Daresbury Laboratory
Science & Technology
Facilities Council

Porting and Optimisation of Code_Saturne on HECToR

Shang Zhi, Charles Moulinec, David R. Emerson, Xiaojun Gu

Computational Science and Engineering Department
STFC Daresbury Laboratory, Warrington WA4 4AD

Outline

- Overview of the project
- Introduction to Code_Saturne
- Programme of works
- Software
- Current status
- Future work

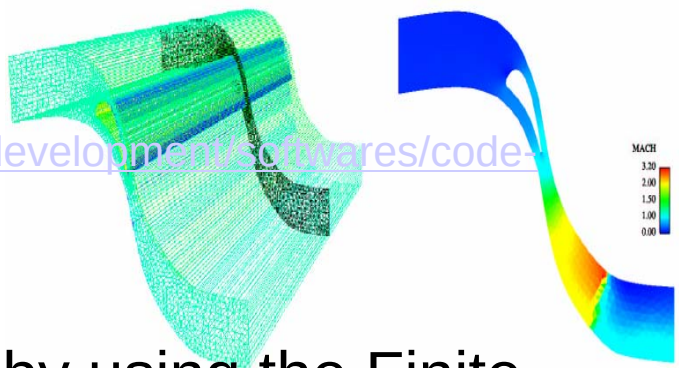
Overview of the project

To improve the current efficiency of Code_Saturne by improving the performance of the pre-processing.

Code_Saturne

Code_Saturne is an open source CFD code developed by Electricte de France (EDF), which can simulate either incompressible or expandable flows with or without heat transfer and turbulence.

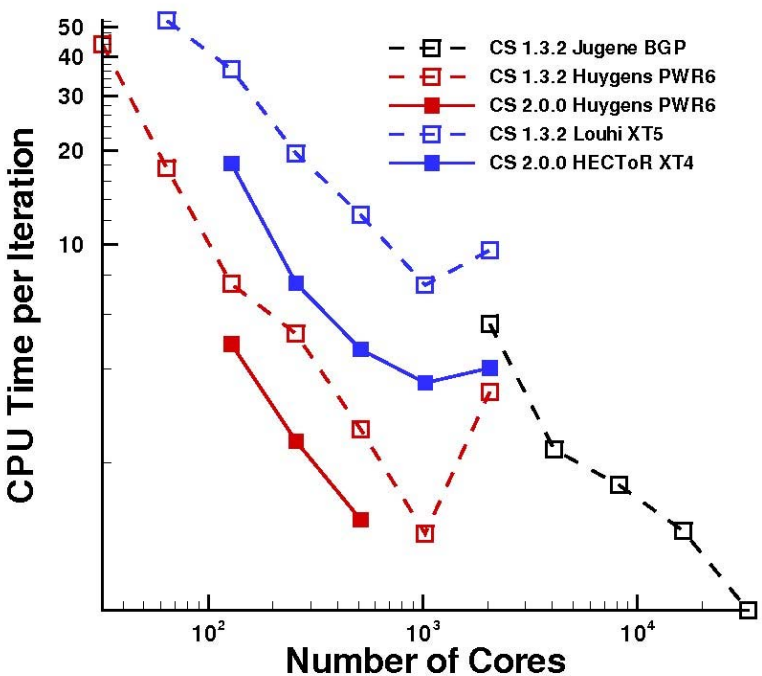
http://research.edf.com/the-edf-offers/research-and-development/softwares/code-saturne-107008.html



It solves the Navier-Stokes equations by using the Finite Volume approach on unstructured grids (elements of any shape, hexa, tetra, prisms, poly)

Code_Saturne's Performance

solver performance



The solver scales well on a number of platforms, including Cray's.

Version 1.3.2 uses the Conjugate Gradient method to solve the pressure and Version 2.0.0 uses multigrid acceleration.

Code_Saturne

Code_Saturne's Challenges - 1

The main challenges for this type of code are the mesh generation, the mesh partitioning, and the pre-processing, when the meshes are big (over 100M cells) and the number of cores is big (up to 100,000). The steps involved in a simulation are:

- Mesh generation
- Mesh partitioning
- Pre-processing
- Solving
- Post-processing

Code_Saturne

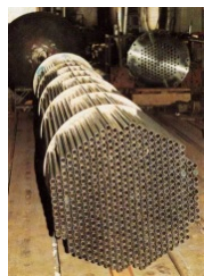
Code_Saturne's Challenges - 2

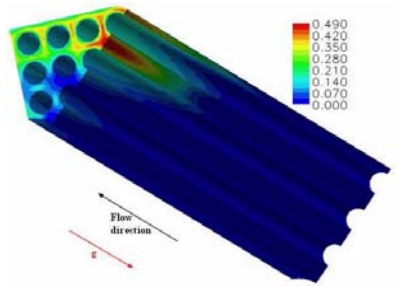
The mesh generators (at least OpenSource) are mostly serial, which limits the number of cells per mesh.

