak flop rate achieved.
- Reasonably high compute to load/store ratio.
- Low I/O usage (for this configuration).
- Modest scalability from 256 to 512 cores on this dataset.

The principal opportunity for optimisation is likely to be in improving the single CPU performance by tuning code for specific CPU architectures. A larger dataset is required for scalability tests on petascale systems.

2.3 CPMD

2.3.1 *Summary*

CPMD is a parallelized plane wave/pseudopotential implementation of Density Functional Theory, particularly designed for ab-initio molecular dynamics.

Code author(s): M. Parrinello, J. Hutter, A. Curioni and others	
Application areas: Computational Chemistry and Condensed Matter Physics	
Language: FORTRAN	**Estimated lines of code**: 40,000
Parallelisation technique(s): MPI and MPI+OpenMP	
URL: http://www.cpmd.org/	

2.3.2 *Benchmark dataset*

The test case used consists of 32 water molecules in the liquid phase at 100 Ry and MT pseudopotentials

2.3.3 *Hardware platforms*

The BSC IBM JS21 TNC contains 2560 nodes each with two IBM PPC970MX 2.3GHz dual core chips (10240 cores in total). The peak flop rate is 9.2 Gflop/s per core. Each core has a 2-way associate 32Kb data cache, and two cores share 1MB of combined L2 cache. Each node has 8 GB of main memory.
The interconnect is Myrinet, and the I/O subsystem consists of 20 storage server nodes with 7 terabytes of capacity each.

2.3.4 *Execution times*

No. of cores	**128**	**256**
Execution time (s)	158.87	78.74

Table 19. Execution times for CPMD on IBM JS21 cluster

2.3.5 *Memory Usage*

Not available

2.3.6 *Profiling*

Not available

2.3.7 *Communication*

Note: the instrumentation to record the communication profile adds significant overhead: the time reported here are not relative to the execution time of the uninstrumented runs reported above, and should only be used as an indication of the relative importance of each MPI routine.

No. of cores	**128**	**256**
Data transmitted per core (MB)	18.2	17.1

Table 20. Data transfer for CPMD on IBM JS21 cluster

MPI routine	**Time (s)**
MPI_Alltoall	63.44
MPI_Barrier	8.12
MPI_Allreduce	3.30
MPI_Init	1.08
MPI_Bcast	0.99
MPI_Recv	0.14
Total	**77.09**

Table 21. Communication profile of CPMD on 128 cores of IBM JS21 cluster

MPI routine	**Time(s)**
MPI_Alltoall	125.25
MPI_Allreduce	8.78
MPI_Barrier	6.87
MPI_Bcast	3.40
MPI_Recv	3.15
MPI_Init	2.40
Total	**150.01**

Table 22. Communication profile of CPMD on 256 cores of IBM JS21 cluster

2.3.8 *I/O*

Operation	**Amount of data (MB)**
Read	1.84
Write	322.46

Table 23. I/O usage of CPMD on 128 cores of IBM JS21 cluster

Operation	**Amount of data (MB)**
Read	2.17
Write	56.00

Table 24. I/O usage of CPMD on 256 cores of IBM JS21 cluster

2.3.9 *CPU and cache*

Event	**128 cores**	**256 cores**
Cycles	5.9089E+10	9.2518E+10
Instructions	3.1830E+10	5.0617E+10

Flops	N/A	N/A
Loads	1.2279E+10	1.8973E+10
Stores	6.3249E+09	1.0407E+10
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	1.8604E+10	2.9380E+10
L2 cache references	N/A	N/A
L3 cache references	N/A	N/A
Memory references	N/A	N/A
L1 cache misses	5.3693E+08	1.2098E+09
L2 cache misses	3.5904E+06	2.0487E+06
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 25. Hardware counter data for CPMD on IBM p575 cluster

2.3.10 *Derived metrics*

Metric	128 cores	256 cores
Efficiency	1.00	1.01
Flop rate (total per cycle)	N/A	N/A
Flop rate (Gflop/s per core)	N/A	N/A
% of peak flop rate	N/A	N/A
% of instructions which are floating point	N/A	N/A
Flops per load/store	N/A	N/A
Ratio of stores to loads	0.52	0.55
Cycles per L2 ref.	N/A	N/A
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	N/A	N/A
L1 cache miss rate	2.89	4.12
L2 cache miss rate	N/A	N/A
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	3242	5402
Cycles per byte of I/O	182.21	1590.32

Table 26. Derived metrics for CPMD on IBM JS21 cluster

2.3.11 *Analysis*

The key features of the data for CPMD are as follows:

- Communication is almost all in collectives, principally MPI_Alltoall
- Excellent scalability in this configuration.
- Modest I/O demands in this configuration.

Further data is required to understand the computational demands of CPMD. The communications should be investigated further to understand whether load imbalance is an important source of overhead. If not, then optimisations to MPI_Alltoall may be beneficial.

2.4 VASP

2.4.1 *Summary*

VASP is a package for performing ab-initio quantum-mechanical molecular dynamics (MD) simulations.

Code author(s): Jurgen Hafner, Jurgen Furthmuller	
Application area: Computational Chemistry and Condensed Matter Physics	
Language: FORTRAN90	**Estimated lines of code**: 100,000
Parallelisation technique(s): MPI	
URL: http://cms.mpi.univie.ac.at/vasp/	

2.4.2 *Benchmark dataset*

The test case is a simulation of the lanthan cuprite (La_2CuO_4) antiferromagnetic state, solving a pure DFT with LDA.

2.4.3 *Hardware platforms*

VASP was run on the BSC IBM JS21 TNC: see Section 2.3.3 for details.

2.4.4 *Execution times*

No. of cores	128	256
Execution time (s)	286.3	170.8

Table 27. Execution times for VASP on IBM JS21 cluster

2.4.5 *Memory Usage*

Not available

2.4.6 *Profiling*

Not available

2.4.7 *Communication*

Note: the instrumentation to record the communication profile adds significant overhead: the times reported here are not relative to the execution time of the uninstrumented runs reported above, and should only be used as an indication of the relative importance of each MPI routine.

No. of cores	128	256
Data transmitted per core (MB)	5.30	3.70

Table 28. Data transfer for VASP on IBM JS21 cluster

MPI routine	Time (s)
MPI_Alltoall	59.46
MPI_Bcast	24.48
MPI_Allreduce	11.73

MPI_Barrier	2.09
MPI_Init	1.21
MPI_Alltoallv	1.16
Total	**100.13**

Table 29. Communication profile of VASP on 128 cores of BM JS21 cluster

MPI routine	**Time (s)**
MPI_Alltoall	82.97
MPI_Bcast	53.37
MPI_Allreduce	8.68
MPI_Barrier	2.10
MPI_Init	1.53
MPI_Alltoallv	1.18
Total	**149.83**

Table 30. Communication profile of VASP on 256 cores of IBM JS21 cluster

2.4.8 *I/O*

Operation	**Amount of data (MB)**
Read	142.63
Write	105.88

Table 31. I/O usage of VASP on 128 cores of IBM JS21 cluster

Operation	**Amount of data (MB)**
Read	285.56
Write	113.18

Table 32. I/O usage of VASP on 256 cores of IBM JS21 cluster

2.4.9 *CPU and cache*

Event	**128 cores**	**256 cores**
Cycles	8.38E+10	1.26E+11
Instructions	4.73E+10	6.94E+10
Flops	N/A	N/A
Loads	2.51E+10	3.54E+10
Stores	9.14E+09	1.14E+10
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	3.43E+10	4.68E+10
L2 cache references	8.28E+08	7.90E+08
L3 cache references	N/A	N/A
Memory references	N/A	N/A
L1 cache misses	8.28E+08	7.90E+08
L2 cache misses	9.09E+07	1.72E+07
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 33. Hardware counter data for VASP on IBM p575 cluster

2.4.10 *Derived metrics*

Metric	128 cores	256 cores
Efficiency	1.00	0.838
Flop rate (total per cycle)	N/A	N/A
Flop rate (Gflop/s per core)	N/A	N/A
% of peak flop rate	N/A	N/A
% of instructions which are floating point	N/A	N/A
Flops per load/store	N/A	N/A
Ratio of stores to loads	0.36	0.32
Cycles per L2 ref.	101.29	158.93
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	N/A	N/A
L1 cache miss rate	2.41	1.69
L2 cache miss rate	10.98	2.18
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	15810	33900
Cycles per byte of I/O	337.37	314.81

Table 34. Derived metrics for VASP on IBM JS21 cluster

2.4.11 *Analysis*

The key features of the data for VASP are as follows:

- Communication time is almost all spent in collectives, principally MPI_Alltoall and MPI_Bcast.
- Cache utilisation is good, with low miss rates and infrequent L2 accesses.
- Modest I/O demands in this configuration.
- Modest scalability from 128 to 256 cores on this dataset.

A larger dataset is required for benchmarking on petascale systems. Collective communications are a possible target for optimisation.

2.5 GADGET

2.5.1 *Summary*

GADGET is a freely available code for cosmological N-body/SPH simulations on massively parallel computers with distributed memory.

Code author(s): V. Springel	
Application area: Astronomy and Cosmology	
Language: C	**Estimated lines of code**: 55,000
Parallelisation technique(s): MPI	
URL: http://www.mpa-garching.mpg.de/galform/gadget/index.shtml	

2.5.2 *Benchmark dataset*

A system containing more than 268 million particles.

2.5.3 *Hardware platforms*

GADGET was run on the SGI Altix 4700 FNC system at LRZ. The system has 19 nodes, each containing 256 Intel Itanium2 Montecito 1.6 Dual Core chips (9728 cores in total). The peak flop rate is 6.4 Gflop/s per core.

Each core has a 16KB, 4-way associative, L1 data cache, a 256KB, 8-way associative, L2 data cache and a 9MB, 12-way associative L3 cache. There is $ GB of main memory per core. The interconnect is SGI's NUMAlink 4. The I/O subsystem has 600 TB of disk, using CXFS.

2.5.4 *Execution times*

No. of cores	**256**	**512**
Execution time (s)	1265.89	705.70

Table 35. Execution times for GADGET on SGI Altix

2.5.5 *Memory Usage*

No. of cores	**256**	**512**
Mean (MB)	609	460
Max (MB)	1600	985
Min (MB)	561	398

Table 36. Memory usage for GADGET on SGI Altix

2.5.6 *Profiling*

Not available

2.5.7 *Communication*

No. of cores	**256**	**512**
Data transmitted per core (MB)	2530	1488

Table 37. Data transfer for GADGET on SGI Altix

MPI routine	**% of execution time**
MPI_Sendrecv	7.92
MPI_Allgather	5.89
MPI_Waitall	2.53
MPI_Allreduce	1.21
MPI_Barrier	0.35
MPI_Alltoall	0.21
MPI_Recv	0.15
MPI_Bcast	0.05

MPI_Irecv	0.04
MPI_Isend	0.03
MPI_Allgatherv	0.02
MPI_Ssend	0.02
MPI_Reduce	0.01
Total	**18.44**

Table 38. Communication profile of GADGET on 256 cores of SGI Altix

MPI routine	% of execution time
MPI_Allgather	11.65
MPI_Sendrecv	10.94
MPI_Waitall	8.35
MPI_Allreduce	3.04
MPI_Isend	1.08
MPI_Alltoall	1.07
MPI_Barrier	0.67
MPI_Recv	0.29
MPI_Allgatherv	0.13
MPI_Bcast	0.12
MPI_Irecv	0.10
MPI_Ssend	0.03
MPI_Reduce	0.03
Total	**37.51**

Table 39. Communication profile of GADGET on 512 cores of SGI Altix

2.5.8 *I/O*

Operation	Amount of data (MB)
Read	32
Write	257

Table 40. I/O usage of GADGET on 256 cores of SGI Altix

Operation	Amount of data (MB)
Read	32
Write	513

Table 41. I/O usage of GADGET on 512 cores of SGI Altix

2.5.9 *CPU and cache*

Event	256 cores	512 cores
Cycles	2.01E+12	1.11E+12
Instructions	3.69E+12	2.01E+12
Flops	4.81E+11	2.41E+11
Loads	3.55E+11	1.94E+11
Stores	2.93E+10	1.64E+10
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A

L1 cache references	1.62E+11	1.05E+11
L2 cache references	2.31E+11	1.17E+11
L3 cache references	1.41E+09	7.14E+08
Memory references	0.00E+00	0.00E+00
L1 cache misses	1.32E+10	6.84E+09
L2 cache misses	1.09E+09	5.53E+08
L3 cache misses	6.54E+08	3.18E+08
Memory stall cycles	N/A	N/A

Table 42. Hardware counter data for GADGET on SGI Altix

2.5.10 *Derived metrics*

Metric	256 cores	512 cores
Efficiency	1.00	0.90
Flop rate (total per cycle)	61.13	111.20
Flop rate (Gflop/s per core)	0.38	0.35
% of peak flop rate	5.97	5.43
% of instructions which are floating point	13.02	11.99
Flops per load/store	1.25	1.15
Ratio of stores to loads	0.08	0.08
Cycles per L2 ref.	8.72	9.46
Cycles per L3 ref.	1423	1556
Cycles per memory ref.	N/A	N/A
L1 cache miss rate	8.13	6.49
L2 cache miss rate	0.47	0.47
L3 cache miss rate	46.25	44.54
Cycles per byte communicated	795.4	747.3
Cycles per byte of I/O	6963	2040

Table 43. Derived metrics for GADGET on SGI Altix

2.5.11 *Analysis*

The key features of the data for GADGET are as follows:

- Memory usage is rather high (~0.5 GB per core)
- Communication demands are fairly high, with both point-to-point and collective communication being important.
- I/O demands are modest in this configuration
- Reasonably good scalability from 256 to 512 cores.
- Low percentage of peak flop rate achieved, despite a fairly high flop to load/store ratio.
- Very frequent accesses to L2 cache, but the L2 miss rate is low.

Optimisation opportunities may lie in the communications, and in improving the achieved flop rate on a single CPU. A larger dataset is required for benchmarking petascale systems.

2.6 CODE_SATURNE

2.6.1 *Summary*

Code_Saturne is a general purpose CFD code, used for nuclear thermalhydraulics, process, coal and gas combustion, aeraulics, etc.

Code author(s): F. Archambeau, N. Méchitoua, M. Sakiz, Y. Fournier	
Application areas: Computational Fluid Dynamics	
Language: C90 and FORTRAN77	**Estimated lines of code**: 400,000
Parallelisation technique(s): MPI	
URL: http://rd.edf.com/code_saturne	

2.6.2 *Benchmark dataset*

The test is carried out for an incompressible flow in a geometry consisting of a grid representing a bundle of pipes with fins attached. This 'mixing grid' geometry is so called as traditionally it is used to cool hot water pipes by mixing with cold water. However, in this test case the temperature is kept constant. Due to its complexity, the mesh is unstructured and built with tetrahedral cells (101M of active cells and 202M of faces). The flow being turbulent, the k-epsilon turbulent model is used, and a scalar is released into the flow. To ensure that the benchmark is carried out for a developed flow, a restarting procedure is activated, the simulation starting from the 16250th iteration.

2.6.3 *Hardware platforms*

The application was executed on EPSRC's IBM p575 FNC. See Section 2.2.3 for details.

2.6.4 *Execution times*

No. of cores	256	512
Execution time (s)	2227.66	1208.02

Table 44. Execution times for CODE_SATURNE on IBM p575 cluster

2.6.5 *Memory Usage*

No. of cores	256	512
Mean (MB)	1079	1000
Max (MB)	1090	1008
Min (MB)	1067	992

Table 45. Memory usage for CODE_SATURNE on IBM p575 cluster

2.6.6 *Profiling*

Not available

2.6.7 *Communication*

No. of cores	256	512
Data transmitted per core (MB)		

Table 46. Data transfer for CODE_SATURNE on IBM p575 cluster

MPI routine	% of execution time
MPI_Send	13.36
MPI_Allreduce	10.55
MPI_Barrier	1.61
MPI_Recv	1.48
MPI_Gather	0.96
MPI_Isend	0.50
MPI_Bcast	0.48
MPI_Waitall	0.26
MPI_Irecv	0.07
Total	**29.27**

Table 47. Communication profile of CODE_SATURNE on 256 cores of IBM p575 cluster

MPI routine	% of execution time
MPI_Send	29.24
MPI_Allreduce	13.79
MPI_Barrier	2.49
MPI_Recv	2.38
MPI_Gather	1.63
MPI_Bcast	1.62
MPI_Isend	0.50
MPI_Waitall	0.24
MPI_Irecv	0.08
Total	**51.98**

Table 48. Communication profile of CODE_SATURNE on 512 cores of IBM p575 cluster

2.6.8 *I/O*

Operation	Amount of data (MB)
Read	14251
Write	30600

Table 49. I/O usage of CODE_SATURNE on 256 cores of IBM p575 cluster

Operation	Amount of data (MB)
Read	14250
Write	30600

Table 50. I/O usage of CODE_SATURNE on 512 cores of IBM p575 cluster

2.6.9 *CPU and cache*

Event	128 cores	256 cores
Cycles	3.1900E+12	1.6870E+12
Instructions	2.5200E+12	1.7300E+12
Flops	3.9530E+11	1.9720E+11

Loads	8.9700E+11	6.0430E+11
Stores	2.9900E+11	2.3540E+11
Floating point loads	4.5100E+11	2.2600E+11
Floating point stores	1.4800E+11	7.4600E+10
L1 cache references	1.2000E+12	8.3970E+11
L2 cache references	6.1800E+10	3.6300E+10
L3 cache references	5.2900E+09	2.7420E+09
Memory references	1.2000E+09	2.0000E+08
L1 cache misses	6.1800E+10	3.6300E+10
L2 cache misses	6.4800E+09	2.9460E+09
L3 cache misses	1.2000E+09	2.0030E+08
Memory stall cycles	N/A	N/A

Table 51. Hardware counter data for CODE_SATURNE on IBM p575 cluster

2.6.10 *Derived metrics*

Metric	**256 cores**	**512 cores**
Efficiency	1.00	0.92
Flop rate (total per cycle)	31.72	59.85
Flop rate (Gflop/s per core)	0.19	0.18
% of peak flop rate	3.10	2.92
% of instructions which are floating point	15.69	11.40
Flops per load/store	0.33	0.23
Ratio of stores to loads	0.33	0.39
Cycles per L2 ref.	51.62	46.47
Cycles per L3 ref.	603.0	615.2
Cycles per memory ref.	2658	8435
L1 cache miss rate	5.15	4.32
L2 cache miss rate	10.49	8.12
L3 cache miss rate	22.68	7.30
Cycles per byte communicated	N/A	N/A
Cycles per byte of I/O	104.25	55.13

Table 52. Derived metrics for CODE_SATURNE on IBM p575 cluster

2.6.11 *Analysis*

The key features of the data for CODE_SATURNE are as follows:

- High memory requirements (around 1GB per core).
- Communication demands are fairly high, with both point-to-point and collective communication being important.
- Moderate I/O demands in this configuration.
- Reasonably good scalability from 256 to 512 cores.
- Low flop to load/store ratio, and low achieved percentage of peak flop rate.
- Frequent accesses to L2 cache with a fairly high miss rate in L2 and L3.

Optimisation opportunities may exist in improving locality, and in optimising communication. A larger dataset is required for benchmarking on petascale systems.

2.7 TORB

2.7.1 *Summary*

TORB simulates a plasma in a cylindrical *θ*-pinch configuration. The code solves the coupled system of gyrokinetic equations for the ions, in the electrostatic approximation, and the quasi-neutrality equation, assuming adiabatically responding electrons.

Code author(s): Jürgen Nührenberg, Francisco Castejon, Alejandro Soba	
Application areas: Plasma Physics	
Language: FORTRAN90	**Estimated lines of code**: 80,000
Parallelisation technique(s): MPI	
URL: N/A	

2.7.2 *Benchmark dataset*

The benchmark case uses a cylindrical equilibrium where the electromagnetic fields are parameterized and fixed. This equilibrium is shared in parts to be distributed in the different processors. The main application read these equilibrium files and runs the evolution of the particles in time in the cylinder, creating the trajectories of 1048574 particles.

