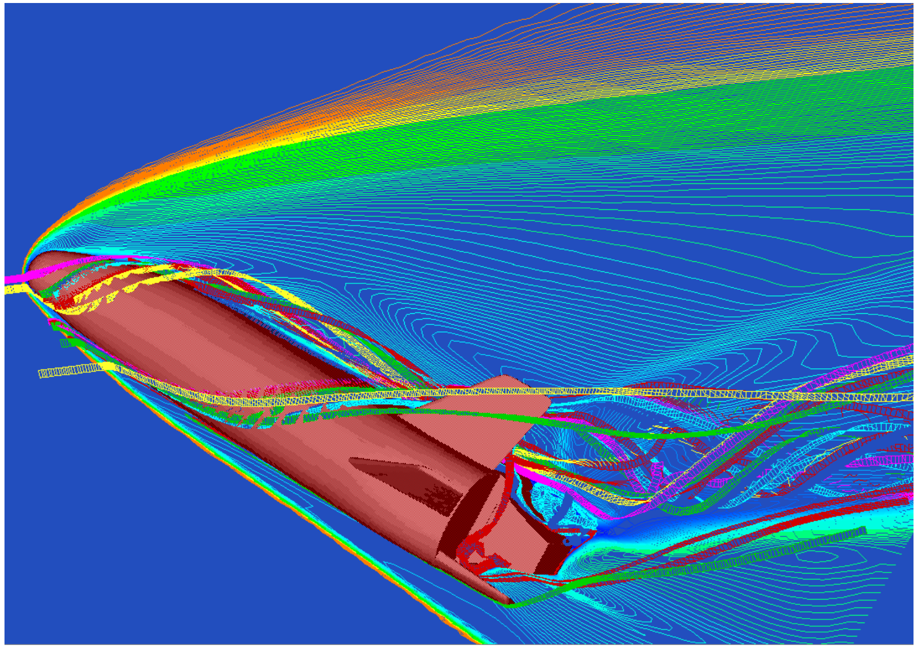
©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

NASA Panel Java Soundbytes



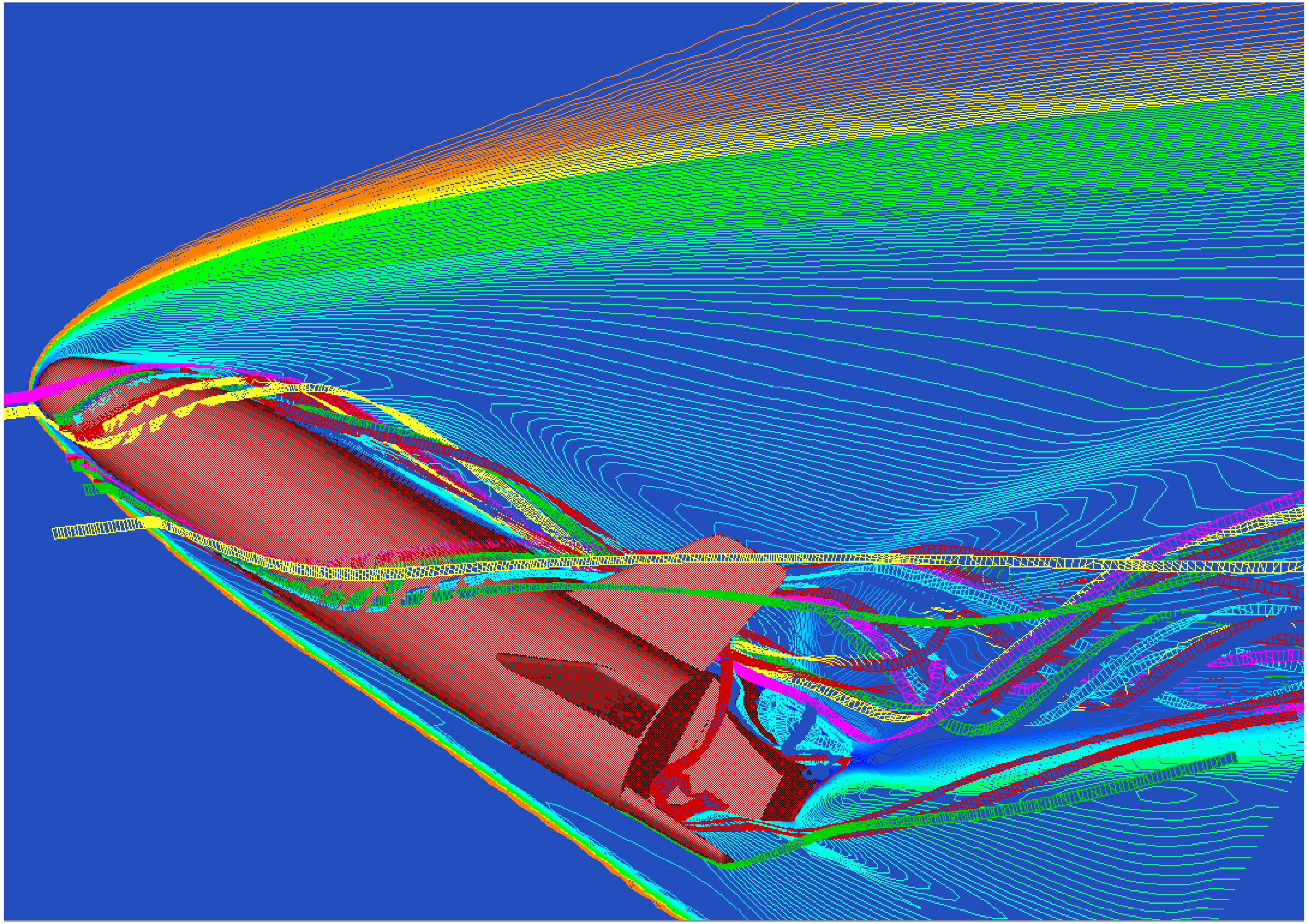
Jochem Häuser, Thorsten Ludewig, Roy D. Williams, Ralf Winkelmann,
Torsten Gollnick, Sharon Brunett, Jean Muylaert

