CUG SUMMIT 2002

Implementation of an integrated efficient parallel multiblock Flow solver

Thomas Bönisch, Panagiotis Adamidis and Roland Rühle
adamidis@hlrs.de

Outline

- Introduction to URANUS
- Why using Multiblock meshes
- Problems and Solutions when using Multiblock meshes
- Results
- Outlook

Re-entry Simulation - X38 (Prototype of CRV)

United States

Sequential 2D/3D URANUS (Non Equilibrium Flows)

- Cell-center oriented finite volume approach
- solving the unsteady, compressible Navier-Stokes equations
- the implicit equation system is solved iteratively by Newton’s method
- two different limiters for second order accuracy

- CVCV multiple temperature gas phase model
- Chapman-Cowling transport coefficients models
- Gaskinetic gas-surface model with different catalysis models
- PARADE/HERTA gas-radiation coupling

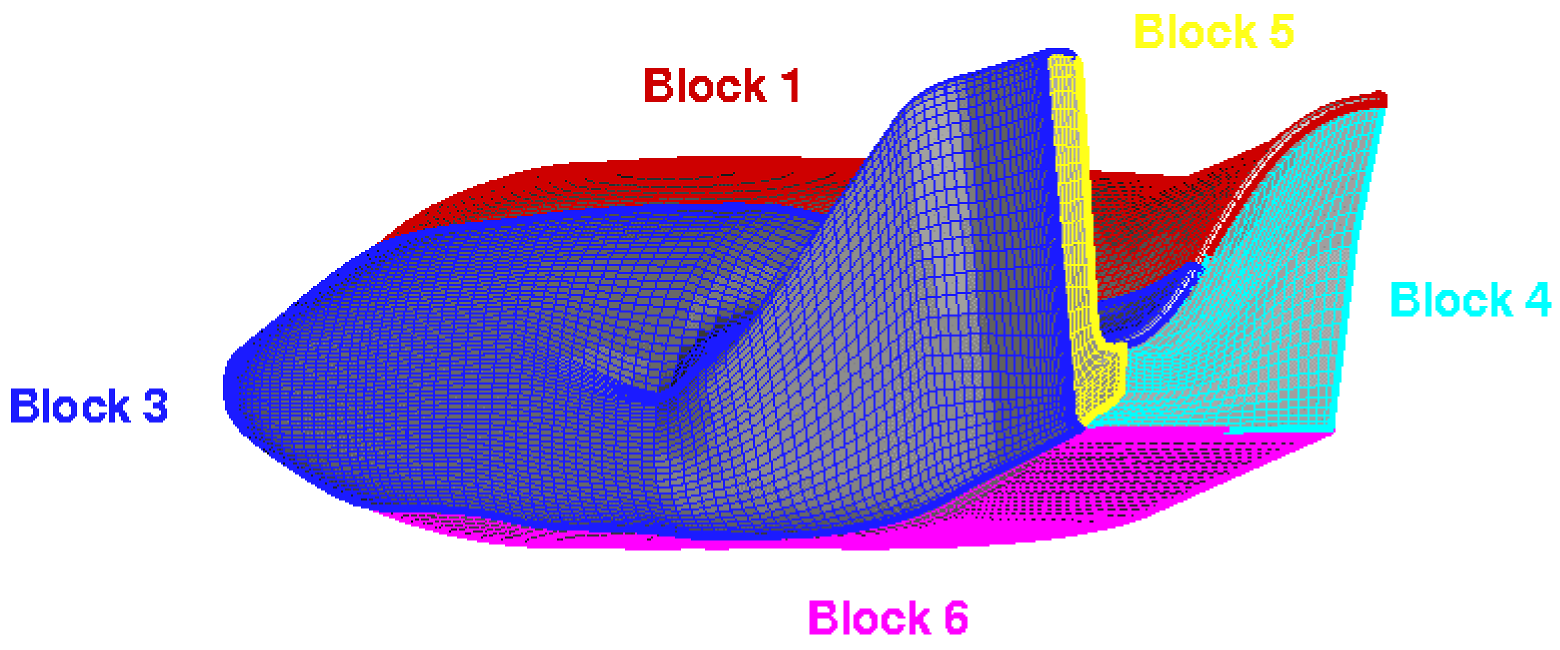
Parallelization

- domain decomposition
  - with two halo cells at the subdomain boundaries
- dynamic data structures using Fortran90
- special solver
- execution model SPMD
- message-passing with MPI
- still working only on C-meshes