2.7.3 *Hardware platforms*

TORB was run on the BSC IBM JS21 TNC: see Section 2.3.3 for details.

2.7.4 *Execution times*

No. of cores	128	256
Execution time (s)	57.16	62.49

Table 53. Execution times for TORB on IBM JS21 cluster

2.7.5 *Memory Usage*

Not available

2.7.6 *Profiling*

Not available

2.7.7 *Communication*

Note: the instrumentation to record the communication profile adds significant overhead: the times reported here are not relative to the execution time of the uninstrumented runs reported above, and should only be used as an indication of the relative importance of each MPI routine.

No. of cores	128	256
Data transmitted per core (MB)	4.48	4.55

Table 54. Data transfer for TORB on IBM JS21 cluster

MPI routine	Time (s)
MPI_Bcast	7.66
MPI_Barrier	2.26
MPI_Sendrecv	1.96
MPI_Init	1.29
MPI_Allreduce	1.17
MPI_Alltoall	0.01
Total	**14.35**

Table 55. Communication profile of TORB on 128 cores of BM JS21 cluster

MPI routine	Time (s)
MPI_Barrier	6.73
MPI_Allreduce	3.08
MPI_Init	2.82
MPI_Sendrecv	0.96
MPI_Bcast	0.65
MPI_Recv	0.04
MPI_Alltoall	0.01
Total	**14.30**

Table 56. Communication profile of TORB on 256 cores of IBM JS21 cluster

2.7.8 *I/O*

Operation	Amount of data (MB)
Read	487
Write	205

Table 57. I/O usage of TORB on 128 cores of IBM JS21 cluster

Operation	Amount of data (MB)
Read	488
Write	102

Table 58. I/O usage of TORB on 256 cores of IBM JS21 cluster

2.7.9 *CPU and cache*

Event	128 cores	256 cores
Cycles	8.8244E+10	1.0667E+10
Instructions	5.4134E+10	5.6347E+09
Flops	N/A	N/A
Loads	2.1901E+10	1.3394E+09
Stores	9.5131E+09	9.0459E+08
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	3.1414E+10	2.2440E+09
L2 cache references	3.9798E+08	1.6345E+07
L3 cache references	N/A	N/A
Memory references	N/A	N/A

L1 cache misses	3.9798E+08	1.6345E+07
L2 cache misses	3.4793E+06	1.4291E+06
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 59. Hardware counter data for TORB on IBM p575 cluster

2.7.10 *Derived metrics*

Metric	128 cores	256 cores
Efficiency	1.00	0.46
Flop rate (total per cycle)	N/A	N/A
Flop rate (Gflop/s per core)	N/A	N/A
% of peak flop rate	N/A	N/A
% of instructions which are floating point	N/A	N/A
Flops per load/store	N/A	N/A
Ratio of stores to loads	0.43	0.68
Cycles per L2 ref.	221.73	652.62
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	N/A	N/A
L1 cache miss rate	1.27	0.73
L2 cache miss rate	0.87	8.74
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	19706.42	2346.72
Cycles per byte of I/O	127.44	18.08

Table 60. Derived metrics for TORB on IBM JS21 cluster

2.7.11 *Analysis*

The key features of the data for TORB are as follows:

- This dataset exhibits no scalability at all between 128 and 256 cores
- A significant proportion of the communication time is spent in MPI_Barrier, which may be an indicator of load imbalance.
- There are very large differences in the hardware counters between the 128 and 256 core runs: this data should be treated with caution, and it is not possible to draw significant conclusions from it.

The configuration used here appears to be unsuitable for benchmarking even at this number of cores: a replacement will have to be found.

2.8 NEMO

2.8.1 *Summary*

NEMO is a numerical platform for simulating ocean dynamics and biochemistry, and sea-ice.

Code author(s): G. Madec and NEMO team	
Application area: Earth and climate science	
Language: FORTRAN90	**Estimated lines of code**: 100,000
Parallelisation technique(s): MPI, MPI+OpenMP, NEC autotasking	
URL: http://www.lodyc.jussieu.fr/NEMO/	

2.8.2 *Benchmark dataset*

NEMO has been run with the GYRE configuration at 0.25 degrees, for 600 timesteps.

2.8.3 *Hardware platforms*

NEMO was run on the IBM Power6 FNC at SARA. The system consists of 104 nodes, with 16 dual core processors (IBM Power6, 4.7 GHz) per node. The peak flop rate is 18.8 Gflop/s per core. Each core has a 64 KByte L1 data cache, and a 4MB L2 cache. 32 MB of L3 cache is shared between the 2 cores. There is either 128 GB or 256 GB of memory per node and a total of 700 TB of disk space divided between scratch space and home file systems. The interconnect is Infiniband.

2.8.4 *Execution times*

No. of cores	256	512
Execution time (s)	208.0	131.7

Table 61. Execution times for NEMO on IBM Power6 cluster

2.8.5 *Memory Usage*

No. of cores	256	512
Mean (MB)	201	155
Max (MB)	217	155
Min (MB)	197	152

Table 62. Memory usage for NEMO on IBM Power6 cluster

2.8.6 *Profiling*

Routine	% of execution time
ldfslp_NMOD_ldf_slp	12.9
traadv_muscl_NMOD_tra_adv_muscl	12.7
zdftke_NMOD_zdf_tke	12.1
traldf_iso_NMOD_tra_ldf_iso	11.1
solsor_NMOD_sol_sor	8.68
mathelp_NMOD_moycum	6.91
dynzdf_imp_NMOD_dyn_zdf_imp	6.63
trazdf_imp_NMOD_tra_zdf_imp	5.17
mathelp_NMOD_ma_ident_r31	2.52
lib_mpp_NMOD_mpp_lnk_3d	1.85
divcur_NMOD_div_cur	1.8

dynvor_NMOD_vor_ens	1.6
dynzad_NMOD_dyn_zad	1.56
dynspg_flt_NMOD_dyn_spg_flt	1.23
dynldf_lap_NMOD_dyn_ldf_lap	1.23
mathelp_NMOD_trans_buff	1.22
zdfevd_NMOD_zdf_evd	1.15
dynkeg_NMOD_dyn_keg	1.06
eosbn2_NMOD_eos_bn2	0.96
dynhpg_NMOD_hpg_zco	0.91
ldfslp_NMOD_ldf_slp_mxl	0.75
eosbn2_NMOD_eos_insitu_pot	0.62
stpctl_NMOD_stp_ctl	0.45

Table 63. Profile of NEMO on 256 cores of IBM Power6 cluster

Routine	% of execution time
ldfslp_NMOD_ldf_slp	12.8
zdftke_NMOD_zdf_tke	12.6
traadv_muscl_NMOD_tra_adv_muscl	11.9
traldf_iso_NMOD_tra_ldf_iso	10.7
solsor_NMOD_sol_sor	8.56
mathelp_NMOD_moycum	7.25
dynzdf_imp_NMOD_dyn_zdf_imp	6.82
trazdf_imp_NMOD_tra_zdf_imp	5.04
lib_mpp_NMOD_mpp_lnk_3d	2.44
mathelp_NMOD_ma_ident_r31	2.16
divcur_NMOD_div_cur	1.78
dynvor_NMOD_vor_ens	1.7
dynldf_lap_NMOD_dyn_ldf_lap	1.29
dynzad_NMOD_dyn_zad	1.23
dynspg_flt_NMOD_dyn_spg_flt	1.15
mathelp_NMOD_trans_buff	1.11
dynkeg_NMOD_dyn_keg	1.1
eosbn2_NMOD_eos_bn2	1.01
zdfevd_NMOD_zdf_evd	0.88
dynhpg_NMOD_hpg_zco	0.86
ldfslp_NMOD_ldf_slp_mxl	0.76
eosbn2_NMOD_eos_insitu_pot	0.61
stpctl_NMOD_stp_ctl	0.48

Table 64. Profile of NEMO on 512 cores of IBM Power6 cluster

2.8.7 *Communication*

No. of cores	256	512
Data transmitted per core (MB)	272	197

Table 65. Data transfer for NEMO on IBM Power6 cluster

MPI routine	% of execution time
Barrier	3.29
Recv	1.25

Send	0.60
Allreduce	0.18
Total	**5.32**

Table 66. Communication profile of NEMO on 256 cores of IBM Power6 cluster

MPI routine	% of execution time
Barrier	3.96
Recv	3.42
Send	0.78
Allreduce	0.30
Total	**8.46**

Table 67. Communication profile of NEMO on 512 cores of IBM Power6 cluster

2.8.8 *I/O*

Operation	Amount of data (MB)
Read	87552
Write	355840

Table 68. I/O usage of NEMO on 256 cores of IBM Power6 cluster

Operation	Amount of data (MB)
Read	29184
Write	345600

Table 69. I/O usage of NEMO on 512 cores of IBM Power6 cluster

2.8.9 *CPU and cache*

Event	256 cores	512 cores
Cycles	1.0900E+11	6.9000E+10
Instructions	2.7000E+11	6.7000E+10
Flops	9.8000E+10	4.9000E+10
Loads	1.1200E+11	4.4000E+10
Stores	4.2000E+10	2.2000E+10
Floating point loads	9.4000E+10	4.8000E+10
Floating point stores	2.8000E+10	1.5000E+10
L1 cache references	1.5500E+11	6.7000E+10
L2 cache references	1.9000E+10	7.0000E+09
L3 cache references	1.7700E+08	7.8000E+07
Memory references	1.0100E+08	3.7000E+07
L1 cache misses	1.9000E+10	7.0000E+09
L2 cache misses	1.7700E+08	7.8000E+07
L3 cache misses	1.0100E+08	3.7000E+07
Memory stall cycles	1.0900E+11	6.9000E+10

Table 70. Hardware counter data for NEMO on IBM Power6 cluster

2.8.10 *Derived metrics*

Metric	256 cores	512 cores
Efficiency	1	0.79
Flop rate (total per cycle)	230.2	363.6
Flop rate (Gflop/s per core)	4.22	3.34
% of peak flop rate	22.48	17.75
% of instructions which are floating point	36.30	73.13
Flops per load/store	0.64	0.74
Ratio of stores to loads	0.38	0.50
Cycles per L2 ref.	5.74	9.86
Cycles per L3 ref.	615.8	884.6
Cycles per memory ref.	1079	1865
L1 cache miss rate	12.26	10.45
L2 cache miss rate	0.93	1.11
L3 cache miss rate	57.06	47.44
Cycles per byte communicated	400.7	350.3
Cycles per byte of I/O	0.25	0.18

Table 71. Derived metrics for NEMO on IBM Power6 cluster

2.8.11 *Analysis*

The key features of the data for NEMO are as follows:

- Modest memory requirements
- A rather flat profile, with computation distributed across a large number of subroutines.
- Low communication demands. Most of the communication time is spent in MPI_Barrier, which may indicate that load imbalance is a problem.
- Very high I/O.
- Modest scalability between 256 and 512 cores.
- High percentage of peak flop rate
- High L1 miss rate, and frequent accesses to L2, but low L2 miss rate.

This benchmark makes far higher I/O demands than any other considered in this report. Load imbalance may require investigation. A larger dataset is required for benchmarking on petascale systems.

2.9 ECHAM5

2.9.1 *Summary*

ECHAM5 is the 5th generation of the ECHAM general circulation weather model. The model being used for PRACE benchmarking is ECHAM5-HAM which is a combination of the general atmospheric circulation model ECHAM5 and the atmospheric chemistry and aerosol model HAM

Code author(s): L. Kornblueth and E. Roeckner	
Application area: Earth and climate science	
Language: FORTRAN95/2003, some C	**Estimated lines of code**: 100,000
Parallelisation technique(s): MPI + OpenMP	
URL: http://www.mpimet.mpg.de/en/wissenschaft/modelle/echam/echam5.html	

2.9.2 *Benchmark dataset*

ECHAM5 was run with a T106L31 benchmark, simulating one month of model time.

2.9.3 *Hardware platforms*

ECHAM5 was run on the Cray XT3 MPP at CSCS. This system has 1664 AMD Opteron Dual Core 2.6 GHz chips (3328 cores in total). The peak flop rate is 5.2 Gflop/s per core. Each core has separate level 1 caches for data and instructions of 64 kB each. The L1 data cache is 2-way set associative. There is a combined data and instruction L2 cache of 1 MB for each core, which is 16-way associative. The L1 data and the L2 cache use 64 byte cache lines. The L2 cache acts as a victim cache for the L1 cache. Data evicted from the L1 cache gets established on the L2 cache. Each dual core chip has 2GB of main memory. The interconnect is a Cray SeaStar 1.2 3D torus. The I/O subsystem contains 32TB of disk, and uses Lustre.

2.9.4 *Execution times*

No. of cores	**256**	**512**
Execution time (s)	5178	3667

Table 72. Execution times for ECHAM5 on Cray XT3

2.9.5 *Memory Usage*

Not available.

2.9.6 *Profiling*

Routine	**% of execution time**
m7_coaset_	6.81
cloud_cdnc_icnc_	3.27
mo_aero_tools_aero_logtail_	2.84
xt_wetdep_	2.72
mo_aero_rad_aero_rad_fitplus_	2.40
mo_tpcore_ppm2m_	2.38
mo_tpcore_map1_ppm_gp_	1.92
vdiff_	1.92
xt_chemistry_	1.79
scan1_	1.79

mo_cirrus_xicehom_	1.72
xt_sedimentation_	1.65
m7_equil_	1.51
m7_dconc_	1.42
cuasc_	1.35
m7_averageproperties_	1.23
m7_equiz_	1.21
mo_mpi_p_recv_real_2d_	1.16
cuini_	1.14
m7_equimix_	1.11
mo_mpi_p_send_real_2d_	1.09

Table 73. Profile of ECHAM5 on 256 cores of Cray XT3

Routine	% of execution time
m7_coaset_	4.80
cloud_cdnc_icnc_	2.30
mo_aero_tools_aero_logtail_	1.92
xt_wetdep_	1.92
mo_tpcore_ppm2m_	1.70
mo_aero_rad_aero_rad_fitplus_	1.68
vdiff_	1.37
mo_tpcore_map1_ppm_gp_	1.36
mo_mpi_p_send_real_2d_	1.31
xt_chemistry_	1.29
mo_mpi_p_recv_real_2d_	1.29
scan1_	1.28
mo_cirrus_xicehom_	1.22
xt_sedimentation_	1.16
m7_equil_	1.06
m7_coaset_	4.80
cloud_cdnc_icnc_	2.30

Table 74. Profile of ECHAM5 on 512 cores of Cray XT3

2.9.7 *Communication*

MPI routine	% of execution time
MPI_sendrecv	16.96
MPI_barrier	6.35
MPI_bcast	3.73
MPI_recv	2.40
MPI_wait	2.71
Total	**32.15**

Table 75. Communication profile of ECHAM5 on 256 cores of Cray XT3

MPI routine	% of execution time
MPI_sendrecv	25.04
MPI_barrier	5.60
MPI_bcast	5.38
MPI_recv	3.68
MPI_wait	3.61
Total	**43.31**

Table 76. Communication profile of ECHAM5 on 512 cores of Cray XT3

2.9.8 *I/O*

Operation	Amount of data (MB)
Read	3270
Write	1116

Table 77. I/O usage of ECHAM5 on 256 cores of Cray XT3

Operation	Amount of data (MB)
Read	3270
Write	1116

Table 78. I/O usage of ECHAM5 on 512 cores of Cray XT3

2.9.9 *CPU and cache*

Event	256 cores	512 cores
Cycles	1.3611E+13	9.7666E+12
Instructions	1.0370E+13	7.2914E+12
Flops	2.2825E+12	1.1621E+12
Loads	N/A	N/A
Stores	N/A	N/A
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	4.3779E+12	3.0782E+12
L2 cache references	7.4191E+10	3.8324E+10
L3 cache references	N/A	N/A
Memory references	1.5051E+10	7.8425E+09
L1 cache misses	6.3586E+10	3.2716E+10
L2 cache misses	1.5051E+10	7.8425E+09
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 79. Hardware counter data for ECHAM5 on Cray XT3

2.9.10 *Derived metrics*

Metric	256 cores	512 cores
Efficiency	1.00	0.71
Flop rate (total per cycle)	42.93	60.92
Flop rate (Gflop/s per core)	0.44	0.31

% of peak flop rate	8.38	5.95
% of instructions which are floating point	22.01	15.94
Flops per load/store	N/A	N/A
Ratio of stores to loads	N/A	N/A
Cycles per L2 ref.	214.1	298.5
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	904.3	1245
L1 cache miss rate	1.45	1.06
L2 cache miss rate	23.67	23.97
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	N/A	N/A
Cycles per byte of I/O	3103	2227

Table 80. Derived metrics for ECHAM5 on Cray XT3

2.9.11 *Analysis*

The key features of the data for ECHAM5 are as follows:

- Communication is mainly in point-to-point routines (MPI_Sendrecv).
- Some time is spent in MPI_Barrier, which is an indicator of possible load imbalance.
- Rather poor scalability from 256 to 512 cores.
- This configuration make modest I/O demands.
- Moderate flop rate
- High L2 cache miss rate, though L2 accesses are not excessively frequent.

Possible load imbalance should be investigated. There is potential to optimise communications: the code currently uses a bespoke FFT implementation which needs updating. A larger dataset is required for benchmarking on petascale systems.

2.10 CP2K

2.10.1 *Summary*

A community code to perform atomistic and molecular simulations of solid state, liquid, molecular and biological systems. It consists of several components for classical molecular dynamics, ab-initio density functional theory. etc.

Code author(s): Juerg Hutter, Joost VandeVondele and others	
Application areas: Computational Chemistry and Condensed Matter Physics	
Language: FORTRAN90	**Estimated lines of code**: 500,000
Parallelisation technique(s): MPI	
URL: http://cp2k.berlios.de/	

2.10.2 *Benchmark dataset*

An ab-initio molecular dynamics run for 512 water molecules using density-functional theory and good quality one-particle basis sets was studied as a test case. This is an actual scientific case obtained from Prof. Kari Laasonen (University of Oulu, Finland), but in the analysis case the simulations were run only over 5 time steps of 1 fs length.

2.10.3 *Hardware platforms*

The measurements were carried out on a Cray XT4 MPP machine in CSC Finland. At the time of the measurements, the system consists of 1012 compute nodes with a 2.3 GHz quad-core Opteron and 1 or 2 GB RAM each running Compute Node Linux, and a Lustre file system. The peak flop rate is 9.2 Gflop/s per core. Each core has a 64Kbyte data cache and a 512KByte combined L2 cache. All four cores on a chip share a combined 2MByte L3 cache. The interconnect of the machine is a Cray propriety network Seastar 2+.