- Mesh partitioning: How to partition a 100M-200M cell mesh efficiently?
- How to pre-process the data for this type of mesh?

Challenges not only for Code_Saturne

If the parallel application can be used on a machine having 100k cores with ~40k cells (meshes) per core (memory dependent), potentially it can deal with a huge mesh size (4000M and above) application. For example to tackle the complete tube bundles in a nuclear reactor.





- Huge mesh generation (how to do ?)
- Mesh partitioning (performance ? It is a NP-complete problem)
- CFD software (ability ?)
- Solving algorithm (accelerator ?)
- Post-processing (how to do ?)

Programme of work

The planned work has been split in two distinct phases as shown in the table

Task	Duration	Objective
Incorporate parallel partition	10 months	Understand effect of partition on load balance issue for parallel computation of code
MPI-IO for post-processing/restart	8 months	Improve scalability of code

Software

- Mesh generation: icemcfd, gmesh, others e.g. GridGen or Pointwise
- Partitioning: Metis, ParMetis, PT-Scotch and Zoltan
- Visualization: PMVIS, Paraview, or other open source codes
- Main code: Code_Saturne

Mesh Generation

GMSH (version 2.4.0)

Gmsh is a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. It is employed to generate the original mesh for Code_Saturne.

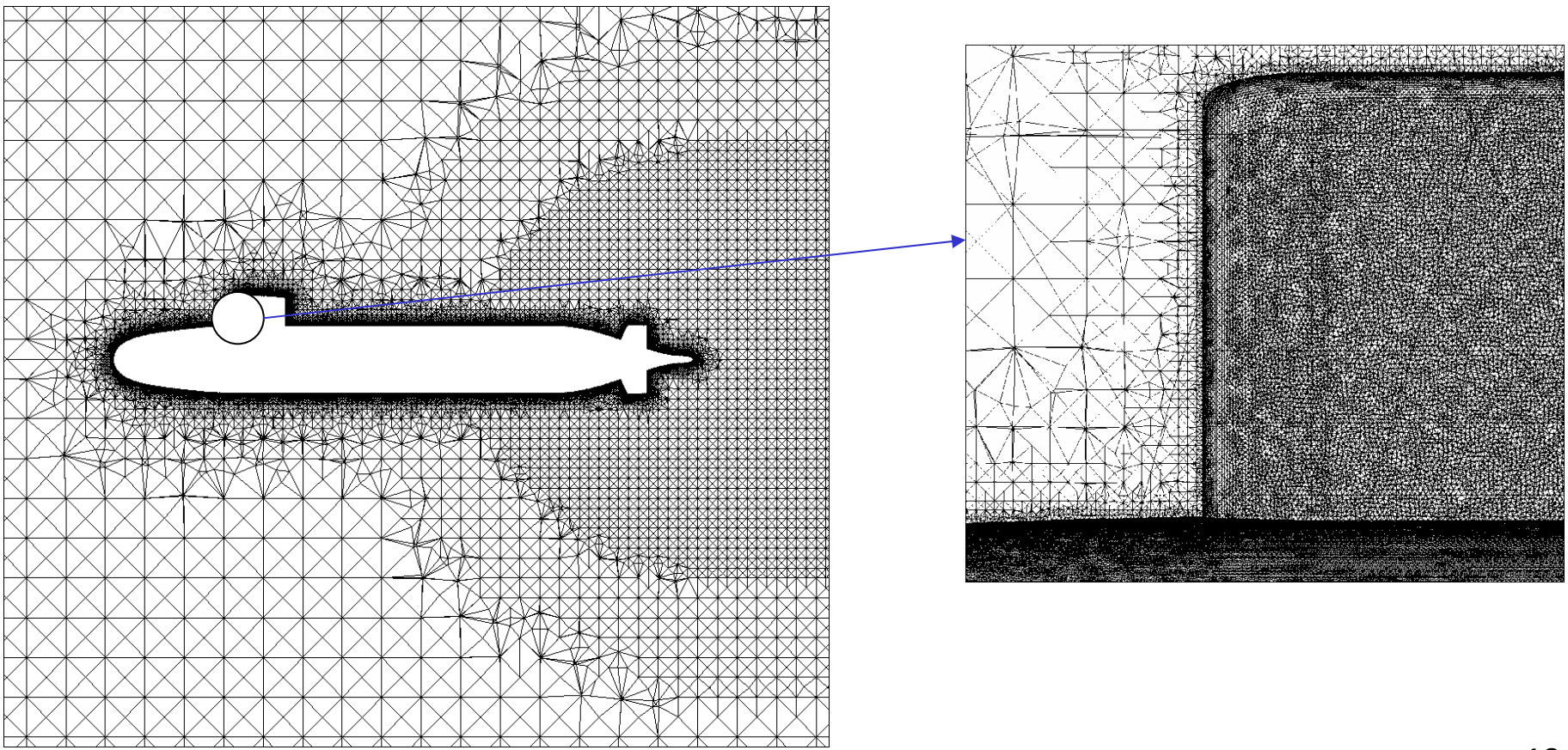
http://www.geuz.org/gmsh/

icemcfd (version 11.0)

Icemcfd is a commercial CFD software package. It is employed as a mesh generation tool to generate the unstructured meshes for Code_Saturne.