Why Using Multiblock Meshes

- There are topologies which cannot be meshed or which are hard to mesh with a C-mesh
- Singularity and sometimes heavily distorted mesh cells are limiting the convergence rate
- using unstructured meshes would result in rewriting the code
- to obtain performance on current Supercomputers is easier with structured meshes
- ➡ using multiblock meshes:
  - meshing of complex topologies is possible
  - structured blocks
  - Performance easier to obtain

A Multiblock Mesh for X-38



Characteristics of Multiblock Meshes

- Each block may have a local coordinate system which is different from that of its neighbours
- A block may have one, two or more neighbours on one block side
- Physical boundaries may occur on each blockside
- Blocks have generally different sizes

$\longrightarrow$ the program must be able to handle all this

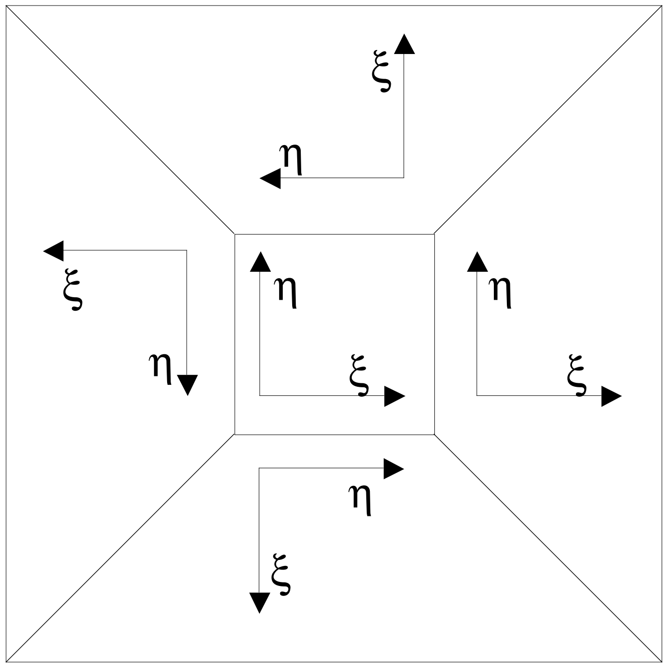
Extensions Necessary to Handle Multiblock Meshes

- Handling of block internal orientation
- Handling of more complex neighbour dependencies
- Handling of physical boundaries at each block side

- Load balancing
- handling of multiple blocks on one processor
- automatic block splitting
- using of a load balancer for block distribution

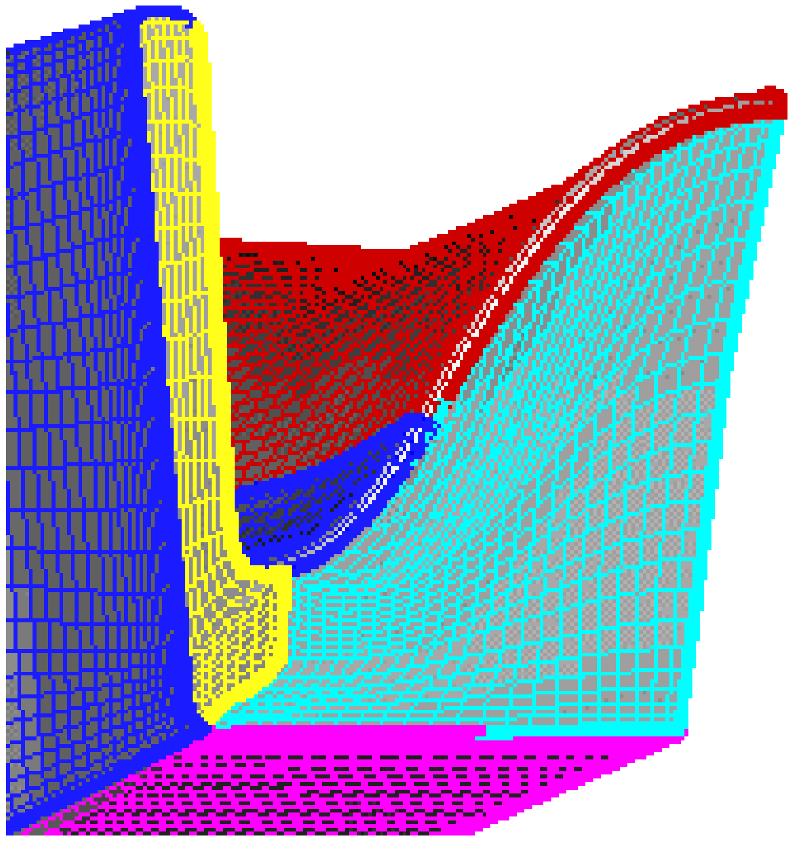
Axis Orientation: Different Local Index Coordiante Systems