2.10.4 *Execution times*

No. of cores	**256**	**512**
Execution time (s)	658	560

Table 81. Execution times for CP2K on Cray XT4

2.10.5 *Memory Usage*

No. of cores	**256**	**512**
Mean (MB)	N/A	719.8
Max (MB)	N/A	N/A
Min (MB)	N/A	N/A

Table 82. Memory usage for CP2K on Cray XT4

2.10.6 *Profiling*

Routine	**% of execution time**
PW_NN_COMPOSE_R_WORK	22.4
DGEMM	15.6
PW_COMPOSE_STRIPE	11.5
main	6.2
CP_SM_FM_MULTIPLY_2D	1.9
malloc	1.5
free	1.5

Table 83. Profile of CP2K on 256 cores of Cray XT4

2.10.7 *Communication*

MPI routine	% of execution time
MPI_Bcast (sync)	11.20
MPI_Recv	5.90
MPI_Allreduce (sync)	3.00
MPI_Reduce (sync)	2.50
MPI_Alltoallv	2.50
MPI_Sendrecv	1.60
MPI_Alltoallv (sync)	1.20
MPI_Send	1.20
MPI_Waitall	1.10
Total	**31.20**

Table 84. Communication profile of CP2K on 256 cores of Cray XT4

MPI routine	% of execution time
MPI_Allreduce (sync)	33.40
MPI_Bcast (sync)	14.00
MPI_Recv	6.20
MPI_Reduce (sync)	2.20
MPI_Alltoallv	2.20
MPI_Waitall	1.60
MPI_Bcast	1.20
MPI_Alltoallv (sync)	1.00
Total	**62.80**

Table 85. Communication profile of CP2K on 512 cores of Cray XT4

2.10.8 *I/O*

Operation	Amount of data (MB)
Read	0.69
Write	1627

Table 86. I/O usage of CP2K on 256 cores of Cray XT4

Operation	Amount of data (MB)
Read	1.13
Write	1627

Table 87. I/O usage of CP2K on 512 cores of Cray XT4

2.10.9 *CPU and cache*

Event	256 cores	512 cores
Cycles	1.5791E+12	1.5329E+12
Instructions	2.4513E+12	2.2199E+12
Flops	1.0801E+12	5.4102E+11
Loads	N/A	N/A
Stores	N/A	N/A
Floating point loads	N/A	N/A

Floating point stores	N/A	N/A
L1 cache references	1.0734E+12	9.5078E+11
L2 cache references	2.3002E+10	1.3482E+10
L3 cache references	N/A	N/A
Memory references	5.4214E+09	4.0083E+09
L1 cache misses	8.6543E+09	6.7261E+09
L2 cache misses	5.4214E+09	4.0083E+09
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 88. Hardware counter data for CP2K on Cray XT4

2.10.10 *Derived metrics*

Metric	256 cores	512 cores
Efficiency	1	0.59
Flop rate (total per cycle)	175.10	180.70
Flop rate (Gflop/s per core)	1.57	0.81
% of peak flop rate	17.10	8.82
% of instructions which are floating point	44.06	24.37
Flops per load/store	N/A	N/A
Ratio of stores to loads	N/A	N/A
Cycles per L2 ref.	182.46	227.90
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	291.27	382.43
L1 cache miss rate	8.06	0.71
L2 cache miss rate	62.64	59.59
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	N/A	N/A
Cycles per byte of I/O	970.15	941.51
% of cycles stalled for memory	N/A	N/A

Table 89. Derived metrics for CP2K on Cray XT4

2.10.11 *Analysis*

The key features of the data for CP2K are as follows:

- Moderate memory requirements
- Computation is concentrated in four key routines
- A significant amount of time is spent in collective communications waiting for other tasks. This is an indicator of load imbalance.
- This configuration has modest I/O demands
- Percentage of peak flop rate is high.
- The scalability of this dataset between 256 and 512 cores is very poor

A larger dataset is required for benchmarking petascale systems. The load imbalance problems should be investigated. Single core performance appears well optimised.

2.11 GROMACS

2.11.1 *Summary*

GROMACS is a versatile package to perform molecular dynamics, i.e. simulate the Newtonian equations of motion for systems with hundreds to millions of particles. It is primarily designed for biochemical molecules like proteins and lipids that have a lot of complicated bonded interactions, but since GROMACS is extremely fast at calculating the nonbonded interactions (that usually dominate simulations) many groups are also using it for research on non-biological systems, e.g. polymers.

Code author(s): Erik Lindahl, David van der Spoel, Berk Hess	
Application areas: Computational Chemistry and Life Sciences	
Language: C and FORTRAN77	**Estimated lines of code**: 1,400,000
Parallelisation technique(s): MPI	
URL: http://www.gromacs.org	

2.11.2 *Benchmark dataset*

The test case is a simulation of a cationic lipid bilayer. This bilayer comprises positively charged DMTAP-lipids (7%), and neutral DMPC-lipids (93%). The bilayer is in water containing some NaCl. The system has in total 332496 (fused-)atoms in 128 DMTAP, 1920 DMPC and 78736 SPC water molecules.

The long-ranged potential model is calculated using the smooth particle-mesh Ewald (SPME) scheme. This scheme calculates the Fourier-space part of the long ranged forces using FFT by assigning the charges to a grid. As 3d-FFTs require all-to-all communication, GROMACS assigns a certain set of processor to only calculating the Fourier-space part of PME, while the others calculate normal short-ranged interactions. On the Cray XT4 with QC processors the optimal choice is to have one core per node calculating PME, while the others calculate short ranged forces. This gives optimal usage of the interconnect (as fast an all-to-all as possible). In Gromacs the 3d-FFT is currently only parallelized in 1-dimension, this unfortunately sets a great deal of restrictions on the grid-dimensions. In these systems the PME parameters were chosen to give a grid-size of 128x128x42, which enabled us divide the PME load evenly on 128 (64) cores for the 512 (256) core simulation. The short-ranged interactions were cut-off at 1.4 nm. The total number of iterations was 20000. The coordinates were written out every 5000 steps.

2.11.3 *Hardware platforms*

The measurements were carried out on a Cray XT4 machine in CSC Finland. See Section 2.10.3 for details.

2.11.4 *Execution times*

No. of cores	**256**	**512**
Execution time (s)	347	208

Table 90. Execution times for GROMACS on Cray XT4

2.11.5 *Memory Usage*

No. of cores	256	512
Mean (MB)	187.6	127.9
Max (MB)	527.4	292.0
Min (MB)	74.3	73.2

Table 91. Memory usage for GROMACS on Cray XT4

2.11.6 *Profiling*

Routine	% of execution time
do_nonbonded	34.00
gmx_pme_do	8.10
gmx_pme_recv_q_x	7.80
gather_f_bsplines	6.20
search_neighbours	5.30
dd_sendrecv_rvec	5.00
gmx_parallel_transpose_xy	2.90
constrain_lincs	2.80
gmx_sumd	2.70
make_bsplines	2.40
gmx_pme_receive_f	2.30
dd_sendrecv2_rvec	2.10
dih_angle	1.90
angles	1.70
ewald_LRcorrection	1.50
bond_angle	1.30
gmx_bcast	1.20
do_listed_vdw_q	1.10
pdihs	1.00

Table 92. Profile of GROMACS on 256 cores of Cray XT4

Routine	% of execution time
do_nonbonded	26.70
dd_sendrecv_rvec	11.50
gmx_sumd	9.50
gmx_pme_receive_f	8.40
gmx_pme_do	8.20
gmx_pme_recv_q_x	7.50
gather_f_bsplines	4.90
search_neighbours	4.30
gmx_parallel_transpose_xy	2.70
make_bsplines	1.80
dd_sendrecv2_rvec	1.50
gmx_bcast	1.40
angles	1.20
ewald_LRcorrection	1.20

Table 93. Profile of GROMACS on 512 cores of Cray XT4

2.11.7 *Communication*

No. of cores	256	512
Data transmitted per core (MB)	4453	3247

Table 94. Data transfer for GROMACS on Cray XT4

MPI routine	% of execution time
MPI_Recv	24.20
MPI_Sendrecv	5.79
MPI_Bcast(sync)	2.76
MPI_Alltoall	2.23
MPI_Waitall	1.37
MPI_Alltoall(sync)	1.34
MPI_Allreduce(sync)	0.81
MPI_Allreduce	0.33
MPI_Isend	0.18
MPI_Irecv	0.11
MPI_Bcast	0.07
Total	**39.20**

Table 95. Communication profile of GROMACS on 256 cores of Cray XT4

MPI routine	% of execution time
MPI_Recv	21.96
MPI_Sendrecv	21.02
MPI_Bcast(sync)	3.79
MPI_Alltoall	2.47
MPI_Waitall	1.63
MPI_Alltoall(sync)	2.47
MPI_Allreduce(sync)	1.60
MPI_Allreduce	0.48
MPI_Isend	0.21
MPI_Irecv	0.12
MPI_Bcast	0.11
Total	**55.86**

Table 96. Communication profile of GROMACS on 512 cores of Cray XT4

2.11.8 *I/O*

Operation	Amount of data (MB)
Read	N/A
Write	21.93

Table 97. I/O usage of GROMACS on 256 cores of Cray XT4

Operation	Amount of data (MB)
Read	N/A
Write	21.93

Table 98. I/O usage of GROMACS on 512 cores of Cray XT4

2.11.9 *CPU and cache*

Event	256 cores	512 cores
Cycles	8.5065E+11	5.6007E+11
Instructions	1.1863E+12	7.9251E+11
Flops	5.5317E+11	2.7630E+11
Loads	N/A	N/A
Stores	N/A	N/A
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	5.3451E+11	3.4820E+11
L2 cache references	N/A	N/A
L3 cache references	N/A	N/A
Memory references	N/A	N/A
L1 cache misses	1.8125E+09	1.0821E+09
L2 cache misses	N/A	N/A
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 99. Hardware counter data for GROMACS on Cray XT4

2.11.10 *Derived metrics*

Metric	256 cores	512 cores
Efficiency	1	0.83
Flop rate (total per cycle)	166.5	252.6
Flop rate (Gflop/s per core)	1.50	1.13
% of peak flop rate	16.26	12.33
% of instructions which are floating point	46.63	34.86
Flops per load/store	1.03	0.79
Ratio of stores to loads	N/A	N/A
Cycles per L2 ref.	469.3	517.6
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	N/A	N/A
L1 cache miss rate	0.34	0.31
L2 cache miss rate	N/A	N/A
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	191.0	172.5
Cycles per byte of I/O	38789	25539
% of cycles stalled for memory	N/A	N/A

Table 100. Derived metrics for GROMACS on Cray XT4

2.11.11 *Analysis*

The key features of the data for GROMACS are as follows:

- Low memory requirements.
- One routine accounts for around 30% of the execution time: otherwise the profile is very flat.
- Most of the communication is in point-to-point routines.
- This configuration has minimal I/O demands.
- The scalability is modest between 256 and 512 cores for this dataset.
- A high percentage of peak flop rate is observed.
- Very good cache utilisation.

A larger dataset is required for benchmarking on petascale systyems. Optimisation efforts should focus on communication.

2.12 N3D

2.12.1 *Summary*

N3D solves the incompressible Navier-Stokes equations by Direct Numerical Simulation (DNS).

Code author(s): Ulrich Rist, Markus Kloker	
Application areas: Computational Fluid Dynamics	
Language: FORTRAN90	**Estimated lines of code**: 40000
Parallelisation technique(s): MPI+ NEC microtasking	
URL: none	

2.12.2 *Benchmark dataset*

In this test case, 256 modes are used to discretise the z-direction.

2.12.3 *Hardware platforms*

The benchmark runs were carried out on the HLRS NEC SX-8 vector system. This system has 72 nodes, each with 8 NEC 2.0GHz vector processors (576 vector cores in total). The peak flop rate is 16 Gflop/s per core. Each node has 128 GB of main memory, and no vector cache. The interconnect is NEC's IXS multistage crossbar. The I/O subsystem is a Fibre Channel Network with global parallel filesystem.

2.12.4 *Execution times*

No. of cores	128	256
Execution time (s)	1724	928.7

Table 101. Execution times for N3D on NEC SX-8

2.12.5 *Memory Usage*

No. of cores	128	256
Mean (MB)	2562	1965

Table 102. Memory usage for N3D on NEC SX-8

2.12.6 *Profiling*

Routine	% of execution time
vradfg (FFT from netlib)	33.4
wtgmod.nlprod	17.3
uvwmod.wuber	9.5
ablmod.ablxx	7.6
uvwmod.vber	5.9
ablmod.ablx_pk	5.3
ablmod.ablx	4.9
ablmod.ablyy	2
parallelmod.f_to_p	1.9
ablmod.ably_nlt	1.4
wtgmod.wtg_spaceop	1.3
vsint	1.1
uvwmod.pendiy_4	1.1
parallelmod.p_to_f	1
wtgmod.wtg_tstep	0.9

Table 103. Profile of N3D on 256 cores of NEC SX-8

2.12.7 *Communication*

No. of cores	128	256
Data transmitted per core (MB)	63000	63000

Table 104. Data transfer for N3D on NEC SX-8

2.12.8 *I/O*

Operation	Amount of data (MB)
Read	0.06
Write	144

Table 105. I/O usage of N3D on 256 cores of NEC SX-8

Operation	Amount of data (MB)
Read	0.06
Write	144

Table 106. I/O usage of N3D on 256 cores of NEC SX-8

2.12.9 *CPU and cache*

Hardware counter data for 256 cores was not available.

Event	128 cores	256 cores
Cycles	3.4480E+12	N/A
Instructions	N/A	N/A

Flops	1.4125E+13	N/A
Loads	N/A	N/A
Stores	N/A	N/A
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	N/A	N/A
L2 cache references	N/A	N/A
L3 cache references	N/A	N/A
Memory references	N/A	N/A
L1 cache misses	N/A	N/A
L2 cache misses	N/A	N/A
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 107. Hardware counter data for N3D on NEC SX-8

2.12.10 *Derived metrics*

Metric	128 cores	256 cores
Efficiency	1.00	0.93
Flop rate (total per cycle)	524.4	N/A
Flop rate (Gflop/s per core)	8.19	N/A
% of peak flop rate	51.21	N/A
% of instructions which are floating point	N/A	N/A
Flops per load/store	N/A	N/A
Ratio of stores to loads	N/A	N/A
Cycles per L2 ref.	N/A	N/A
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	N/A	N/A
L1 cache miss rate	N/A	N/A
L2 cache miss rate	N/A	N/A
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	54.73	N/A
Cycles per byte of I/O	23944.44	N/A
% of cycles stalled for memory	N/A	N/A

Table 108. Derived metrics for N3D on NEC SX-8

2.12.11 *Analysis*

The key features of the data for N3D are as follows:

- Reasonably high scalability between 128 and 256 cores
- High memory requirements
- Most used routine is FFT, but profile is otherwise quite flat
- High communication load

- Very high percentage of peak flop rate achieved (though this might be expected on a vector system, and cannot be compared with other applications running on non-vector processors).

2.13 HELIUM

2.13.1 *Summary*

HELIUM simulates the behaviour of helium atoms using time-dependent solutions of the full-dimensional Schrödinger equation.

Code author(s): Jonathan Parker	
Application areas: Atomic Physics	
Language: FORTRAN90	**Estimated lines of code**: 14,500
Parallelisation technique(s): MPI	
URL: none	

2.13.2 *Benchmark dataset*

A 1364x1364 grid was used for the benchmark case. The simulation was run for 80 steps. Due to constraints on process geometries, HELIUM was run on 253 and 496 cores.

2.13.3 *Hardware platforms*

HELIUM was run on the EPSRC Cray XT4: for details see Section 2.2.3.

2.13.4 *Execution times*

No. of cores	**253**	**496**
Execution time (s)	1598	668

Table 109. Execution times for HELIUM on Cray XT4

2.13.5 *Memory Usage*

No. of cores	**253**	**496**
Mean (MB)	735	371
Max (MB)	N/A	N/A
Min (MB)	N/A	N/A

Table 110. Memory usage for HELIUM on Cray XT4

2.13.6 *Profiling*

Routine	**% of execution time**
local_ham_matrix_incr_result_w_1_over_r12_terms	50.99
global_linear_algebra_decrement_v_with_cx	8.48
local_ham_matrix_incr_with_1st_deriv_op_in_r1	6.42

local_ham_matrix_incr_with_1st_deriv_op_in_r2	6.34
propagators_arnoldi_propagate	4.84
mpi_communications_all_processor_barrier	4.27
hamiltonians_init_w_atomicham_x_psi	3.72
global_linear_algebra_real_local_inner_product	3.46
mpi_communications_get_fresh_remote_bndries	2.58
hamiltonians_incr_w_intham_x_psi	2.30
global_linear_algebra_increment_psi_with_zx	1.87
local_ham_matrix_incr_with_2nd_deriv_in_r1	1.73
global_linear_algebra_self_local_inner_product	1.61
local_ham_matrix_incr_with_2nd_deriv_in_r2	1.54
local_ham_matrix_incr_with_laplacian_in_r1_	1.04

Table 111. Profile of HELIUM on 253 cores of Cray XT4

Routine	**% of execution time**
local_ham_matrix_incr_result_w_1_over_r12_terms	41.77
global_linear_algebra_decrement_v_with_cx	10.59
local_ham_matrix_incr_with_1st_deriv_op_in_r1	8.13
mpi_communications_all_processor_barrier	6.44
local_ham_matrix_incr_with_1st_deriv_op_in_r2	6.29
propagators_arnoldi_propagate	6.14
hamiltonians_init_w_atomicham_x_psi	5.48
mpi_communications_get_fresh_remote_bndries	4.66
global_linear_algebra_real_local_inner_product	4.05
hamiltonians_incr_w_intham_x_psi	2.86
global_linear_algebra_self_local_inner_product	2.40
global_linear_algebra_increment_psi_with_zx	2.21
local_ham_matrix_incr_with_2nd_deriv_in_r2	1.86
local_ham_matrix_incr_with_2nd_deriv_in_r1_ (s)	1.85

Table 112. Profile of HELIUM on 496 cores of Cray XT4

2.13.7 *Communication*

No. of cores	**253**	**496**
Data transmitted per core (MB)	7574	5519

Table 113. Data transfer for HELIUM on Cray XT4

MPI routine	**% of execution time**
mpi_sendrecv_replace	1.94
Total	**1.94**

Table 114. Communication profile of HELIUM on 253 cores of Cray XT4

MPI routine	**% of execution time**
mpi_sendrecv_replace	3.49
Total	**3.49**

Table 115. Communication profile of HELIUM on 496 cores of Cray XT4

2.13.8 *I/O*

Operation	Amount of data (MB)
Read	0.00
Write	17981

Table 116. I/O usage of HELIUM on 253 and 496 cores of Cray XT4

2.13.9 *CPU and cache*

Event	253 cores	496 cores
Cycles	4.4280E+12	1.9284E+12
Instructions	3.1435E+12	1.6958E+12
Flops	1.6703E+12	8.4091E+11
Loads	N/A	N/A
Stores	N/A	N/A
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	1.8991E+12	1.0718E+12
L2 cache references	1.6147E+11	5.2996E+10
L3 cache references	N/A	N/A
Memory references	4.0141E+10	1.0963E+10
L1 cache misses	1.2123E+11	4.3410E+10
L2 cache misses	4.0141E+10	1.0963E+10
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 117. Hardware counter data for HELIUM on Cray XT4

2.13.10 *Derived metrics*

Metric	253 cores	496 cores
Efficiency	1	1.22
Flop rate (total per cycle)	95.44	216.29
Flop rate (Gflop/s per core)	1.06	1.22
% of peak flop rate	18.86	21.80
% of instructions which are floating point	53.14	49.59
Flops per load/store	N/A	N/A
Ratio of stores to loads	N/A	N/A
Cycles per L2 ref.	27.42	36.39
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	110.31	175.89
L1 cache miss rate	6.38	4.05
L2 cache miss rate	24.86	20.69
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	584.60	349.38
Cycles per byte of I/O	246.25	112.03

Table 118. Derived metrics for HELIUM on Cray XT4

2.13.11 *Analysis*

The key features of the data for HELIUM are as follows:

- Moderate memory requirements.
- One routine is responsible for about half the computation time, but the rest of the profile is very flat.
- Communication accounts for a small percentage of execution time, and is all point-to-point.
- This configuration has moderate I/O demands.
- Percentage of peak flop rate is high, despite relatively poor use of caches.
- Scalability is superlinear between 253 and 496 cores.

Optimisation effort should be focussed on single CPU performance.

2.14 GPAW

2.14.1 *Summary*

GPAW is a density-functional theory (DFT) code based on the projector-augmented wave (PAW) method. It uses real-space uniform grids and multigrid methods.

Code author(s): J. J. Mortensen, C. Rostgaard and others	
Application areas: Computational Chemistry and Condensed Matter Physics	
Language: C90 and Python	**Estimated lines of code**: 40,000
Parallelisation technique(s): MPI	
URL: https://wiki.fysik.dtu.dk/gpaw/	

2.14.2 *Benchmark dataset*

The test case used is a simple simulation of 256 water molecules.

2.14.3 *Hardware platforms*

The measurements were carried out on a Cray XT4 machine in CSC Finland See Section 2.10.3 for details.