presented at

5th National Symposium
on Large-Scale Analysis, Design and Intelligent Synthesis Environments

Williamsburg, VA, October 12th to 15th, 1999

Java in Science and Engineering



Java for Scientific and Engineering Applications

- Object oriented formulation
- Threads for parallel computing (no additional software libraries, like MPI or PVM, are needed)
- Distributed computing on networks supported within the language (remote objects, implemented by RMI)
- Database access through JDBC
- Graphics packages for GUIs and visualization of scientific data (Swing and Java3D)
- Robustness, portability, maintainability, reusability, productivity, distributed project management, exception handling

©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

JParFw (Java Parallel Framework)

JparFw is pure Java template code for concurrent scientific and engineering computation on arbitrary solution domains.

The code consists of 3 parts:

- client (User interface, geometry data, solver module)
- share (interface between client and server through remote objects, RMI)
- server (parallel framework, geometry framework, generic solver)

JParFw (Java Parallel Framework)

Parallelism is achieved by decomposing the solution domain into subdomains and solving the particular equations, i.e. Navier-Stokes Eqs. within each subdomain.

Each subdomain is implemented by a thread.

Communication between subdomains (threads) is via shared memory.

Load imbalance created by advanced numerical schemes, for instance, GMRES, is optimized by the automatic thread scheduling of the OS and not by the user.

©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

Parallel Square Root Program

multiple threads are used to implement parallel structure

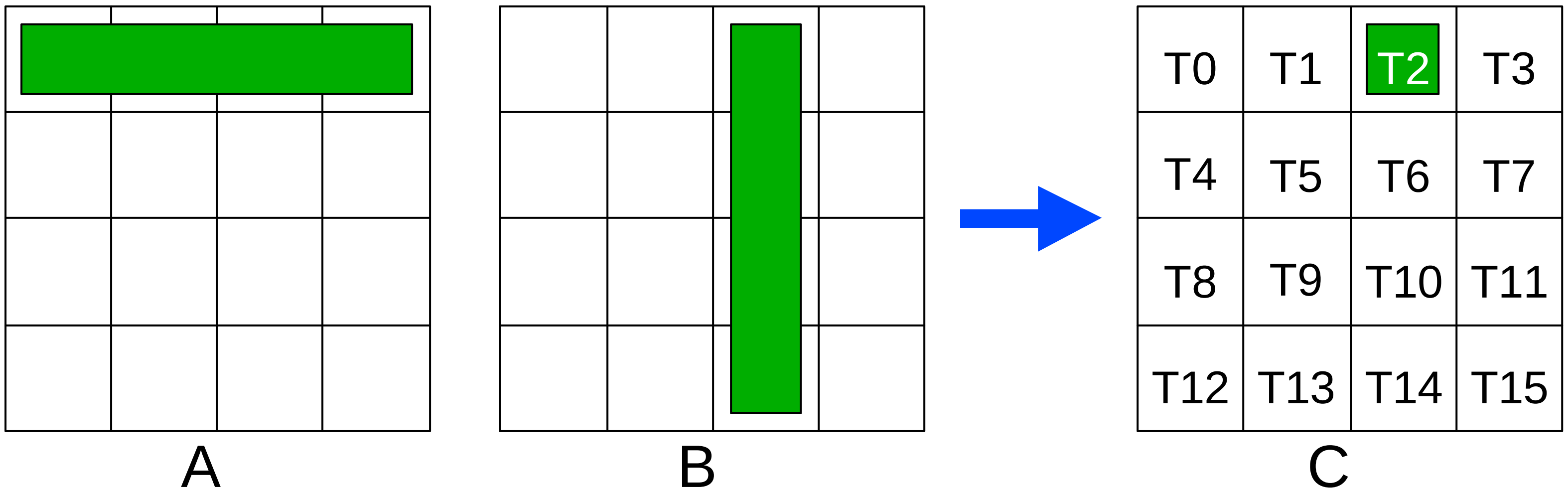
Parallel speedup on HP V-Class machine, Caltech