- Reason:



- Solution: Changing storage order according to the difference during the communication (Currently sender side)

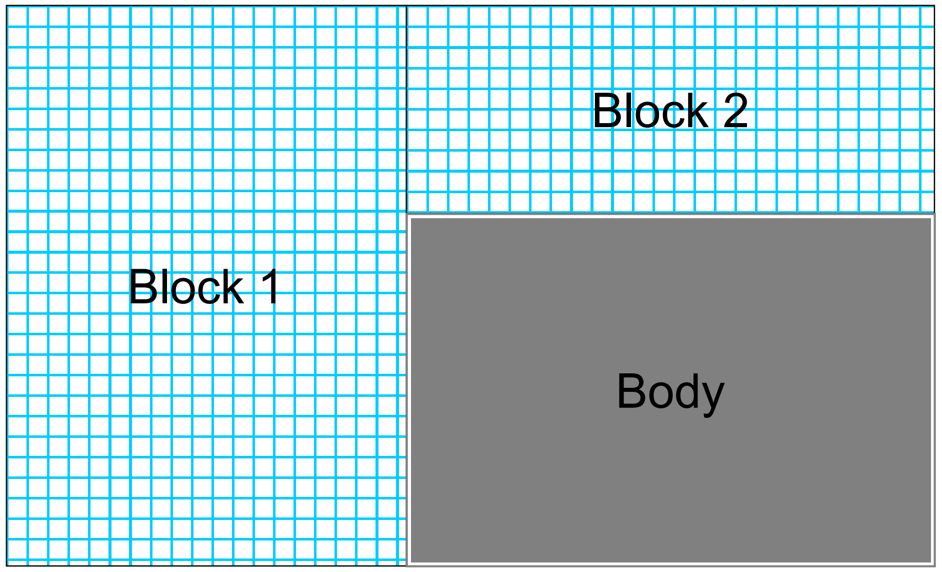
Neighbour Dependencies - Occurence



- A block may have more than one neighbour at one blockside

Physical Boundaries

- C-mesh:
  - a specific physical boundary type is bound to a specific block side
- physical boundaries in multiblock meshes can occur on all of the block sides:



Efficient Calculation of Boundaries

- Special data structure for each boundary type:
  - location of each boundary
  - subtype of the boundary
- Only one code segment for boundary handling
  - no doubling of code for each side
    - **one code to update and maintain**
    - **no cut and paste bugs**
- No branches
  - chance of performance improvement

Load Balancing

- Target: Efficient use of Massively Parallel Processors

- Blocks have different size
- Block number is generally different from number of used processors
- Initial load balancing is necessary

- Problems to solve:
  - There are blocks which are too large to be calculated efficiently onto one processor
  - Block splitting necessary
  - There are blocks which are too small to be calculated alone onto one processor
  - Process should be able to calculate more than one block at a time