2.14.4 *Execution times*

No. of cores	**256**	**512**
Execution time (s)	813.7	520.5

Table 119. Execution times for GPAW on Cray XT4

2.14.5 *Memory Usage*

Not available

2.14.6 *Profiling*

Routine	% of execution time
gemm	34.90
main	24.80
r2k	14.00
rk	7.40

Table 120. Profile of GPAW on 256 cores of Cray XT4

Routine	% of execution time
main	37.90
gemm	23.60
r2k	9.40
rk	4.90

Table 121. Profile of GPAW on 512 cores of Cray XT4

2.14.7 *Communication*

No. of cores	256	512
Data transmitted per core (MB)	1205	727

Table 122. Data transfer for GPAW on Cray XT4

MPI routine	% of execution time
MPI_Bcast(sync)	10.10
MPI_Wait	2.90
MPI_Allreduce(sync)	1.20
Total	**14.20**

Table 123. Communication profile of GPAW on 256 cores of Cray XT4

MPI routine	% of execution time
MPI_Bcast(sync)	14.00
MPI_Wait	3.70
MPI_Allreduce(sync)	1.40
Total	**19.10**

Table 124. Communication profile of GPAW on 512 cores of Cray XT4

2.14.8 *I/O*

Not available.

2.14.9 *CPU and cache*

Event	256 cores	512 cores
Cycles	1.8537E+12	1.2496E+12
Instructions	1.5446E+12	1.0935E+12
Flops	3.1063E+11	1.7051E+11
Loads	N/A	N/A
Stores	N/A	N/A

Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	7.2664E+11	4.9423E+11
L2 cache references	N/A	N/A
L3 cache references	N/A	N/A
Memory references	N/A	N/A
L1 cache misses	5.6419E+09	3.9366E+09
L2 cache misses	N/A	N/A
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 125. Hardware counter data for GPAW on Cray XT4

2.14.10 *Derived metrics*

Metric	256 cores	512 cores
Efficiency	1	0.78
Flop rate (total per cycle)	42.90	69.86
Flop rate (Gflop/s per core)	0.77	0.63
% of peak flop rate	8.38	6.82
% of instructions which are floating point	20.11	15.59
Flops per load/store	0.43	0.35
Ratio of stores to loads	N/A	N/A
Cycles per L2 ref.	N/A	N/A
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	N/A	N/A
L1 cache miss rate	0.78	0.80
L2 cache miss rate	N/A	N/A
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	1538.08	1718.37
Cycles per byte of I/O	N/A	N/A
% of cycles stalled for memory	N/A	N/A

Table 126. Derived metrics for GPAW on Cray XT4

2.14.11 *Analysis*

The key features of the data for GPAW are as follows:

- Computation is concentrated in a few key routines, including DGEMM.
- Communication demands are low, but a significant time is spent in collective operations waiting for other tasks.
- Scalability is modest between 256 and 512 cores for this dataset.
- Code achieves a moderate percentage of peak flop rate. The ratio of flops to load/stores is low, despite the use of DGEMM.

Optimisation effort should investigate possible load imbalance, and focus on tuning for single CPU performance.

2.15 ALYA

2.15.1 *Summary*

ALYA is a finite element code for Large Eddy Simulation of compressible and incompressible flows.

Code author(s): Guillaume Houzeaux, Mariano Vazquez, Jose M. Cela	
Application areas: Computational Fluid Dynamics	
Language: FORTRAN90	**Estimated lines of code**: 250,000
Parallelisation technique(s): MPI+OpenMP	
URL: none	

2.15.2 *Benchmark dataset*

The test case is a simulation of 3D compressible subsonic flow in a cavity.
It uses a fractional explicit scheme. There are 15.8 million nodes and 15.6 hexahedral elements with 8 integration Gauss points. The simulation is run for 20 iterations.

2.15.3 *Hardware platforms*

ALYA was run on the BSC IBM JS21 cluster: see Section 2.3.3 for details.

2.15.4 *Execution times*

Note that the execution times below exclude a start-up phase which reads in the restart file data.

No. of cores	**256**	**512**
Execution time (s)	194.4	97.6

Table 127. Execution times for ALYA on IBM JS21 cluster

2.15.5 *Memory Usage*

Not available

2.15.6 *Profiling*

Not available

2.15.7 *Communication*

Note: the data in this section includes the start-up phase, so is not directly comparable to the execution times reported above.

No. of cores	**256**	**512**
Data transmitted per core (MB)	29.38	19.14

Table 128. Data transfer for ALYA on IBM JS21 cluster

MPI routine	**% of execution time**
MPI_Sendrecv	7.20
MPI_Allreduce	4.47
MPI_Barrier	3.27

MPI_Init	1.59
MPI_Bcast	0.11
MPI_Recv	0.02
Total	**16.67**

Table 129. Communication profile of ALYA on 256 cores of IBM JS21 cluster

MPI routine	% of execution time
MPI_Allreduce	6.80
MPI_Barrier	5.29
MPI_Sendrecv	5.24
MPI_Init	3.61
MPI_Recv	0.37
MPI_Bcast	0.23
MPI_Comm_rank	0.04
MPI_Comm_size	0.01
Total	**21.59**

Table 130. Communication profile of ALYA on 512 cores of IBM JS21 cluster

2.15.8 *I/O*

This test case has minimal I/O requirements.

2.15.9 *CPU and cache*

Note: the data in this section includes the start-up phase, so is not directly comparable to the execution times reported above.

Event	256 cores	512 cores
Cycles	4.9270E+11	2.9262E+11
Instructions	5.4242E+11	3.1362E+11
Flops	N/A	N/A
Loads	1.7475E+11	1.0271E+11
Stores	1.2555E+11	7.5317E+10
Floating point loads	N/A	N/A
Floating point stores	N/A	N/A
L1 cache references	3.0030E+11	1.7803E+11
L2 cache references	4.3635E+10	2.2554E+10
L3 cache references	N/A	N/A
Memory references	N/A	N/A
L1 cache misses	4.3635E+10	2.2554E+10
L2 cache misses	3.5544E+07	1.5145E+07
L3 cache misses	N/A	N/A
Memory stall cycles	N/A	N/A

Table 131. Hardware counter data for ALYA on IBM JS21 cluster

2.15.10 *Derived metrics*

Metric	256 cores	512 cores
Efficiency	1	1.00
Flop rate (total per cycle)	N/A	N/A

Flop rate (Gflop/s per core)	N/A	N/A
% of peak flop rate	N/A	N/A
% of instructions which are floating point	N/A	N/A
Flops per load/store	N/A	N/A
Ratio of stores to loads	0.72	0.73
Cycles per L2 ref.	11.29	12.97
Cycles per L3 ref.	N/A	N/A
Cycles per memory ref.	N/A	N/A
L1 cache miss rate	14.53	12.67
L2 cache miss rate	0.08	0.07
L3 cache miss rate	N/A	N/A
Cycles per byte communicated	16772.12	15285.57
Cycles per byte of I/O	N/A	N/A
% of cycles stalled for memory	N/A	N/A

Table 132. Derived metrics for ALYA on IBM JS21 cluster

2.16 PEPC

2.16.1 *Summary*

PEPC is a parallel tree-code for computation of long-range Coulomb forces. The forces are calculated based on the Barnes-Hut algorithm. The code takes advantage of multipole-groupings of distant particles to reduce the original $O(N^2)$ scaling of the calculation to an O(N log N) scaling.

Code author(s): Paul Gibbon	
Application areas: Plasma Physics	
Language: FORTRAN90	**Estimated lines of code**: 24,500
Parallelisation technique(s): MPI	
URL: http://www.fz-juelich.de/jsc/pepc	

2.16.2 *Benchmark dataset*

A simulation of 5,000,000 particles run for 20 iteration steps.

2.16.3 *Hardware platforms*

The test case was run on the FZJ IBM Power6 p575 FNC. The system has 14 nodes, each with 16 dual-core Power6 4.7GHz chips (a total of 448 cores). The peak flop rate is 18.8 Gflop/s per core. Each core has a 64 KByte L1 data cache, and a 4MB L2 cache. 32 MB of L3 cache is shared between the 2 cores. Each node has 128Gbytes of main memory and the interconnect is Infiniband. The program was run in both normal mode (one thread per core) and in SMT mode (two threads per core) by doubling the number of MPI tasks.

2.16.4 *Execution times*

Note: these execution times exclude the start-up phase where input data is read from disk.

No. of cores	128	256
Execution time (s)	200	107

Table 133. Execution times for PEPC in normal mode on IBM Power6 cluster

No. of cores	128	256
Execution time (s)	66.7	38.6

Table 134. Execution times for PEPC in SMT mode on IBM Power6 cluster

2.16.5 *Memory Usage*

No. of cores	128	256
Mean (MB)	360	328
Max (MB)	353	321
Min (MB)	367	335

Table 135. Memory usage for PEPC in normal mode on IBM Power6 cluster

2.16.6 *Profiling*

Profiling data for 128 cores was not available.

Routine	% of execution time
sum_force	21.0
tree_walk	14.0
_barrier_onnode	11.9
mac_choose	8.0
key2addr	2.7
_stripe_hal_newpkts_FUiP11hal_param_t	2.6
_barrier_onnode_firstn	1.5
tree_build	1.3
LDScan	1.2

Table 136. Profile of PEPC in normal mode on 256 cores of IBM Power6 cluster

2.16.7 *Communication*

Note: this profile includes the start-up phase where data is read from disk, so these times are not directly comparable with the execution times given above. The total execution time for the profiled run is 302 s. Communication data for 128 cores was not available.

MPI routine	Time(s)
MPI_Barrier	44.45
MPI_Alltoallv	7.64
MPI_Allgather	6.09

MPI_Alltoall	1.13
MPI_Allreduce	0.58
MPI_Waitany	0.64
MPI_Isend	0.39
MPI_Allgatherv	0.08
MPI_Irecv	0.03
MPI_Reduce	0.03
Total	**61.07**

Table 137. Communication profile of PEPC in normal mode on 256 cores of IBM Power6 cluster

2.16.8 *I/O*

Not available.

2.16.9 *CPU and cache*

Event	128 cores	256 cores
Cycles	1.3570E+11	8.3471E+10
Instructions	1.0069E+11	6.2912E+10
Flops	1.5566E+10	7.8170E+09
Loads	2.5473E+10	1.5710E+10
Stores	1.2173E+10	7.9636E+09
Floating point loads	N/A	N/A
Floating point stores	1.7723E+09	7.1410E+08
L1 cache references	4.1040E+10	2.3527E+10
L2 cache references	1.0248E+10	6.0697E+09
L3 cache references	2.0503E+06	1.0648E+06
Memory references	4.8946E+05	2.1660E+05
L1 cache misses	1.8836E+09	9.8850E+08
L2 cache misses	4.3211E+07	4.4325E+07
L3 cache misses	7.1350E+05	3.8297E+05
Memory stall cycles	N/A	N/A

Table 138. Hardware counter data for PEPC in normal mode on IBM Power6 cluster

2.16.10 *Derived metrics*

Metric	128 cores	256 cores
Efficiency	1	0.93
Flop rate (total per cycle)	14.68	23.97
Flop rate (Gflop/s per core)	0.54	0.44
% of peak flop rate	2.87	2.34
% of instructions which are floating point	15.46	12.43
Flops per load/store	0.41	0.33
Ratio of stores to loads	0.48	0.51
Cycles per L2 ref.	72.04	84.44
Cycles per L3 ref.	66190	78390

Cycles per memory ref.	277300	385400
L1 cache miss rate	4.59	4.20
L2 cache miss rate	2.29	4.48
L3 cache miss rate	34.80	35.97
Cycles per byte communicated	N/A	238.23
Cycles per byte of I/O	N/A	N/A

Table 139. Derived metrics for PEPC on IBM Power6 cluster

2.16.11 *Analysis*

The key features of the data for PEPC are as follows:

- Moderate scalability on this dataset size.
- A very significant benefit is obtained from running in SMT mode. The reasons for this are not clear at present.
- Most of the computation is concentrated in 3 or 4 key routines.
- Collectives dominate the communications. The large time spent in MPI_Barrier is thought to occur in the start-up phase where input data is being read in.
- The percentage of peak flop rate is low, as is the number of flops per load store.
- References to L3 cache and memory are very infrequent.

A larger dataset is required for benchmarking on petascale systems. There should be possibilities to improve the single CPU performance.

2.17 Summary of Application Requirements

In this section we present an overview of the data presented in previous sections, in order to facilitate comparisons between the applications and to summarise the applications' requirements in terms of CPU, memory system, communication and I/O.

Each application was executed on two core counts. Figure 1 shows the efficiency for each application on the higher core count compared to the first: an efficiency of 1 corresponds to perfect scaling. About half the application show an efficiency of 0.9 or more, while a few application (ECHAM5, CP2K and TORB) scale poorly. There may be several reasons for this: the benchmark test cases may not be large enough, or else are not executing for long enough to make start-up effects insignificant. On the other hand, some applications may be intrinsically not scalable.

All the remaining data in this section is for 256 core executions (except for HELIUM which is on 253 cores). This is to permit direct comparisons between applications.