number of threads	time in s	parallel speedup
1	12:41	1
2	6:34	1,93
3	4:20	2,92
4	3:17	3,86
8	1:39	7,69
9	1:28	8,65
16	0:50	15,22
32	**0:26**	**29,27**

©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

Matrix Multiply

Parallel matrix multiplication is implemented by block matrices, as shown in the figure matrices **A** and **B** are multiplied to compute matrix **C**



The multi-threaded matrix multiplication is performed by splitting matrix C into partitions. Each partition is then calculated by one thread, with the thread numbering as shown for matrix C. Concurrent access to the memory containing A and B is necessary: here we see the memory that thread 2 accesses.

©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

Sequential Matrix Multiplication

Hardware and Software specifications	MFlops per second for dirfferent matrix size		
	30x30	100x100	300x300
HP Vclass, C-code	**242,00**	237,00	114,00
HP Vclass, Java 1.1.7	**9,33**	9,57	9,54
Sun E450, C-code	176,86	157,73	35,24
Sun E450, Java 1.1.7	**6,35**	6,72	5,87
Sun E450, Java 1.2	**17,08**	12,65	8,90
Pentium, C-Code	**90,00**	91,74	39,82
Pentium, Java IBM 1.1.6	**24,80**	22,79	11,21

The performance, in megaflops, of the sequential matrix-multiply program on the one processor of the HP-Vclass, one processor of the Sun E450, and a Pentium II PC running Linux.

©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

Multi-threaded Matrix Multiplication

Megaflop rates for the pure Java multithreaded matrix-multiply benchmark.

number of threads	HP using 16 CPUs		HP using 1 CPU		Sun using 4 CPUs	
	30x30	300x300	30x30	300x300	30x30	300x300
1	7,01	8,64	6,51	8,66	13,40	9,06
4	11,38	33,49	3,86	8,68	19,21	23,56
9	6,33	72,53	2,40	8,73	12,25	22,00
16		**118,68**		8,69		28,13
25	2,62	112,97	1,04	8,65	5,14	27,84
36	1,83	110,66	0,75	8,64	3,75	30,07
100	0,64	109,53	0,29	8,44	1,57	33,93

On the HP architecture a maximal speedup of 13.74 using 16 processors for the 300x300 matrix example was measured.

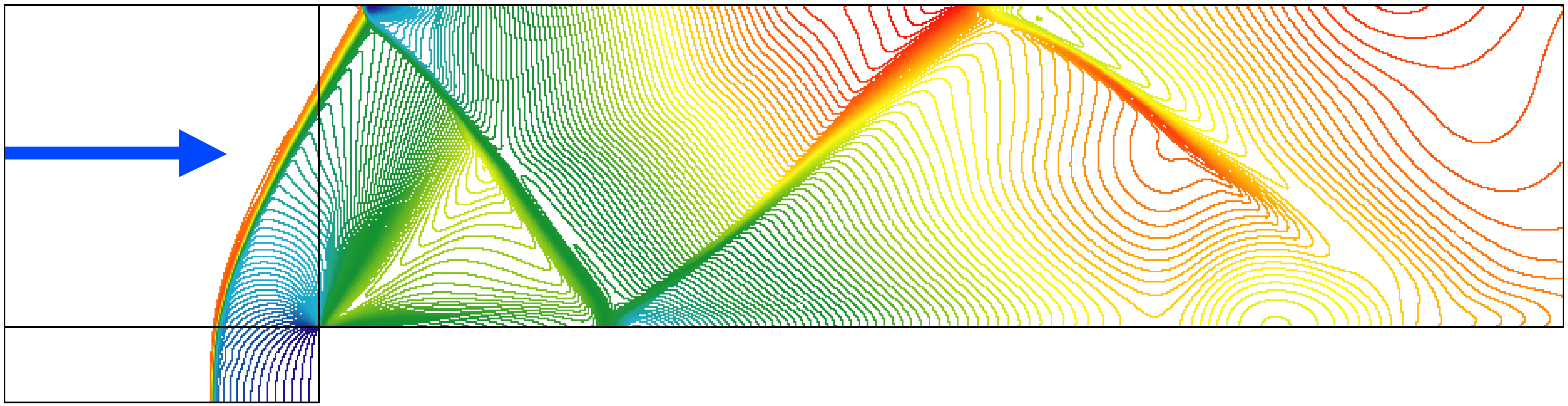
©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