Extensions for block handling

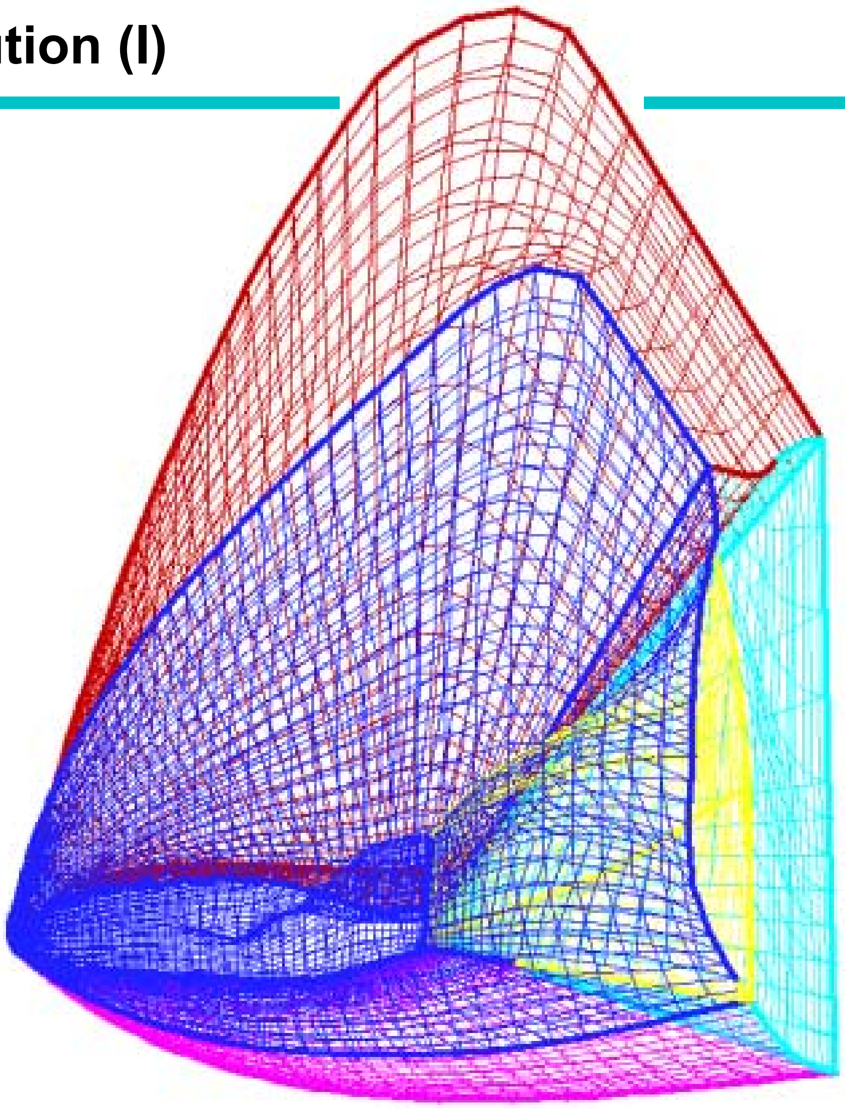
- Different block numbers on a process
  - Extension of the subroutines and algorithms or block loops around subroutines
  - Communication between blocks on one process is done using MPI
  - Extension of the communication structure, so that each incoming message reaches its block

Load Balancing Step

- Using (parallel) jostle to distribute the obtained blocks to the available processors
- Generating a graph out of the block distribution with:
  - nodes representing the blocks
  - node weight representing the block size (computational effort)
  - edges representing the neighbour dependencies between blocks
- block redistribution according to jostle's suggestion

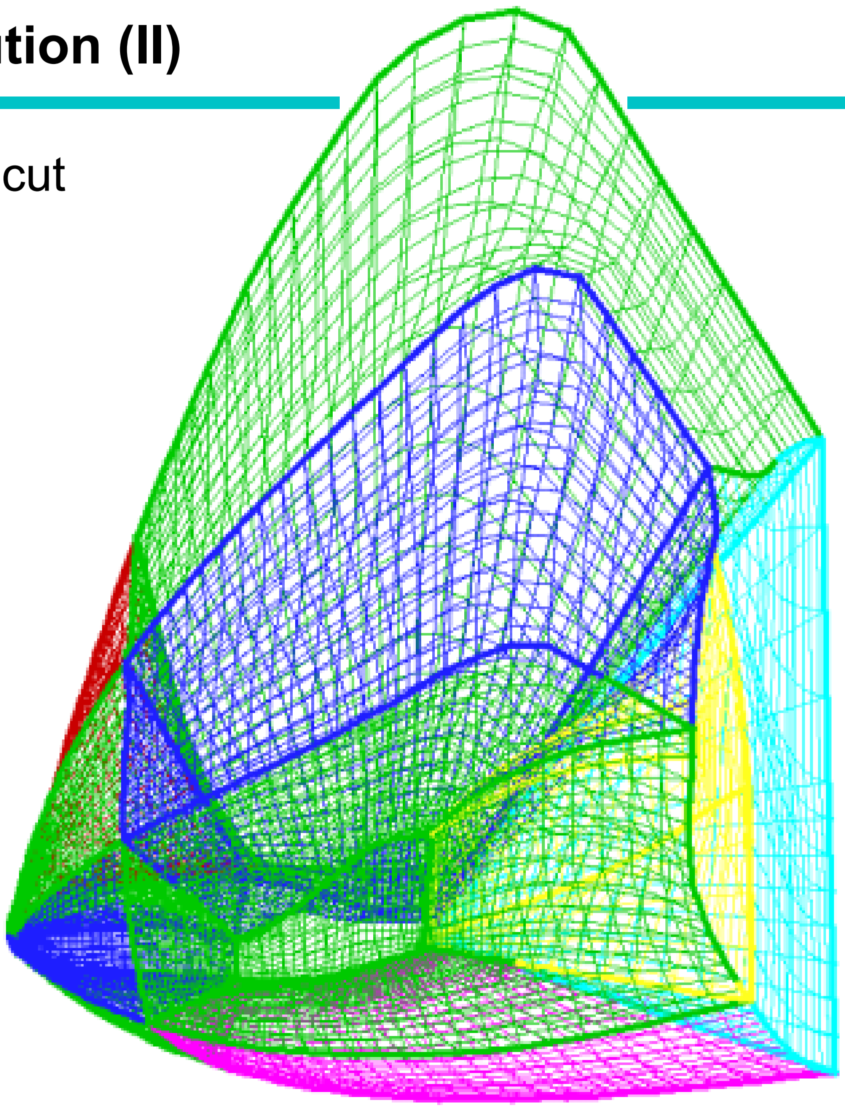
Example Block Distribution (I)