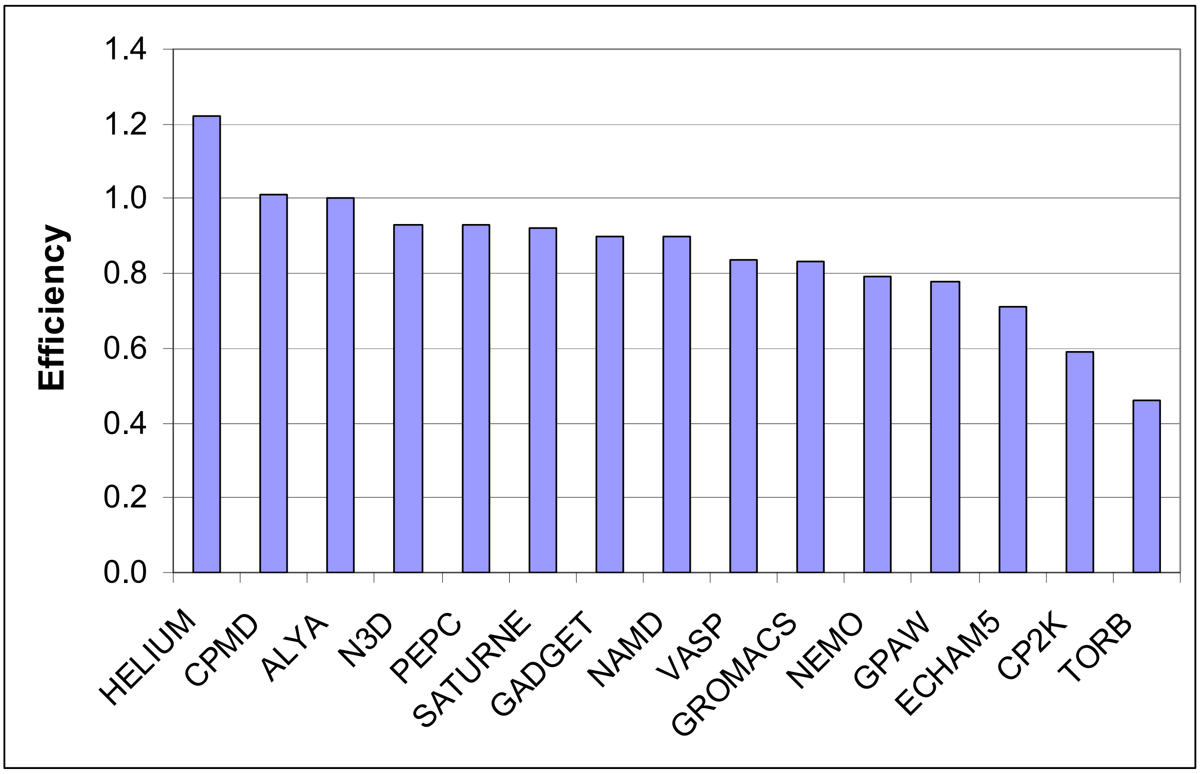


Figure 1. Application efficiency between the two core counts used

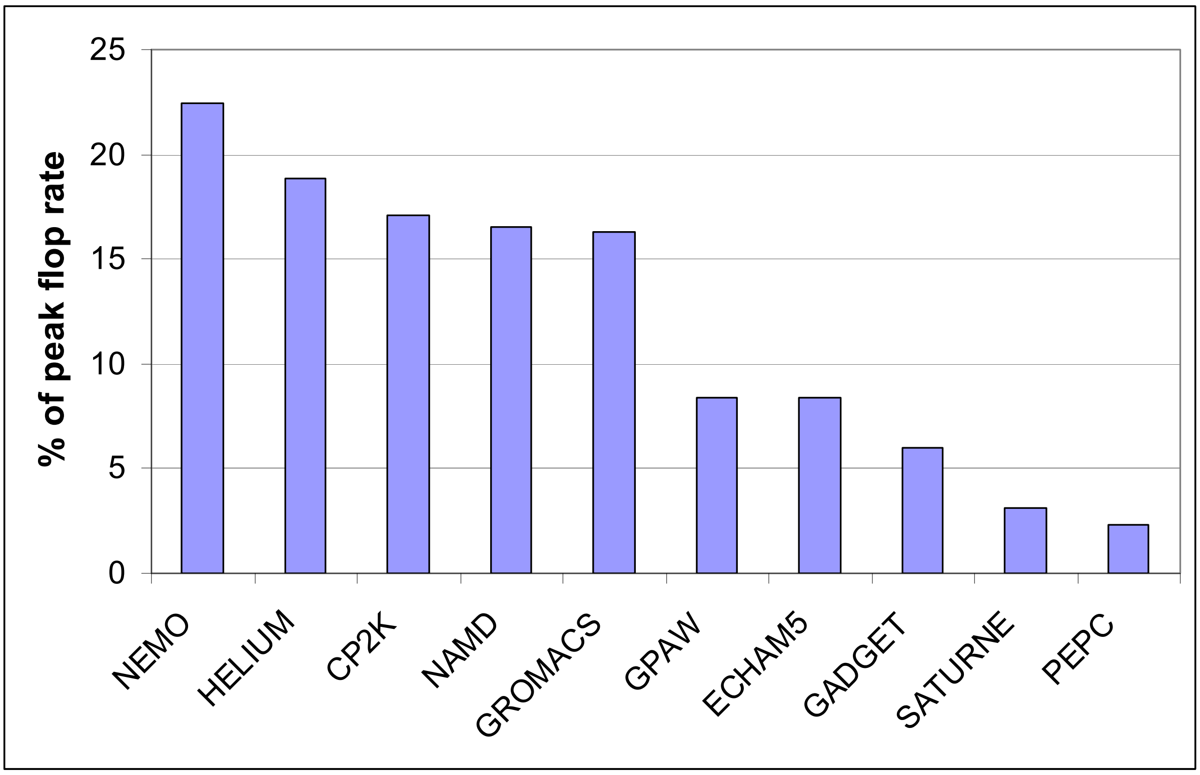


Figure 2. Percentage of peak flop rate attained

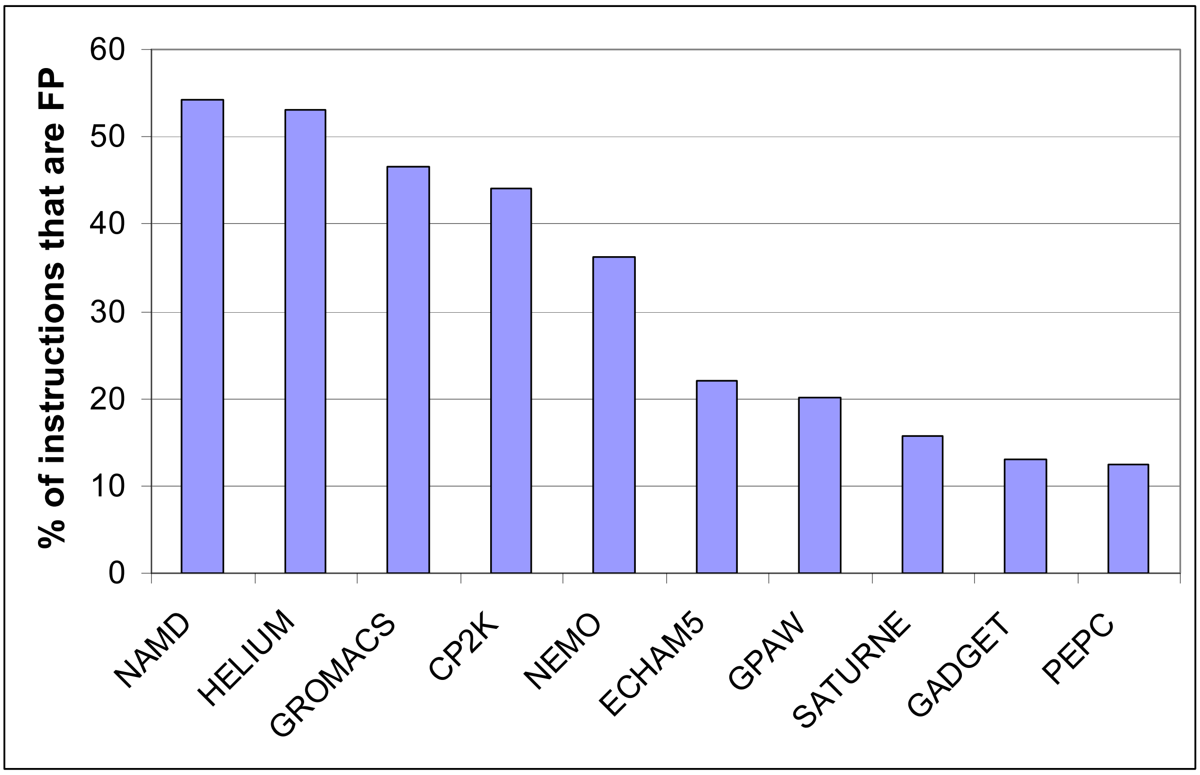


Figure 3. Percentage of instructions executed that are floating point

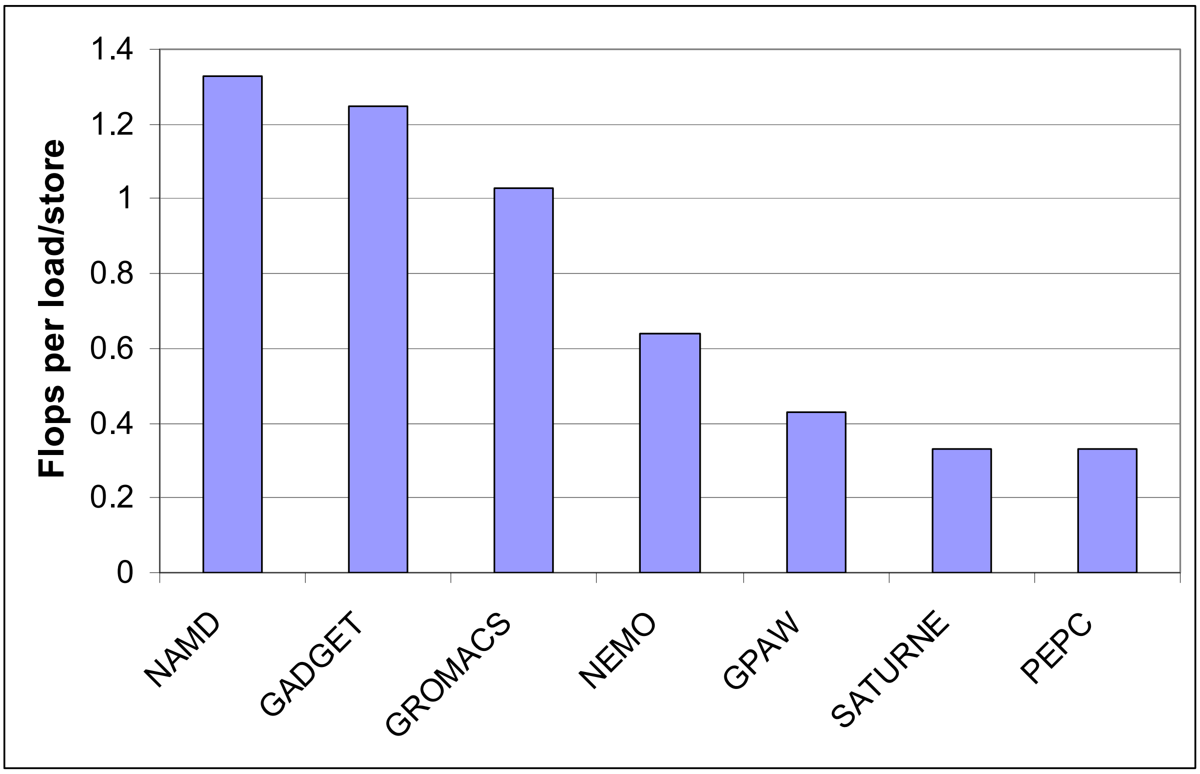


Figure 4. Ratio of floating point operations to loads and stores

Figure 2 shows the percentage of peak flop rate attained for each application where this measurement was possible. This metric varies from 2% to 22%: a figure of 10% is often considered reasonable on modern RISC processor architectures. Applications with high percentage of peak flop rate may benefit from CPUs with a higher clock rate, or which can execute more flops per cycle. Applications for which this metric is low will see little benefit from such enhancements to the hardware.

Figure 3 and Figure 4 show, respectively, the percentage of instructions executed which are floating point, and the ratio of floating point operations to load and store operations, for the applications where these metrics are available. Applications with high values of these metrics are floating point intensive, and are likely to benefit from enhanced floating point capability in CPUs. GROMACS is an interesting case: it has a low percentage of instructions which are floating point, but a high ration of flops to load/stores. This suggests that it performs a significant amount of non-floating point arithmetic (for example integer arithmetic, address calculations, or branches). This implies that it would benefit from a CPU with enhanced non-floating point capabilities.

Figure 5 shows the number of cycles per Level 2 cache reference for the applications where this metric was obtained. There is a high degree of variation (note the logarithmic scale) from around 5 for NEMO to over 650 for TORB. Figure 6 shows the number of cycles per main memory reference: again a high degree of variation is observed, from about 110 for HELIUM to nearly 400,000 for PEPC. Applications for which these metrics are low make frequent references to L2 cache and memory respectively. These applications will benefit from low latencies to, respectively, L2 cache and main memory.

Unfortunately it is unsafe to draw conclusions from these data about memory bandwidth requirements. The presence of hardware prefetching engines in almost all current supercomputer architectures means that applications which consume a lot of memory bandwidth do not necessarily record frequent accesses to main memory. If prefetching is successful, then the data is brought into L2 or L1 cache before it is referenced, so the hardware counters will record a cache hit. It is therefore very difficult, or even impossible, to get an accurate measure of the memory bandwidth consumed by applications.

Figure 7 shows the cycles per byte of communication per core for the applications where this metric was available. There are large variations, from under 200 for GROMACS to nearly 34,000 for VASP. Applications for which this metric is small will benefit from a high performance interconnect, whereas applications for which this metric is large will be much less sensitive to interconnect performance.

Figure 8 shows the cycles per byte of total I/O performed. Only NEMO shows a very high requirement for I/O performance. However, for many of the applications, the benchmark test case is chosen deliberately to exclude or minimise I/O. Furthermore, the I/O requirements of applications tend to vary significantly between different use cases. It is therefore difficult to make any firm conclusions about the I/O requirements of the applications from this data.

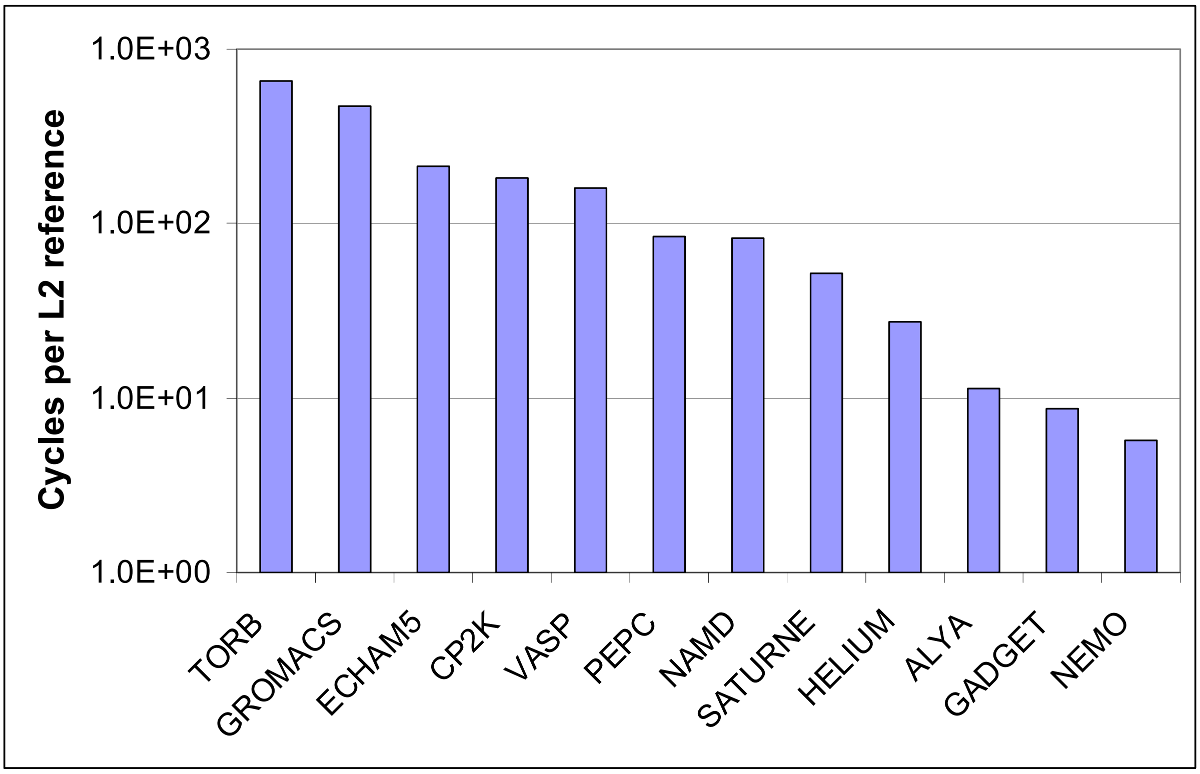


Figure 5. Cycles per L2 cache reference

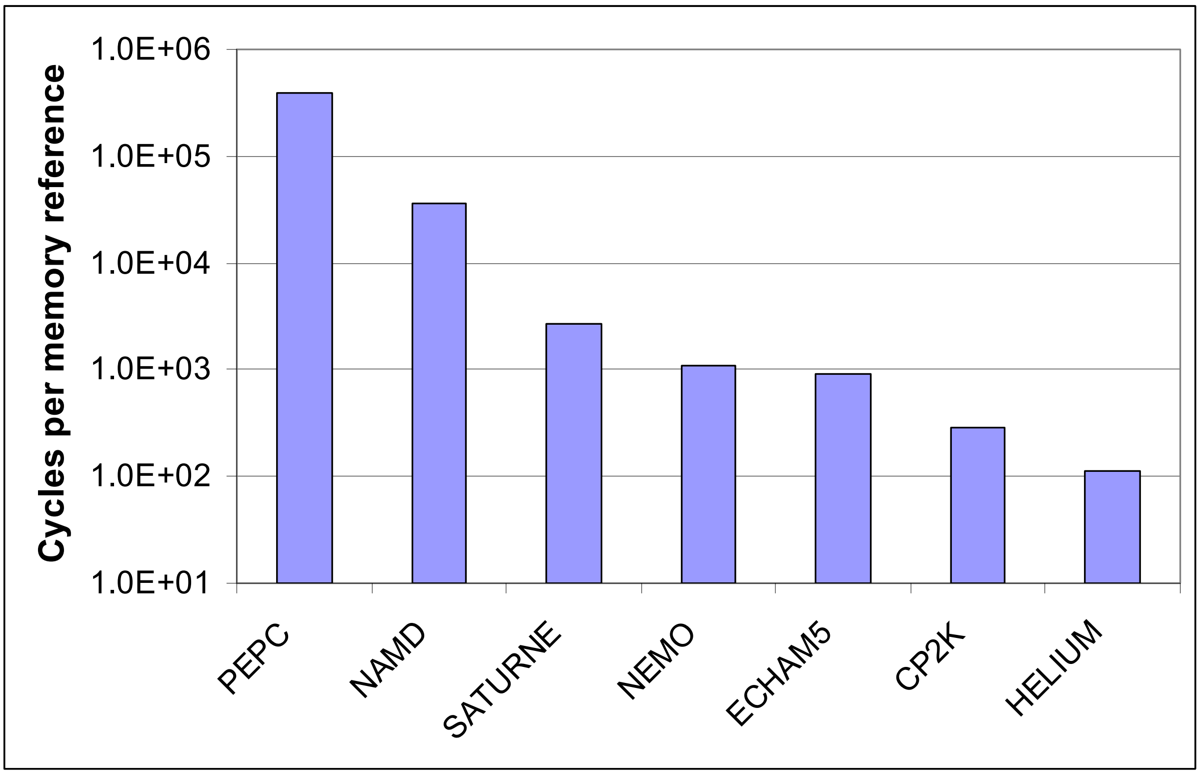


Figure 6. Cycles per memory reference

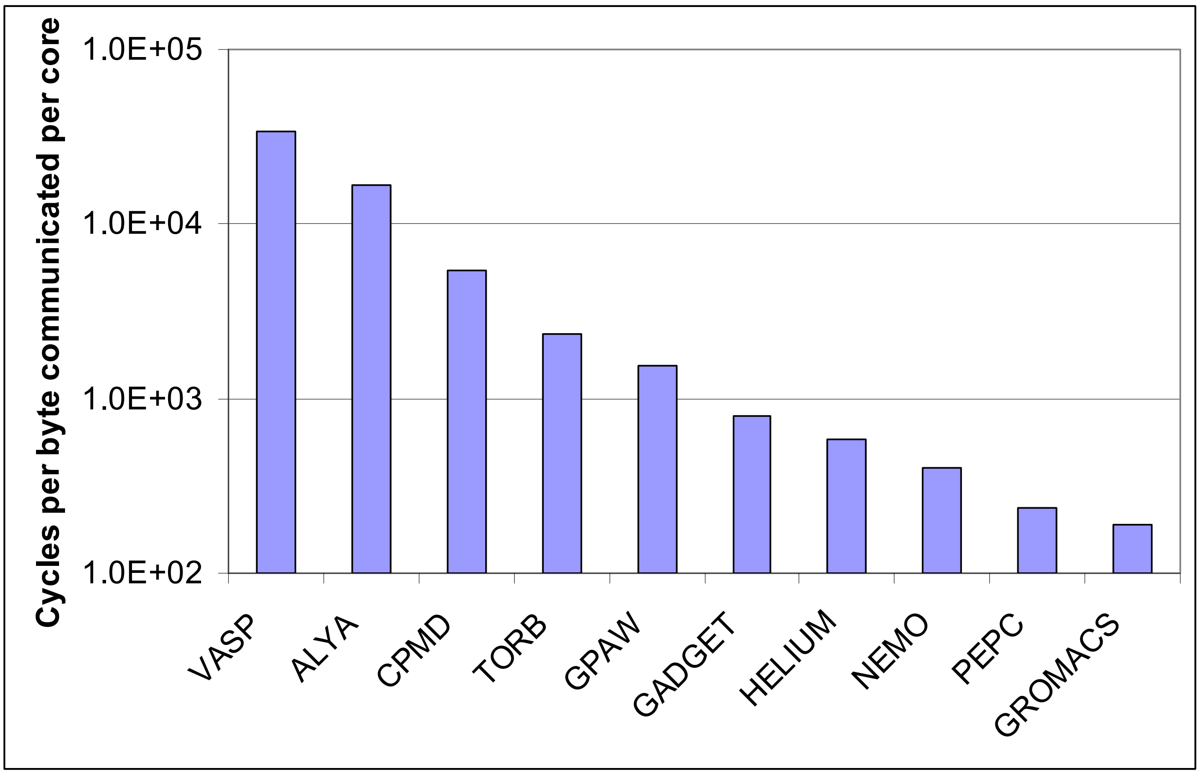


Figure 7. Cycles per byte communicated per core

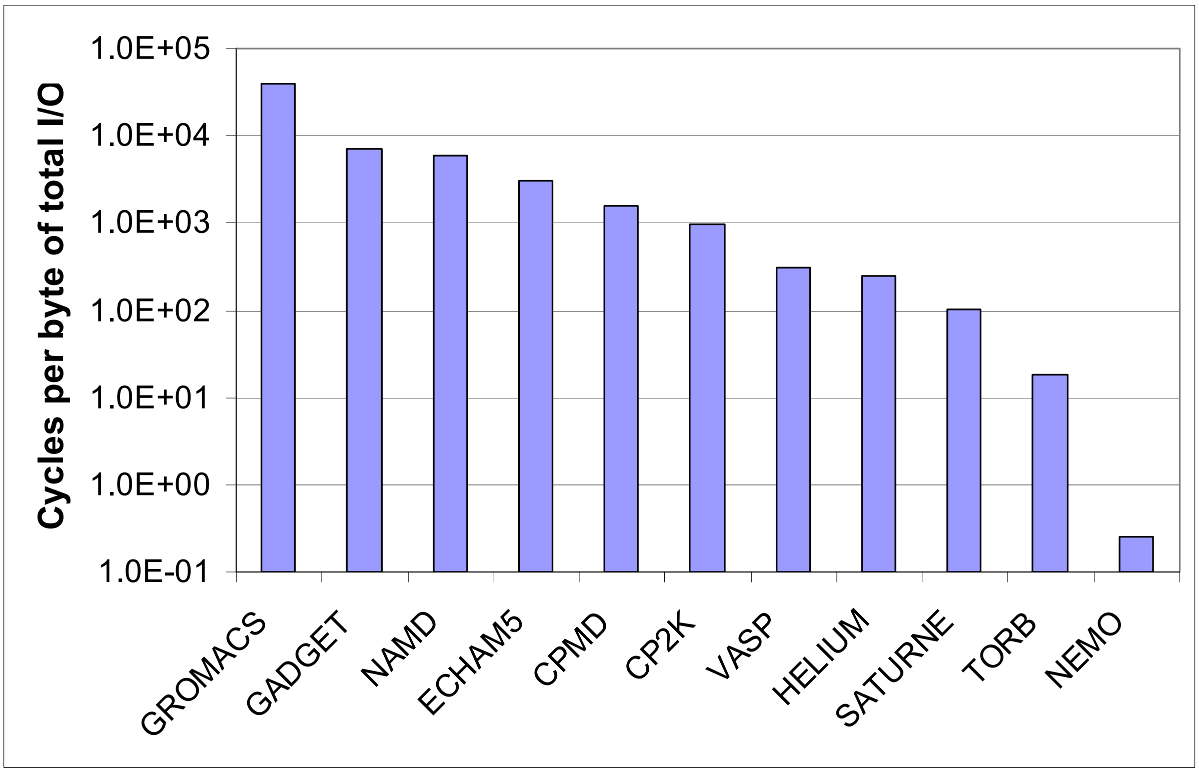


Figure 8. Cycles per byte of total I/O

3 Survey of Users

As the PRACE project is concerned with future platforms, we assessed the needs of European supercomputing users by gathering a survey on their current and planned usage of high performance systems. This is intended as a complementary investigation to the detailed analysis of currently deployed applications. The survey was sent to the Top 10 users of each PRACE partner, as provided by WP3. From these 160 people we got 68 replies to the survey. A copy of the survey can be found in Appendix A.

General overall comments on a potential petaflop/s system in Europe are listed in Table 140. This was the final question in the survey and displays some opinions of current HPC users. The main points that are expressed are concerning the memory, network, scalability, training, collaboration and reliability.