JParEuler

Euler solver plugin for the JParFw

Test case is a Mach 3 Euler flow past a forward facing-step

The simulations are based on structured multi-block grids



The computation is explicit and first order accurate. Shown is the Mach-number distribution.

©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

JParEuler

number of blocks	number of cells	time in seconds		parallel speedup
		single processor	multi processor (16)	
16	121104	3246,73	541,13	6,00
16	200704	6908,88	1077,20	6,41
16	484416	12905,88	2720,48	4,74
48	**118803**	**2980,93**	**225,76**	**13,20**
48	202800	5190,54	436,09	11,90
48	480000	12663,30	1162,54	10,89

HP V-Class execution times for 320 iterations
(forward facing step)

Conclusions

- Thread concept delivers full parallel efficiency for a sufficient number of threads and sufficient computational work within a thread can be provided.
- Scientific and engineering problems generally satisfy these requirements.
- Substantial performance improvements in the execution speed of java programs can be expected with the release of new compiler versions.
- Therefore, research should concentrate on demonstrating parallel and numerical scalability as well as producing a generic parallel Java solver.

©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

Conclusions

Distributed computing and distributed project management for a large number of numerical problems in science and engineering can be successfully addressed by the combination of OOP, Thread and Client-Server approach.

Further work will be needed, but we follow Kernighan's rules *Make it right before you make it faster* as well *Don't patch bad code, rewrite it*, the latter rule being the reason for a pure Java flow solver.

©1999 University of Applied Sciences, Braunschweig/Wolfenbuettel and CLE, Department of High Performance Computing

Acknowledgments

The *Test Suite for High-Performance Parallel Java* is part of the *JavaPar* project which is partly funded by the ministry of Sciences and Culture of the State of Lower Saxony, Germany and the European Commission under contract *JavaPar* 1997.262.

References

Eiseman, Peter R., GridPro v4.1, The CFD Link to Design, Topology Input Language Manual, Program Development Corporation Inc., 300 Hamilton Ave., Suite 409, White Plains, NY 10601, 1999.

Fox, G.C. (ed.), Java for Computational Science and Engineering- Simulation and Modeling I, Concurrency Practice and Experience, Vol. 9(11), June 1997,Wiley.

Fox, G.C. (ed.), Java for Computational Science and Engineering- Simulation and Modeling II, Concurrency Practice and Experience, Vol. 9(11), November 1997,Wiley.

James Gosling, Henry McGilton, The Java Language Environment - A White Paper, Sun Microsystems October 1995, http://www.javasoft.com/docs/.

Häuser J., Williams R.D, Spel M., Muylaert J., ParNSS: An Efficient Parallel Navier-Stokes Solver for Complex Geometries, AIAA 94-2263, AIAA 25th Fluid Dynamics Conference, Colorado Springs, June 1994.

Häuser, J., Xia, Y., Muylaert, J., Spel, M., Structured Surface Definition and Grid Generation for Complex Aerospace Configurations, In: Proceedings of the 13th AIAA Computational Fluid Dynamics Conference Open Forum, June 29 - July 2, 1997, Part 2, pp. 836-837, ISBN 1-56347-233-3.

Häuser J., Williams R.D., Strategies for Parallelizing a Navier-Stokes Code on the Intel Touchstone Machines, Int. Journal for Numerical Methods in Fluids 15,51-58. , John Wiley & Sons, June 1992.

Häuser, J., Ludewig, Th., Gollnick, T., Winkelmann, R., Williams, R., D., Muylaert, J., Spel, M., A Pure Java Parallel Flow Solver, 37th AIAA Aerospace Sciences Meeting and Exhibit, AIAA 99-0549 Reno, NV, USA, 11-14 January 1999.

Winkelmann, R., Häuser J., Williams R.D, Strategies for Parallel and Numerical Scalability of Large CFD Codes, Comput. Methods Appl. Mech. Engrg. 174 (1999) 433-456, 1999.

our preferred textbook for Java: *CoreJava* by C. Horstmann et al., Sunsoft Press, Prentice Hall,1999.