- X-38 mesh:
  - 6 blocks
  - 68712 cells
  - largest: 32256
  - smallest: 1176



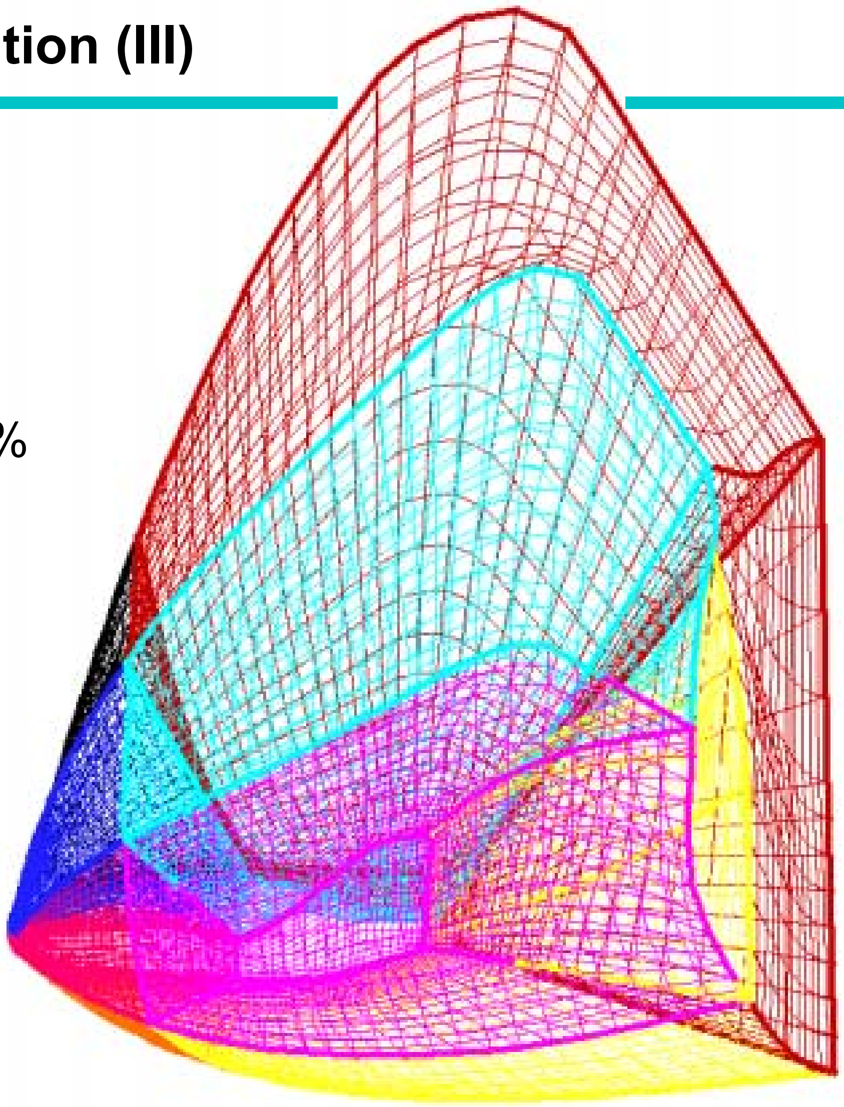
Example Block Distribution (II)

- Same mesh with blocks cut automatically:
  - 9 procs
  - 11 blocks



Example Block Distribution (III)

- Blocks distrubuted to the processors:
  - largest 8064
  - smallest 6048
  - load imbalance: 18 %