Do you have any other comments on possible future petaflop/s systems?
fast communication is needed
I think that we all realize that the programming model is going to change. It would be very useful to have advance warning of what is going to be available, to start planning algorithms and which problems to tackle.
We hope they will be ready as soon as possible :-)
Instead of a very high FLOP-rate we would prefer a large and fast amount of main memory per node.
Any petaflop/s system will clearly demand massively parallel applications. However, parallel programing is not one of the standard skills of the average scientist. Any such project is doomed to be a failure unless (very experienced !) software engineers do a serious work to help the research groups in the application enabling.
our applications need hardware with sufficient memory bandwidth. Architectures like the Cell Broadband Engine are therefore currently very hard to use efficiently. This would improve if the local store would be increased in future versions (rather than adding more SPEs).
Massively parallel machine desirable since communication is a bottleneck
There should be limits on the amount of time allocated to particular fields. QCD is *not* the only application of importance.
There will need to be very large memory machines with fewer cpus for analysis
We do relatively little I/O; 16 MB data cache per core would be desirable
HOT-Swap of CPU's should be possible, so that jobs running needn't be restarted everytime ONE single CPU fails. This is VERY time consuming.
Would like to see systems adapted to enable very large shared memory jobs requiring several Tb memory and simultaneously allow for very fast read/write of individual files also Tbs in size. Access to this memory and disk individual one off runs of up to a week would be ideal. This would enable very demanding quantum chemistry calculations (such as coupled cluster) to be run with codes like NWChem and Gaussian
I think it is important to find a way to do the post processing and the visualization easily!
We would like an emphasis on increasing bandwidth, rather than increasing the number of processors to help improve scaling.
per precessor memory should not fall below 1 GB
the ratio of memory per core should not deteriorate compared to the SX-8, which is a great machine for our purposes!
our code is vectorized (fluid dynamics) --- runs very effective on vector machines like SX-8
Now I run mostly in the USA, Jaguar at ORNL. petaflop/s would be desirable
They are needed. In the installations the user forum should be created and an efficient communication system between users and system managers established.

Information exchange between lattice community and computer manufacturers has in the past produced highly cost effective computers applicable for extremely demanding problems (e.g. APE, QCDOC, BG). This kind of collaboration could be useful for future petaflop/s systems.
Our applications scales perfectly on the Blue Gene architecture. Thus our application is highly scalable and such systems are fine for us. We need large memory and network bandwith, low memory and network latency: The performance is typically bandwidth limited.

Table 140. Comments on possible future petaflop/s systems.

The survey was divided into four groups of questions, regarding:
1. The user;
2. Usage patterns;
3. HPC infrastructure;
4. Upcoming algorithms.

The information gathered from these groups of questions is discussed in the following sections.

3.1 Information about the users

We received 68 survey responses, which is almost half of all the Top 10 users from all partners. These responses represented all the scientific fields covered in deliverable D6.2.1; their distribution is shown in Figure 9. The fields specified in the 'Other' category were:
- Theoretical Physics
- Econometrics
- Computational Quantum Mechanics
- Economics
- Nanooptics

The majority of answers were given by users from the field of computational chemistry and condensed matter physics.

It is interesting to note that most groups are developing their own codes, to be run on the HPC systems: 55 of the 68 responses stated, that the working group is developing the application themselves.

In Table 141 the applications that are used on the partner systems are listed along with the number of groups using it, and the scientific fields they are used in. The majority of codes ware specified as self written without further specifying a name. Multiple answers were allowed to this question on applications used.

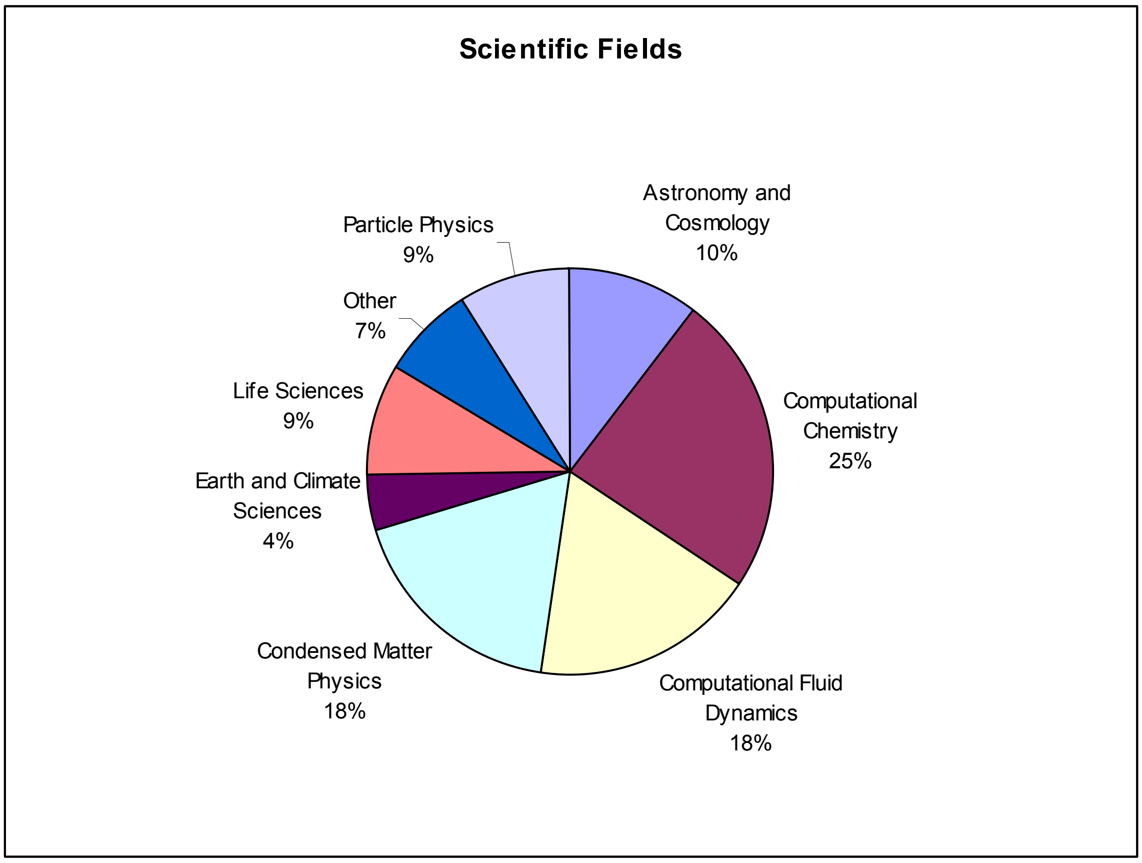


Figure 9. Share of responses by scientific fields.

The sizes of the working groups ranged from 2 people working together on their self written code, not used outside this group, to 100 people in a group working on a code that is also used by other groups. The mean working group consists of around 16 people.

Applications:				
Application	Nr	Field A	Field B	Field C
Unnamed self-written	26		Every Field	
Gaussian	9	Computational Chemistry	Life Sciences	
VASP	8	Computational Chemistry	Condensed matter physics	
CPMD	7	Computational Chemistry	Condensed matter physics	Life Sciences
CP2K	6	Computational Chemistry	Condensed matter physics	Life Sciences
NAMD	5	Condensed matter physics	Life Sciences	
AMBER	4	Computational Chemistry	Life Sciences	
Gamess	4	Computational Chemistry	Life Sciences	
Nwchem	3	Computational Chemistry	Condensed matter physics	
Molpro	3	Computational Chemistry	Condensed matter physics	Life Sciences
Quantum-Espresso	3	Computational Chemistry	Condensed matter physics	
Gromacs	3	Computational Chemistry	Life Sciences	
GADGET	2	Astronomy and Cosmology		
RAMSES	2	Astronomy and Cosmology		
Turbomole	2	Computational Chemistry	Condensed matter physics	
SIESTA	2	Computational Chemistry	Condensed matter physics	
Charmm	2	Computational Chemistry	Life Sciences	
Quickstep	2	Computational Chemistry		
DL_POLY	2	Computational Chemistry		
CASTEP	2	Computational Chemistry		

ADF	2	Computational Chemistry
Pkdgrav	1	Astronomy and Cosmology
Oslo Stagger Code	1	Astronomy and Cosmology
Gasoline	1	Astronomy and Cosmology
DYN5D	1	Computational Chemistry
Dirac	1	Computational Chemistry
Chemshell	1	Computational Chemistry
Tinker	1	Computational Chemistry
CRYSTAL	1	Computational Chemistry
Jaguar	1	Computational Chemistry
BAND	1	Computational Chemistry
Dalton	1	Computational Chemistry
Casino	1	Computational Chemistry
OpenFOAM	1	Computational Fluid Dynamics
NS-Solver	1	Computational Fluid Dynamics
AVBP	1	Computational Fluid Dynamics
Fenfloss	1	Computational Fluid Dynamics
Simson	1	Computational Fluid Dynamics
NSMB	1	Computational Fluid Dynamics
Wien2k	1	Condensed matter physics
FLEUR	1	Condensed matter physics
DMFT(QMC)	1	Condensed matter physics
DeCo	1	Condensed matter physics
Parla	1	Condensed matter physics
Stata/MP	1	Econometrics
R (GNU S)	1	Econometrics
XPLOR	1	Life Sciences
CNS	1	Life Sciences
UNRES	1	Life Sciences
DD-HMC	1	Particle Physics
Chroma	1	Particle Physics
BQCD	1	Particle Physics
PHMC	1	Particle Physics
openCMISS	1	Life Sciences
ASReml	1	Life Sciences

Table 141. Applications used by responding groups on the HPC systems in the specified scientific fields. Multiple answers were allowed.

The last aspect we asked in this group of user-related questions was about the usage of the deployed code outside the working group, and if a open source licence was used to release the application to a broader community. 42 groups stated an external usage of their codes, and 16 release the developments under an open source licence. Two of the groups with open source licences did not know of users outside their own working group.

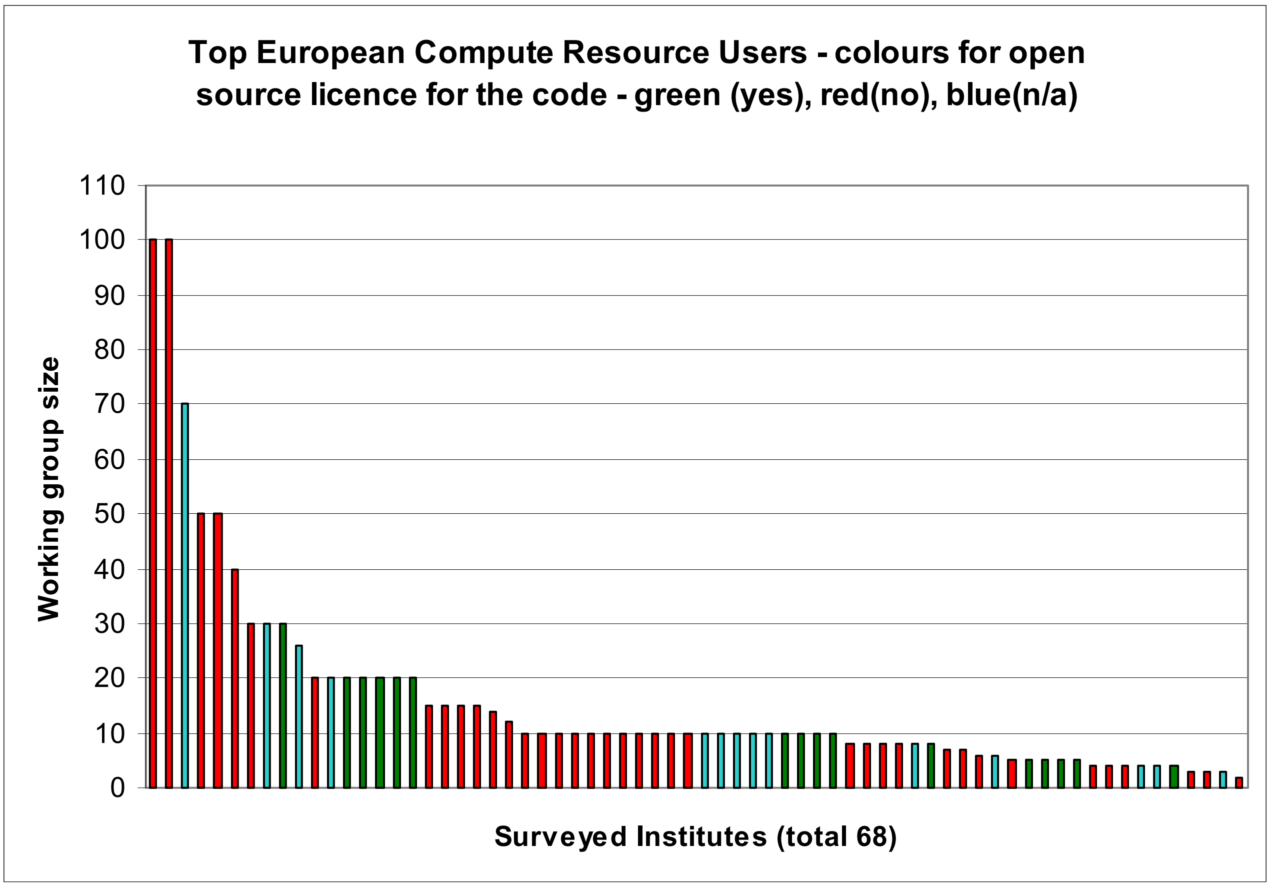


Figure 10. Working group sizes and their correlation with open source licenses.

In Figure 10, the working group sizes are shown with a colouring, which indicates the use of open source licences.

3.2 Usage patterns for HPC systems

In this group of questions, we captured the typical usage patterns on current super computing facilities, with the aim to find the requirements of the jobs that have to be run.

The first criteria we asked was the size of the job in terms of fraction of the executing machine: *What fraction of the machine does a job typically consume (how much computational power does a single job require; e.g. 25% of the number of available processors on Machine X)?* The answers ranged from below 1% up to 100%, with an mean of 22 %.

The distribution of the job sizes is shown in Figure 11 along the answers about the minimal job length: *How long has a job to be allowed to run in a minimum (e.g. jobs have to run at least 6 hours without restart)?* The combination of those two pieces of information provides the computational effort needed for a single job. Minimal job lengths ranged from 0 up to 720 hours, with an mean of 37 hours. Generally jobs requiring a larger part of the machine are not required to run as long as smaller jobs the average number of complete machine hours spent on a single job is nearly 4. The maximal number of machine hours is found in the job requesting the longest runtime of 720 hours for 10% of the machine (72 machine hours). The full machine is only requested for a runtime of 24 hours in maximum, resulting in 24 machine hours.

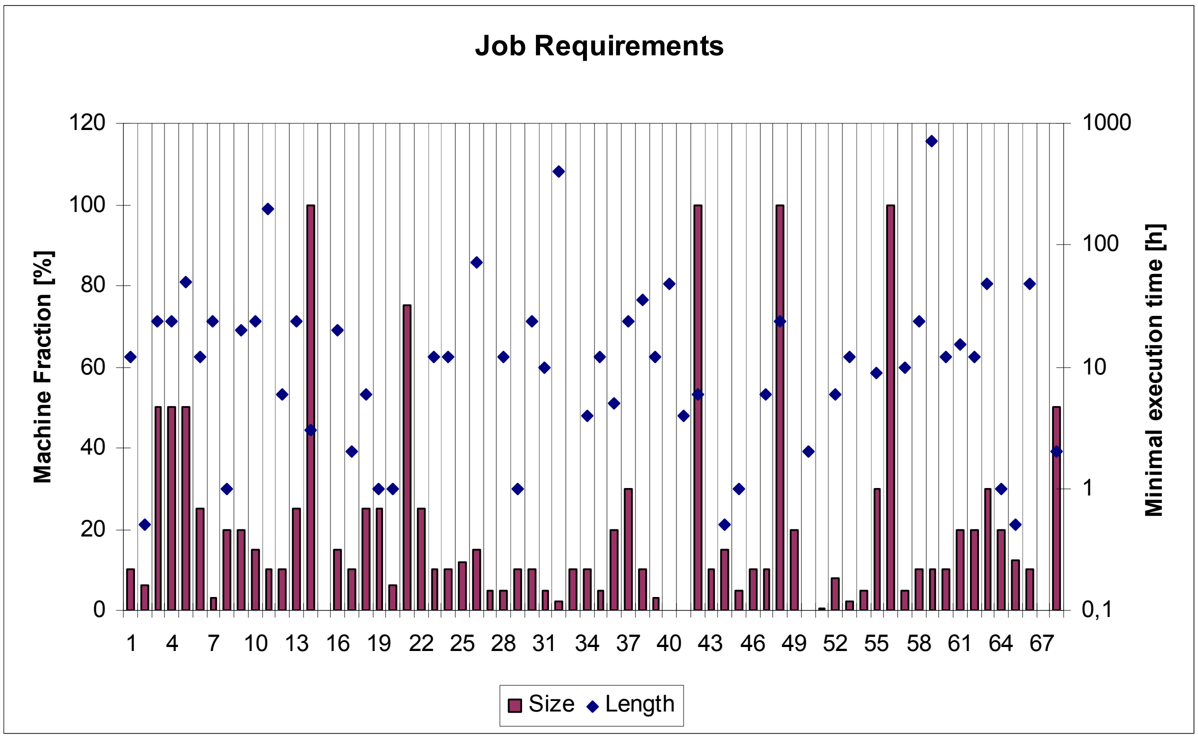


Figure 11. Typical job sizes and their execution lengths.

The time users can wait on their results ranged from *real-time* (3 groups) to *unimportant* (11 groups). The mean tolerated time to solution is 95 days. The tolerated waiting time in the queue ranged from half an hour to 1440 hours; on average the users accepted waiting times of 56 hours.

Shorter times waiting in the queue are also the main motivation for smaller jobs. This can be seen in Table 142 along with the other reasons given why smaller jobs are used.

Reasons for smaller jobs:	
Shorter Queue waiting	27
Bad application scaling	16
Parameter Studies/ Fixed Problemsize	13
Not more allowed	5
Pre-/Post-processing/Testing	5
Limited Result storage resources	2
System Reliability	1

Table 142. Motivations for smaller jobs than the complete machine and the number of groups stating this reason (multiple answers allowed).

The survey also showed that:

- 32 groups use checkpointing to spread their simulations over multiple jobs.
- 28 groups mainly do single big simulations and are striving for high capability computing.
- 22 groups mainly deploy many small jobs exploiting the machine with high capacity computing.
- 13 groups use both usage schemes in their simulations.

These shares are shown in Figure 12.

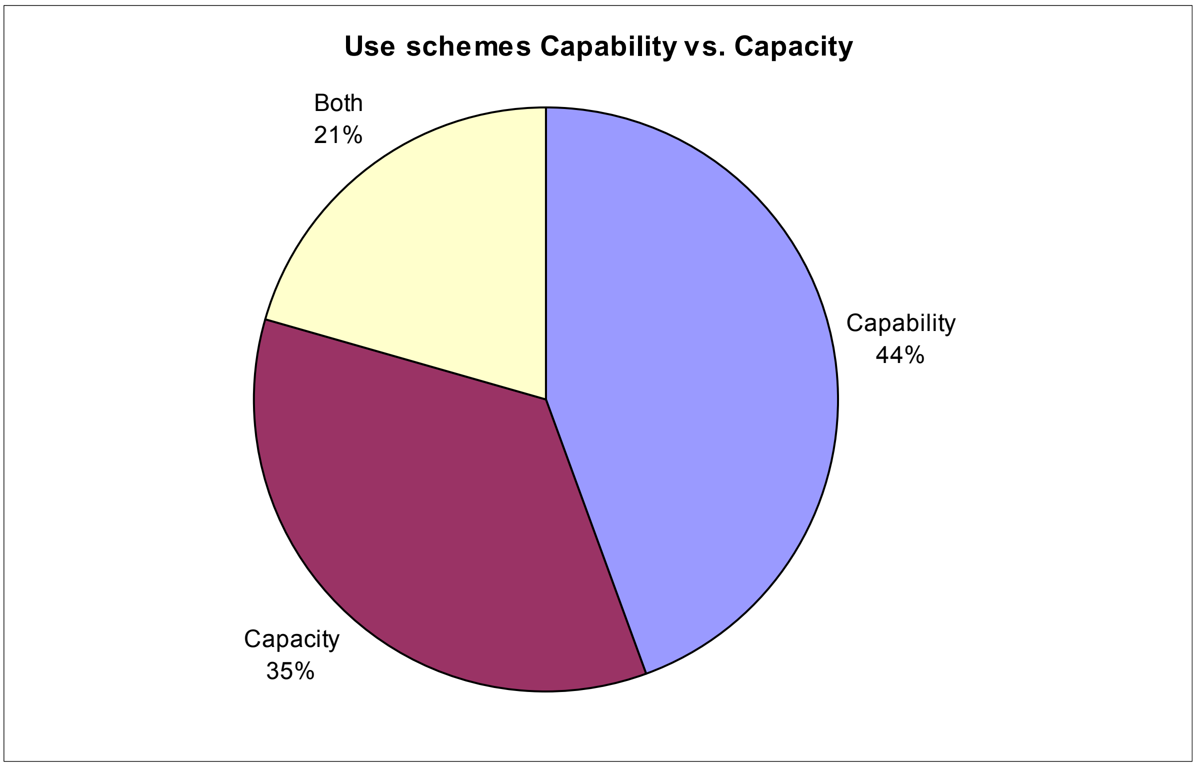


Figure 12. Shares of Capability and Capacity computing.