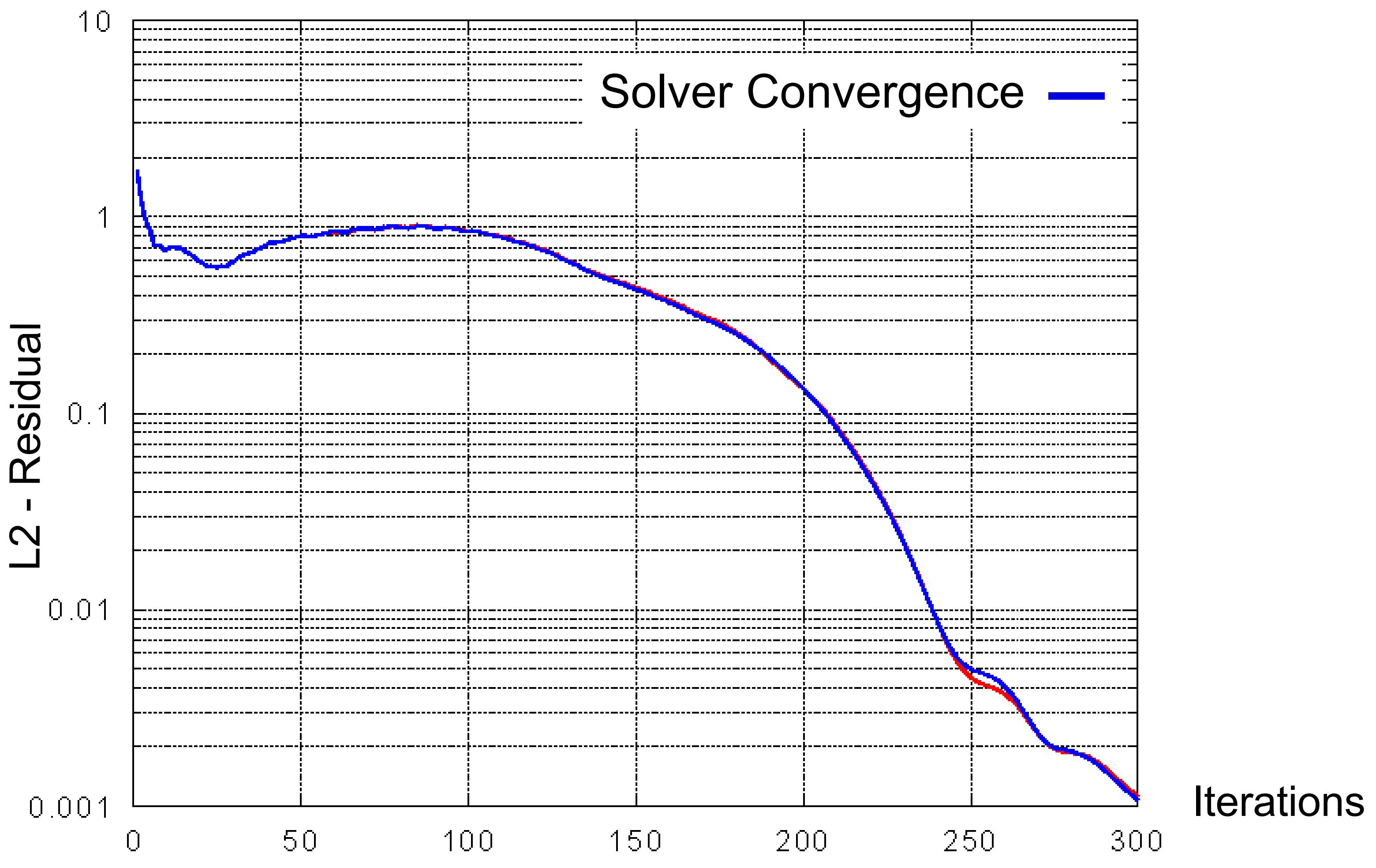


Parallelization Solver -- Idea

$$A = L + M$$

$$A = \left(\begin{array}{cccc|cccc|cccc}
m_1 & u_1 & & & & & & & & & & \\
l_2 & m_2 & u_2 & & & & & & & & & \\
 & l_3 & m_3 & u_3 & & & & & & & & \\
 & & l_4 & m_4 & & & & & & & & \\
\hline
 & & & & m_5 & u_5 & & & & & & \\
 & & & & l_6 & m_6 & u_6 & & & & & \\
 & & & & & l_7 & m_7 & u_7 & & & & \\
 & & & & & & l_8 & m_8 & & & & \\
\hline
 & & & & & & & & m_9 & u_9 & & \\
 & & & & & & & & l_{10} & m_{10} & u_{10} & \\
 & & & & & & & & & l_{11} & m_{11} & u_{11} \\
 & & & & & & & & & & l_{12} & m_{12}
\end{array}\right) + \left(\begin{array}{cccc|cccc|cccc}
 & & & & & & & & & & & \\
 & & & & & & & & & & & \\
 & & & & & & & & & & & \\
 & & & & u_4 & & & & & & & \\
\hline
 & & & l_5 & & & & & & & & \\
 & & & & & & & & & & & \\
 & & & & & & & & & & & \\
 & & & & & & & & u_8 & & & \\
\hline
 & & & & & & & l_9 & & & & \\
 & & & & & & & & & & & \\
 & & & & & & & & & & & \\
 & & & & & & & & & & &
\end{array}\right)$$