Further comments we received on the usage patterns are given in Table 143.

Comments:
Development queue for parallel applications on larger number of processors desirable.
Larger Variety of Job-Limits would enable new investigations.
Disk access is becoming a limiting factor.
Storage System should be very reliable.
Only few users per system, for efficient usage.
Powerful networks are crucial.
Need for visualization tools on the HPC system.
Good, comprehensible Scheduler is important.
Interest in coupled systems.
Parallelism on limited single processor performance is hard to use efficiently.
Large storage capacity becomes essential, as transfer is getting infeasible.
Checkpointing by the system instead of the application would be convenient.
Tight integration of massive parallel and thick node systems is desirable.
Large jobs should have highest priorities.
Submitting job chains would be convenient.

Table 143. Further comments regarding the usage of HPC systems.

3.3 Feedback on the desired infrastructure

In this section, we asked the users about the desired infrastructure at the supercomputing center.

The answers to the first question regarding the main memory: *Required memory size for your typical test cases (e.g. 1GB, 10GB, 100GB, 1TB of main memory)?* are shown in Figure 13. They ranged from 1 MB up to 10 TB with a mean of 373 GB. Six groups responded with a required memory space of 1 GB per process.

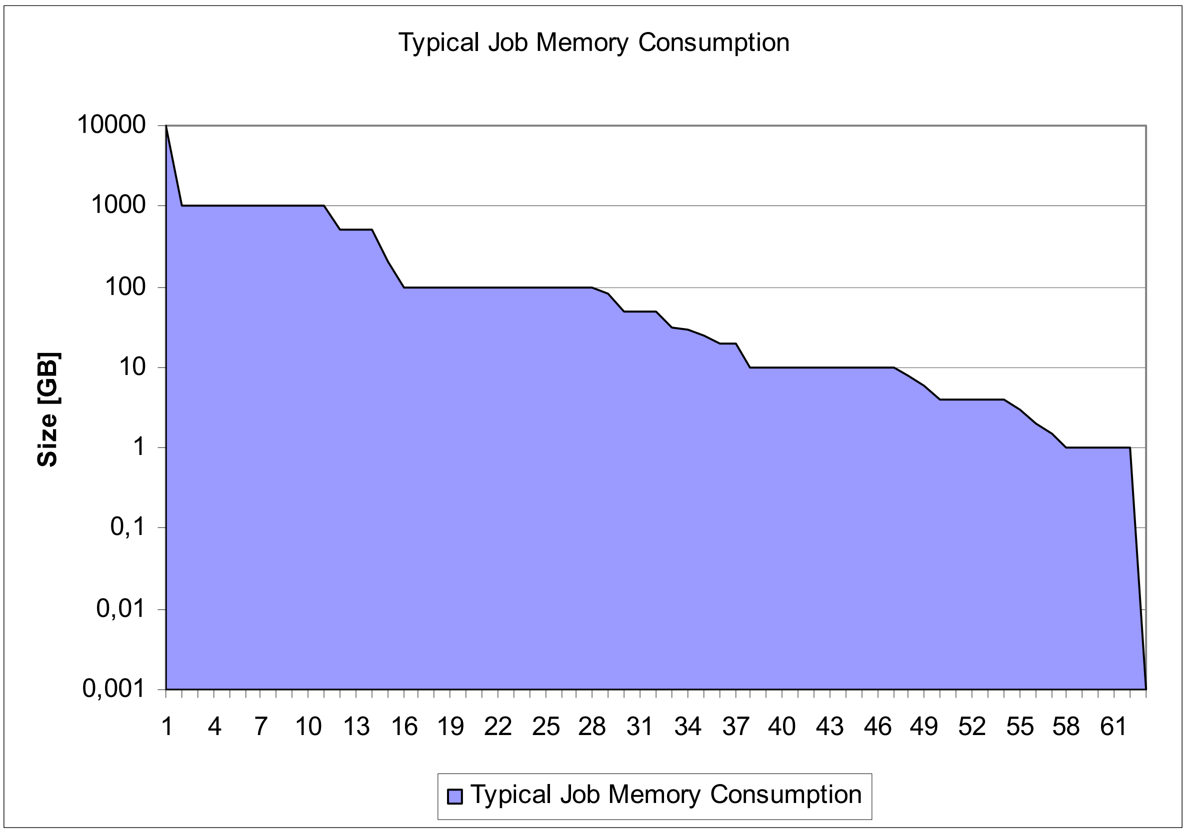


Figure 13. Required memory space by typical jobs. Six responses not captured in this figure, as they stated a size per process of 1 GB.

The next questions were concerned with disk usage. In Figure 14, the required disk space for typical jobs is shown in a colour code indicating the type of access needed. This required disk space ranges from 100 TB (which needs to be accessed immediately by the application throughout the job runtime) to 1 GB (which only needs to be stored away for later retrieval). The mean required disk space is 6 TB.

We then asked if the produced result data has to be transferred back to the home sites of the users. The answers are divided into *Yes*, *Partially*, *No*, *No If* and *Other*, and are shown in Figure 15. The *No If* category covers the answers, where the transfer back is not necessary if certain constraints are matched by the supercomputing centre. The provided prerequisites are listed in Table 144.
There was one answer stating, that a data transfer is necessary to other high performance computing sites. This is covered by the *Other* category.

Transfer back
Ifs (availability at center):
Postprocessing
Large SMP system
Community Data Storage
Free Interactive Queues

Table 144. The conditions for the *No If* category of transferring data back to the home site of the user.

By providing the prerequisites for the conditional keeping of data at the centre, the load on network communication between the centre and the home sites could be reduced, enhancing the usability of the system.

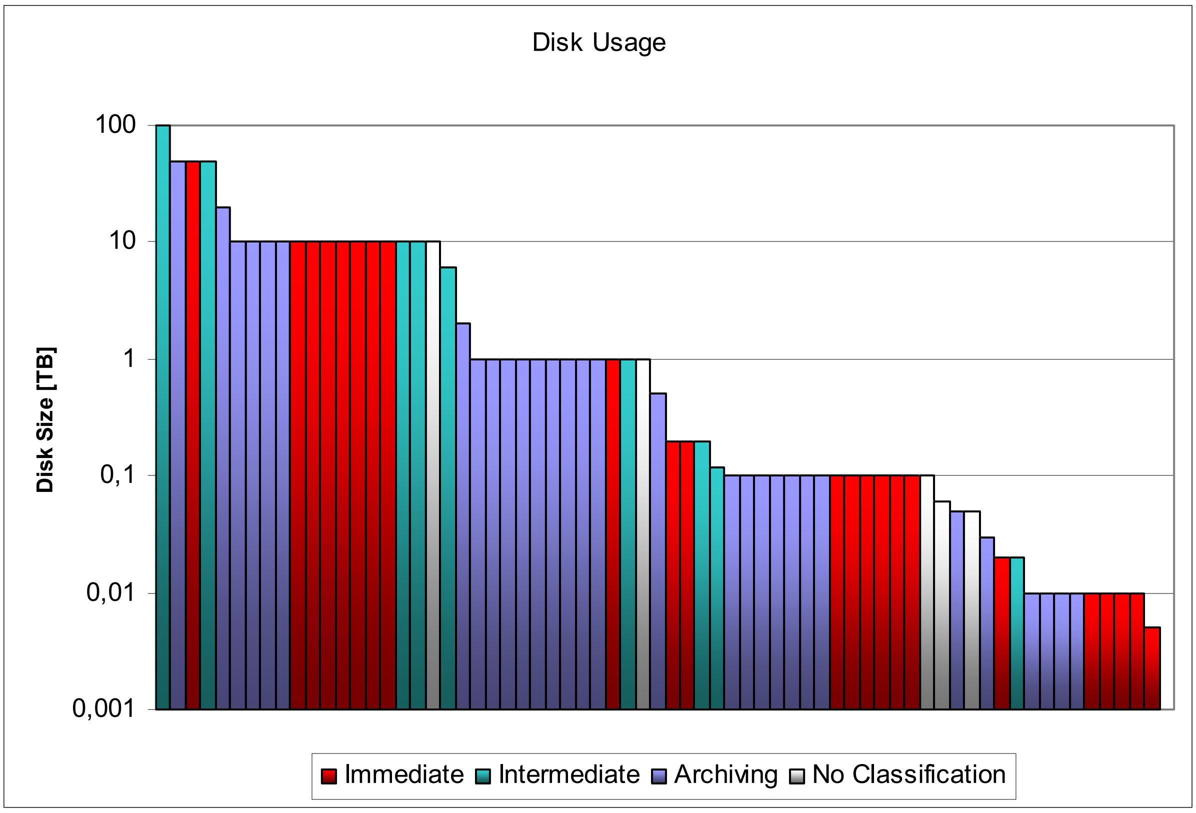


Figure 14. Disk Usage: Amount of needed disk space for typical jobs colour coded with the needed accessibility of the stored data.
Three classes: Immediate – Data is needed by the application during runtime; Archiving – Data just has to be stored away so it can be retrieved later; Intermediate – Some data is needed, or the data is only needed occasionally.

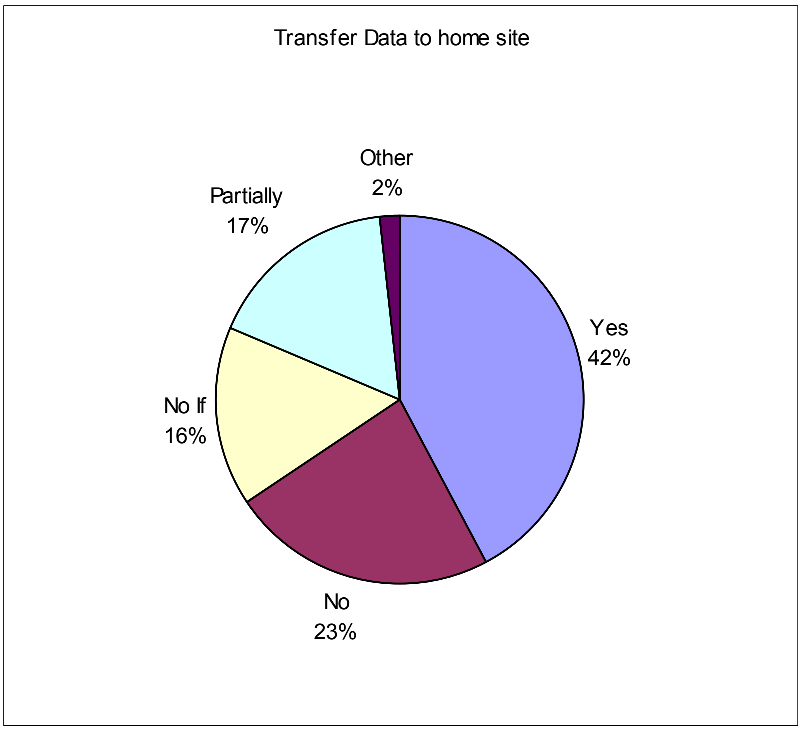


Figure 15. Need for data transfer of produced data back to the home site of the working group, split into 5 categories.
***Yes*, *Partially* and *No* have their obvious meanings. *No If* is covering the answers, where *no* transfer is needed *if* certain constraints are met by the super omputing center and *Other* is regarding to the transfer to other HPC centers.**

Closely related to this transferring back of result data is the question regarding post processing at the HPC center: *Is it possible to do pre-/postprocessing on the HPC-System (e.g. data production, data reduction, visualization)?* The answers are listed in Table 145 divided in the categories *Yes*, *Yes If*, *No* and *Partially*. The *Yes If* category is covering the cases where Pre/Post processing is only possible if some conditions are met by the center.

Postprocessing at center	
Yes	35
Yes If	6
No	16
Partially	11

Reasons for „Yes If“
Ifs (availability at center):
Reliable Storage System
Large SMP system
Interactive Access
Free Interactive Queues

Table 145. Is pre/post processing possible to be done at the HPC center?

When asked about the preferred access method, the majority of users responded with SSH access as their preference. This may be due to the fact, that most groups are themselves developers. Two answers mentioned VNC as a convenient access, and 3 stated that any access method would be fine. The detailed listing of answers to the question *What is your preferred access method (SSH, Unicore, Webinterface, ...)?* is given by Table 146.

Access Method	
SSH	64
Unicore	4
VNC	2
Web	1
Grid	1
Any	3

Table 146. Preferred ways of access to the super computing facility (multiple answers allowed).

Tools that users view as useful on HPC systems are listed in Table 147. Major topics here are Python support, Visualization, development tools and Queueing/Monitoring.

Tools to ease usage
Application aware queueing and data manipulation GUI.
GridFTP
bbFTP
Cryptocard access from anywhere
Mail confirmation after job end
Visualization Tools (like Visit)
Python including SciPy
Matplotlib
Remote mountable filesystems
IDL
Realtime monitoring of job progress
Long term archiving
Matlab/Octave
Remote graphical Interfaces
Scons
Direct Data connection
Visual debugger
Good profiling tools
NCO tools
Simple queuing system
nxclient
LatFor data grid tools

Table 147. Additional useful tools.

Further comments we received on the topic of HPC infrastructure are listed in Table 148.

Comments on the Infrastructure
Good user support is important.
Experiments (5-10% of production) are crucial to prepare production, even so they mostly fail.
HPC systems are a great resource, that give scientists a competitive edge.
SAFE is good for managing resources.
Bandwidth from HPC-System to user has to be increased.
Appropriate Hardware for post processing important, because data transfer is increasingly difficult.
Provide enough storage space for the results of large simulations.
Courses on current programming paradigms and analysis of efficiency.

Table 148. Comments regarding the HPC infrastructure.

3.4 Future algorithms

We asked the users about what they are planning to do in the future, and what new kind of numerical algorithms they view as upcoming techniques in their field of research. The answers are summarized and sorted by scientific field in the following subsections.

3.4.1 *Computational Chemistry (total 16 working groups)*

- (5 groups) Improvement in parallel linear algebra algorithms, matrix inversion and diagonalization
- (4 groups) No changes planned. Some use third party codes
- (3 groups) Application specific algorithms (Hybrid variation-perturbation, reducing computational complexity, SPME approaches)
- (3 groups) Parallelization Improvements

- Other answers were: implementing multigrid methods, reduce computational complexity, architecture driven tunings, searching and sorting, FFTs

3.4.2 *Computational Fluid Dynamics (total 12 working groups)*

- (4 groups) Dynamic load balancing
- (3 groups) Adaptive meshing methods or multiblock methods
- (3 groups) Application specific changes (Implementation of Spectral methods, Semi-implicit projection schemes)
- (2 groups) Higher order methods
- (1 group) No changes planned

- Other answers were: Hybrid parallelization (MPI+OpenMP)

3.4.3 *Condensed Matter Physics (total 12 working groups)*

- (5 groups) No changes planned
- (4 groups) Application specific changes — Hybrid electrodynamics, realspace finite difference methods, quantum cluster approaches, continuous time approach in QMC solver, algorithms targetting the relevant effective potential,
- (2 groups) Matrix diagonalization algorithms

- Other answers were: FFT optimization and hybrid parallelization (MPI+OpenMP)

3.4.4 *Astronomy and Cosmology (total 7 working groups)*

- (6 groups) Application specific changes (staggered grid MHD, treecodes with active messaging, Piecewise Parabolic Method; HLLE scheme, implicit methods, spectral algorithms)
- (2 groups) Higher order methods
- (1 group) No changes planned

3.4.5 *Life Sciences (total 6 working groups)*

- (2 groups) Application specific changes — computational protein design, coarse grained MD with UNRES force field

- (2 groups) No changes planned

- Other answers were: Parallelization improvements, multigrid methods

3.4.6 *Particle Physics (total 6 working groups)*

- (4 group) Hybrid Monte Carlo, sparse matrix algebra, deflation, matrix inversion and diagonalization
- (1 group) Application specific changes (Including heavier quark determinant)
- (1 group) Parallelization improvements via domain decomposition

3.4.7 *Earth and Climate Sciences (total 3 working groups)*

- (1 group) Application specific changes (Ploy-mesh LES)
- (1 group) Dense linear algebra with Lapack
- (1 group) No changes planned

3.4.8 *Other scientific fields*

Economics (2 groups): No changes planned

Computational Quantum Mechanics (1 group): parallel linear algebra algorithms and Monte Carlo

Theoretical Physics (1 group): Application specific changes (DVR basis sets)

Nanooptics (1 group): Application specific changes (3d photonic crystal solver, surface plasmon packages, ftdt code for split rings)

Plasma Physics (1 group): No changes planned

3.4.9 *Combined Overview*

Taking together the responses from the various scientific fields, we find:

- 22 groups plan to implement application specific algorithm improvements – please refer to previous paragraphs for details
- 17 groups plan no changes to the present codes
- 13 groups plan improvements in the parallel linear algebra (sparse & dense) methods – mainly matrix diagonalization algorithms & CG methods
- groups mostly CFD users plan to implement dynamic load balancing and adaptive meshing methods
- groups plan to implement higher order methods
- groups plan to implement better parallelization strategies (mostly MPI+OpenMP)
- groups intend to improve Hybrid Monte Carlo method
- 2 groups intend to implement Multigrid methods
- 2 groups intend to improve FFT performance

4 Summary and Future Work

This deliverable forms a final report on the requirements of applications for the PRACE petascale machines to be installed in 2010/11. The report consists of two sections: an analysis of the behaviour of a set of applications on current hardware platforms, and a survey of key users

In Chapter 2 we have presented an analysis of the characteristics and requirements of (most of) the PRACE application set that was identified and selected in Task 6.1. It builds on the work presented in Deliverable D6.2.1 in two important ways: the set of applications considered here is the relevant set for the PRACE project, and the analysis is based on actual measured data rather than on users' opinions.

The analysis presented here is in some ways preliminary, as it forms the start of the process of porting, optimising and petascaling the applications: these activities are the principal responsibilities of the remaining activities in WP6. In Task 6.3, the applications will be ported to a range of different architectures. Task 6.4 will be concerned with optimising the single CPU performance of the applications, while Task 6.5 will focus on making the applications scale to larger numbers of processing cores.

Ideally, we would have liked to analyse all the application on the same architecture, as this would make comparisons between them more meaningful. However, in order to carry out the analysis in a short time scale, it has not been possible to do this. As part of the process, one application (NAMD) has been analysed on two different architectures, and the results are similar enough to give us some confidence that the analyses are reasonably architecture independent. One of the goals of Task 6.3 is to port all the applications to one common architecture. This will allow a more direct comparison of the characteristics of the application set to be made.

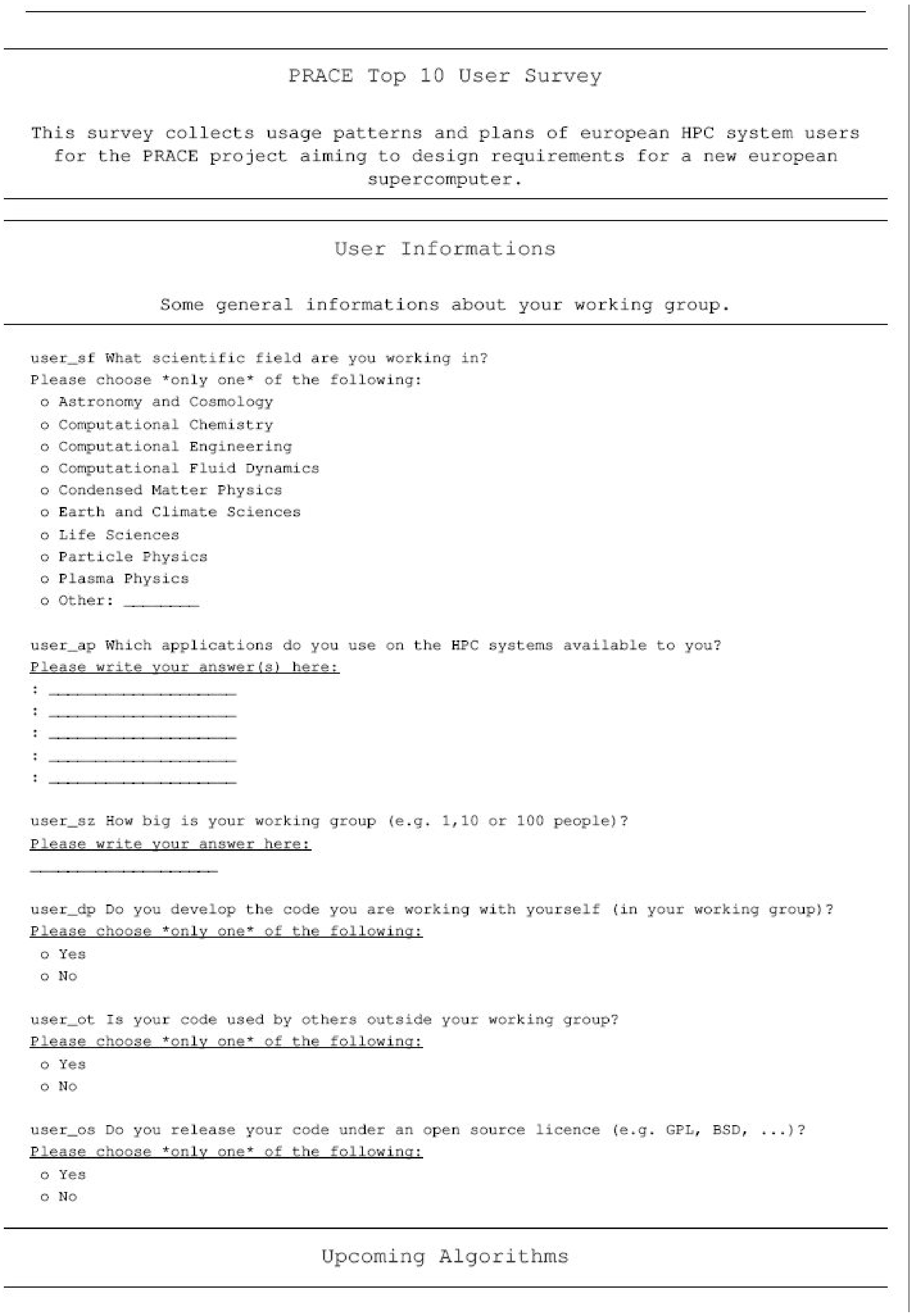
Another restriction imposed by the time constraint is that, for many of the applications, it was not possible to source or create input datasets which are ideal for benchmarking on petascale systems. This is clear from the analysis of the scalability of the applications: many do not achieve a speedup close to a factor of two between 256 and 512 (or even between 128 and 256 cores). Obtaining or creating larger datasets suitable for large scale machines will form one of the key tasks in the remainder of WP6. Despite the incompleteness of the data, we have been able to identify applications which could benefit from faster CPUs, improved memory subsystems or better interconnects.

From the applications included in this deliverable there are a number of general conclusions that can be made. First, serial optimisation is key to a few applications, such as HELIUM, NAMD, GADGET, and GPAW. The next task in WP6 will deal with these optimisations. Second, communications are another key area in which scaling can be improved. Many applications, such as CPMD, require optimisation to the All-To-All MPI communications if they are to scale to petaflop/s systems. Thirdly, load imbalance may be a problem for some codes, although at this stage further investigation is required to determine likely causes. Finally, the main problem facing petascaling of these applications is access to large enough datasets. However, significant progress is being made on this within WP6.

In Chapter 3, we have presented the results of a user survey, intended to complement the survey of systems and applications presented in D6.1.
We asked the Top 10 users from each PRACE partner a series of questions about their current and planned usage of large scale HPC system, and for their comments on future petascale systems. We obtained responses from 68 users, which have been collated and analysed. The results provide useful information about topics that include job sizes and lengths, memory requirements, the need for data transfer off the HPC systems, preferred access methods and requirements for supporting software packages.

The results of this study, together with D6.1 and D6.2.1, will provide WP7 (which has the responsibility of defining requirements, evaluation criteria and procurement processes for the petascale machines to be installed in 2010/11) with an up-to-date view of the requirements of current applications and the likely trends in those requirements on the relevant timescales. By using this information, WP7 will be able to base its work on the solid foundations of carefully acquired and analysed data from real applications, systems and users.

Appendix A: User Survey



PRACE Top 10 User Survey

This survey collects usage patterns and plans of european HPC system users for the PRACE project aiming to design requirements for a new european supercomputer.

User Informations

Some general informations about your working group.

user_sf What scientific field are you working in?
Please choose *only one* of the following:

o Astronomy and Cosmology
o Computational Chemistry
o Computational Engineering
o Computational Fluid Dynamics
o Condensed Matter Physics
o Earth and Climate Sciences
o Life Sciences
o Particle Physics
o Plasma Physics
o Other: ________

user_ap Which applications do you use on the HPC systems available to you?
Please write your answer(s) here:

: ____________________
: ____________________
: ____________________
: ____________________
: ____________________

user_sz How big is your working group (e.g. 1,10 or 100 people)?
Please write your answer here:

____

user_dp Do you develop the code you are working with yourself (in your working group)?
Please choose *only one* of the following:

o Yes
o No

user_ot Is your code used by others outside your working group?
Please choose *only one* of the following:

o Yes
o No

user_os Do you release your code under an open source licence (e.g. GPL, BSD, ...)?
Please choose *only one* of the following:

o Yes
o No

Upcoming Algorithms

UA Please explain the algorithms you are planning to deploy in the near future. If they differ from the current ones, state the differences.
Please write your answer here:

____
____
____
____
____
____
____
____
____

Usage Patterns

Please explain your usage of the HPC-Systems in the following questions.

UP_size What fraction of the machine does a job typically consume (how much computational power does a single job require; e.g. 25% of the number of available processors on Machine X)?
Please write your answer here:

____

UP_reas If you don't use the largest possible one, what is your reason to run smaller jobs (e.g.: no greater power for a single run needed, shorter queue waiting times, ...)?
Please write your answer here:

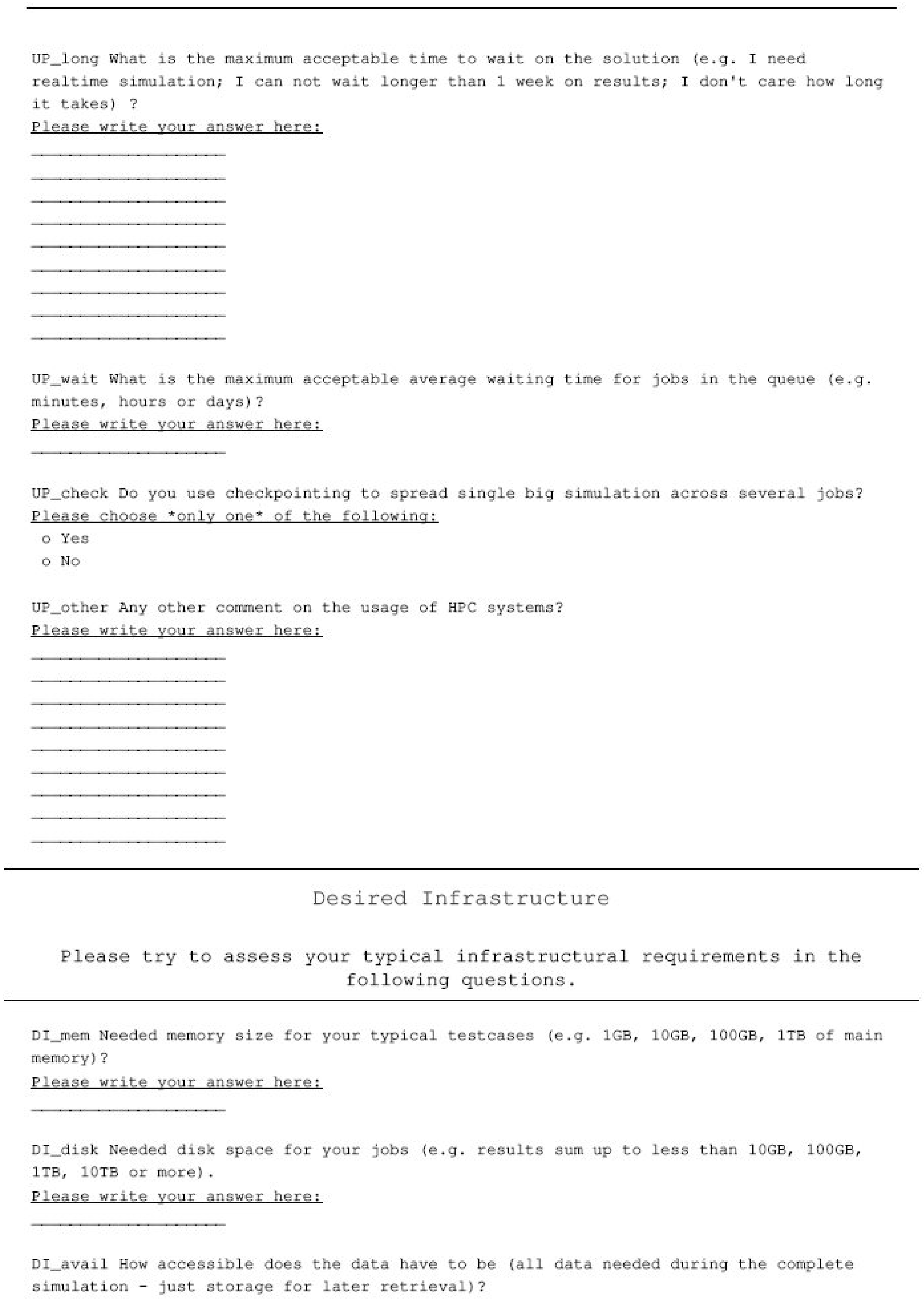
____
____
____
____
____
____
____
____
____

UP_num Do you tend to do single big simulation or rather many smaller simulations in a parameter study?
Please write your answer here:

____
____
____
____
____
____
____
____
____

UP_time How long has a job to be allowed to run in a minimum (e.g. jobs have to run at least 6 hours without restart)?
Please write your answer here:

____



UP_long What is the maximum acceptable time to wait on the solution (e.g. I need realtime simulation; I can not wait longer than 1 week on results; I don't care how long it takes) ?

Please write your answer here:

____
____
____
____
____
____
____
____
____

UP_wait What is the maximum acceptable average waiting time for jobs in the queue (e.g. minutes, hours or days)?

Please write your answer here:

____

UP_check Do you use checkpointing to spread single big simulation across several jobs?

Please choose *only one* of the following:

o Yes
o No

UP_other Any other comment on the usage of HPC systems?

Please write your answer here:

____
____
____
____
____
____
____
____
____

Desired Infrastructure

Please try to assess your typical infrastructural requirements in the following questions.

DI_mem Needed memory size for your typical testcases (e.g. 1GB, 10GB, 100GB, 1TB of main memory)?

Please write your answer here:

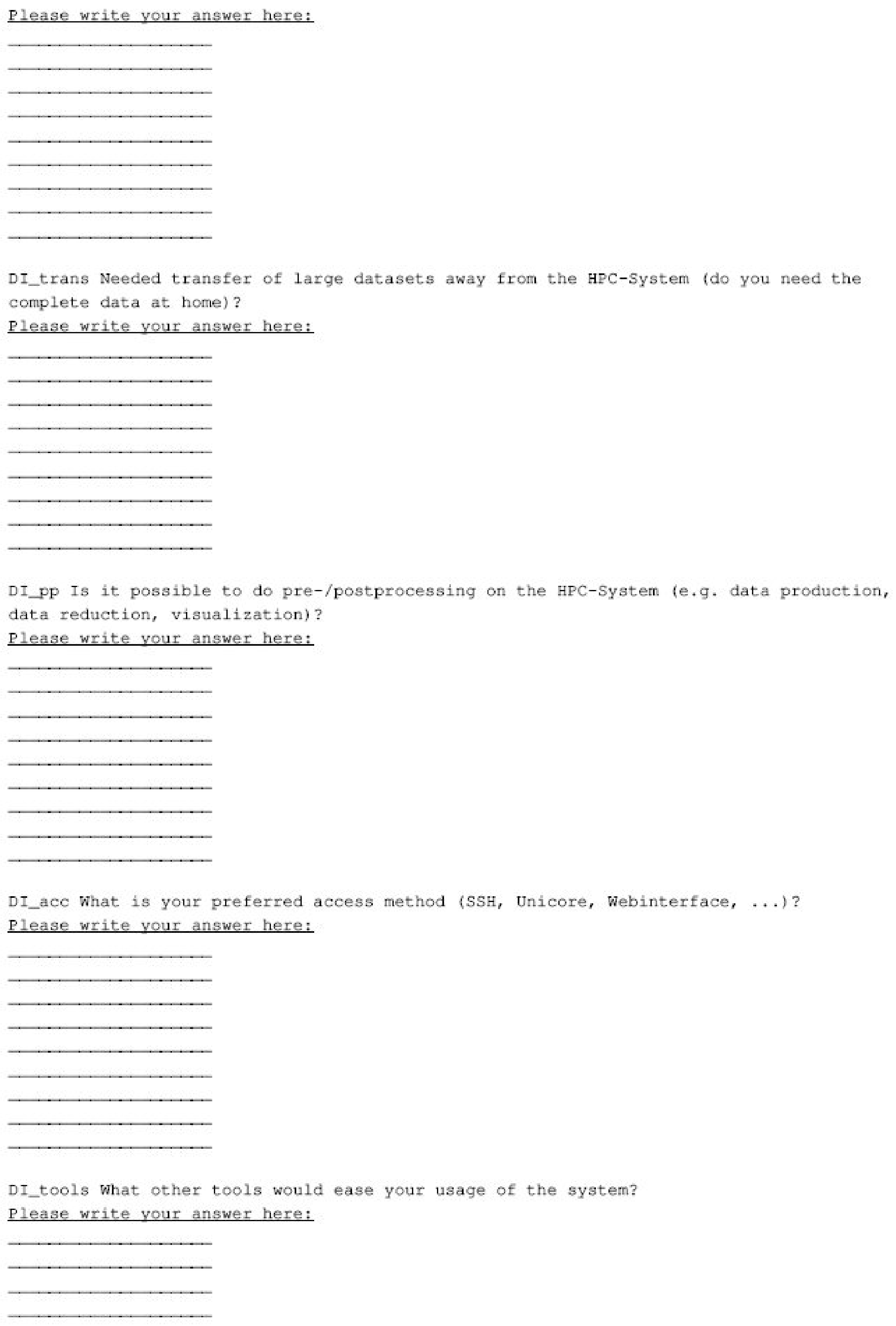
____

DI_disk Needed disk space for your jobs (e.g. results sum up to less than 10GB, 100GB, 1TB, 10TB or more).

Please write your answer here:

____

DI_avail How accessible does the data have to be (all data needed during the complete simulation - just storage for later retrieval)?



Please write your answer here:

____

____

____

____

____

____

____

____

____

DI_trans Needed transfer of large datasets away from the HPC-System (do you need the complete data at home)?

Please write your answer here:

____

____

____

____

____

____

____

____

____

DI_pp Is it possible to do pre-/postprocessing on the HPC-System (e.g. data production, data reduction, visualization)?

Please write your answer here:

____

____

____

____

____

____

____

____

____

DI_acc What is your preferred access method (SSH, Unicore, Webinterface, ...)?

Please write your answer here:

____

____

____

____

____

____

____

____

____

DI_tools What other tools would ease your usage of the system?

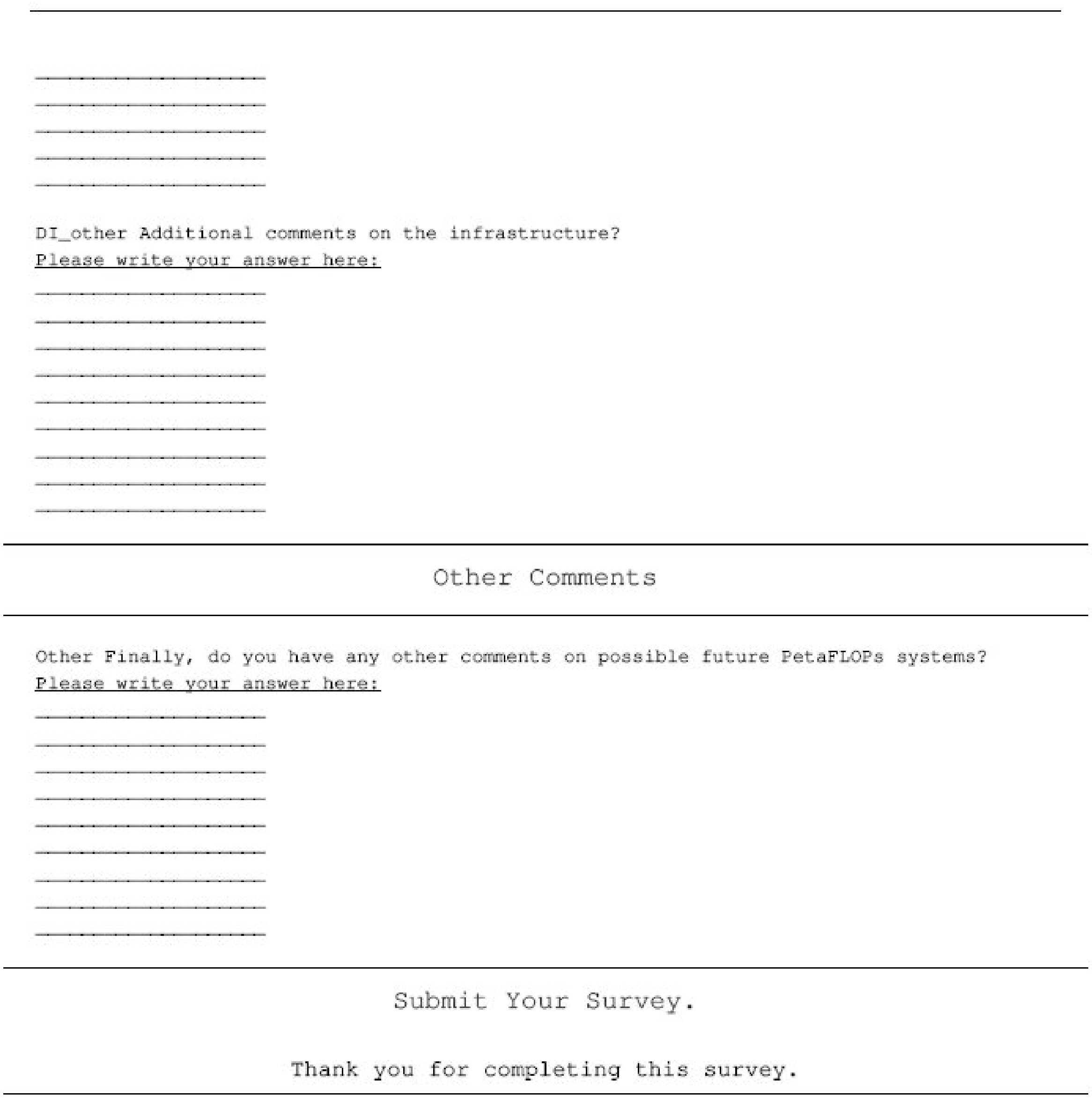
Please write your answer here:

____

____

____

____



____
____
____
____
____

DI_other Additional comments on the infrastructure?
Please write your answer here:

____
____
____
____
____
____
____
____
____

Other Comments

Other Finally, do you have any other comments on possible future PetaFLOPs systems?
Please write your answer here:

____
____
____
____
____
____
____
____
____

Submit Your Survey.

Thank you for completing this survey.