Parallelization Solver -- Convergence

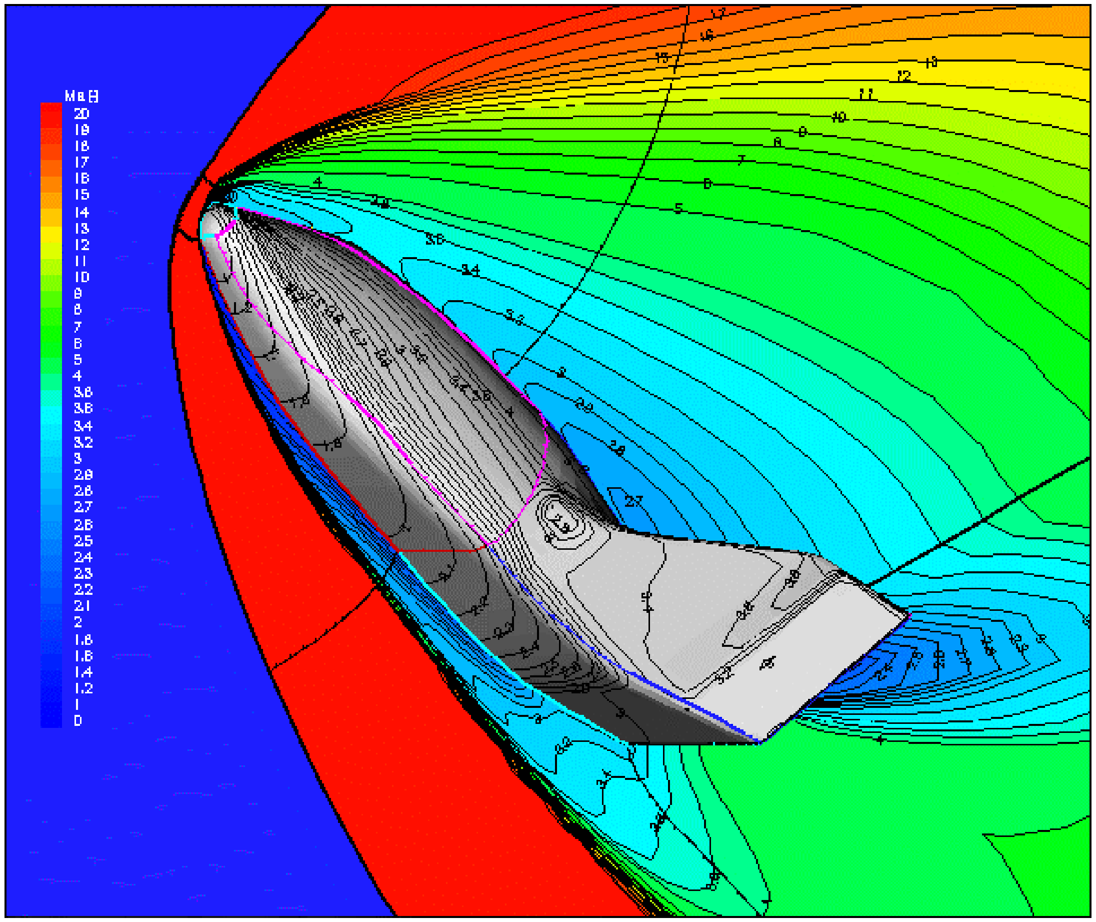


Results: Portability

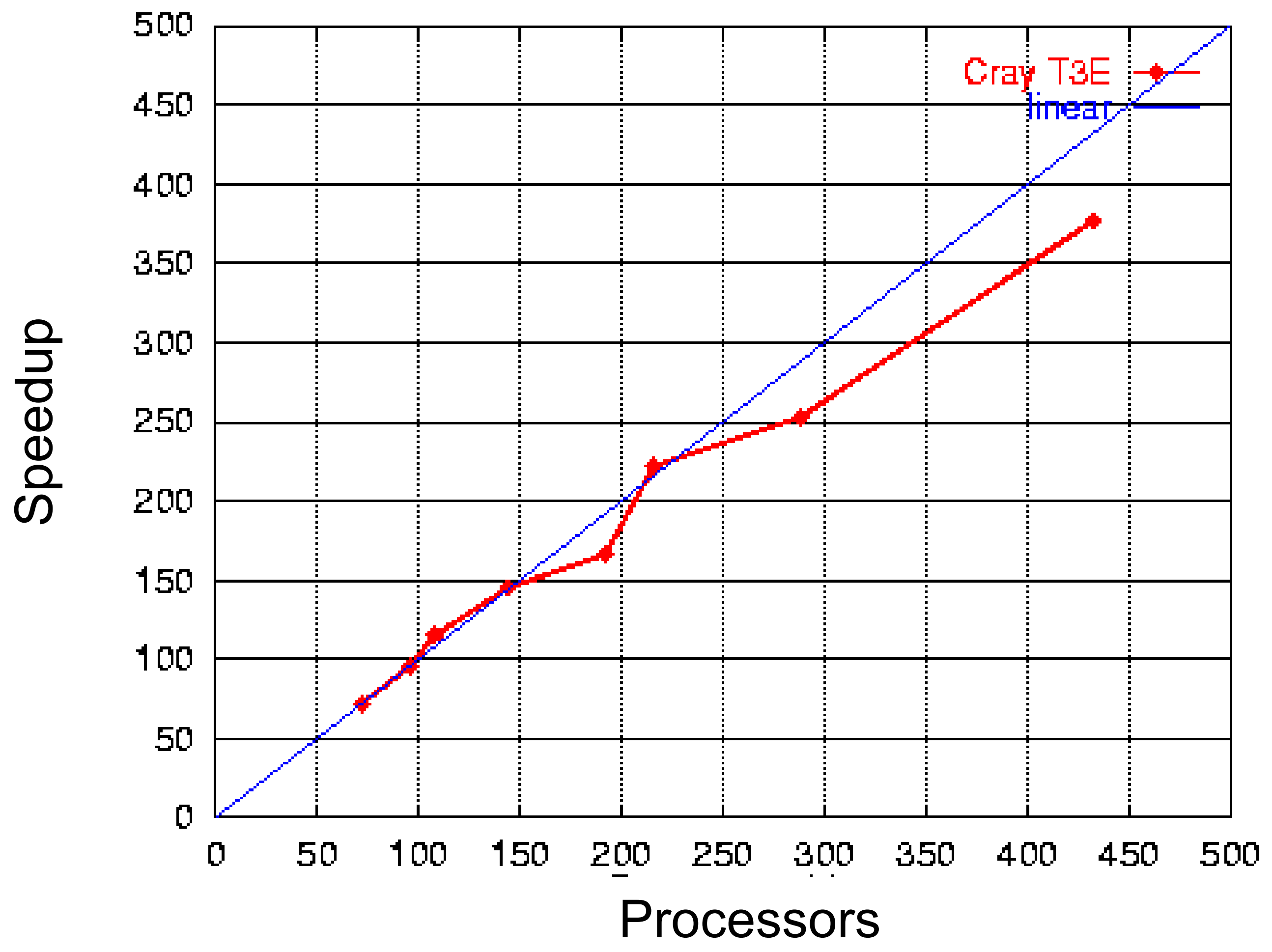
- Portability is achieved due to the usage of
  - MPI
  - Fortran90
- Was tested on:
  - Cray T3E
  - SX-5
  - SR8000
  - IA 64
  - PIII 1GHz
- Earlier Version run on:
  - IBM SP
  - Compaq Alpha-Cluster

Results

- Solution (X-38 NG)
  - mach 19.8
  - angle of attack 40°



Results Speedup



Results Scaleup T3E

Mesh size	Mesh blocks	Blocksno. for calculation i.e. Proc. count	Simulation time	Efficency
24 000	5	18	255.3	1.0
192 000	5	144	285.7	0.893

Outlook

- Adding the viscous fluxes for Navier-Stokes
- Adding the necessary algorithmic extensions for adaptive mesh refinement
  - refinement algorithm
  - interpolation at block boundaries
  - refinement criteria (gradient ?)
  - data structures are already prepared
- Migration of the metacomputing extensions

Summary: Parallel 3D-Multiblock URANUS

- Portable data parallel simulation program
  - Fortran90 (dynamic data structures)
  - message passing using MPI
- Domain decomposition based on structured multiblock meshes
- Different index directions within blocks
- Physical and inner boundaries on all block sides
- Different neighbour numbers on each block side possible
- Handling of different block sizes (automatic initial block distribution)
- Blocks not fitting on one process (load imbalance) are split automatically
- Number of blocks on each process only limited by memory