Mesh Generation

107M tetra mesh generated by icemcfd for a submarine



Mesh Generation Discussion

Currently the 121M tetra mesh generation by icemcfd for the submarine is finished on SGI machine in Daresbury Lab.

The critical issues are:

1. the memory usage reaching 96.66Gb (the available memory in SGI is 96.69Gb)
2. the time usage is over 24 hours (1.5GHz CPU speed)

Future mesh generation

1. try other software for the large mesh generation
2. develop some patching codes for the huge mesh (4000M or above)

Current status of mesh partitioning

- Metis
- ParMetis
- PT-Scotch
- Zoltan 

Partition methods

¶ The concept of a partition is based on a graph. The graph contains vertices and edges.

vertices = objects to be partitioned

edges = communication between objects

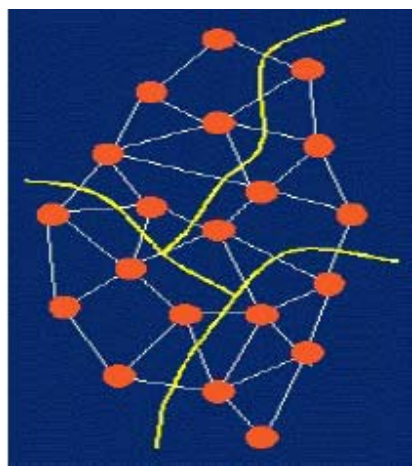
The partition means to separate the vertices into a number of groups through cutting the edges.

Therefore, before partitioning, the original meshes have to be transferred as a graph that can be partitioned. The transfer can be carried out by the partitioning software itself or by the user's own developed code.

¶ Metis and ParMetis uses k-way partitioning

¶ PT-Scotch use dual recursive bi-partitioning

¶ Zoltan uses graph partitioning



Metis, ParMetis

- Metis (version 4.0 and 5.0, sequential)

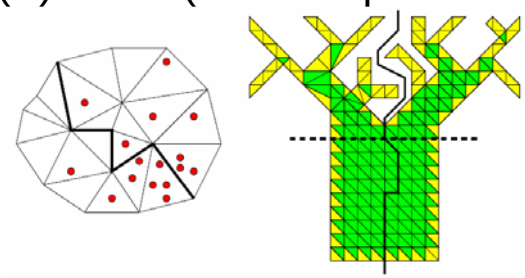
  METIS, developed by Karypis lab. in USA, is a family of programs for partitioning unstructured graphs and hypergraphs and computing fill-reducing orderings of sparse matrices. It can be employed as the pre-processing of mesh distributions for parallel computation among multi-processors.

- ParMetis (version 3.1.1, parallel)

  ParMetis is the parallel version of Metis.

  Math principle is : Graph=(Vertices, Edges) partitioned to k parts (ways) $\{V_1,V_2,...,V_k\}$ minimizing (edge cuts) $\epsilon C(P)$ containing (load imbalance) $LI(P) \leqslant 1+\delta$ (for example $0 \leqslant \delta \leqslant 0.2$).

  http://glaros.dtc.umn.edu/gkhome/views/metis



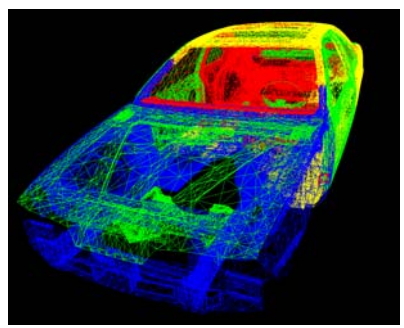
PT-Scotch

¶ PT-Scotch (version 5.1, parallel)

PT-Scotch, developed by Laboratoire Bordelais de Recherche en Informatique in France, is a family of software package and libraries for sequential and parallel graph partitioning, static mapping, and sparse matrix block ordering, and sequential mesh and hypergraph partitioning. It is the parallel version of Scotch.

Math principle is: mapping the source graph S to the target graph T ($S \longrightarrow T$) including vertices $V(S) \longrightarrow V(T)$ and edges $E(S) \longrightarrow P(E(T))$ minimizing the communication cost function $f_C(\tau_{S,T}, \rho_{S,T})$, where $\tau_{S,T}$ is the communication of vertices and $\rho_{S,T}$ is communication of edges.

http://www.labri.fr/perso/pelegrin/scotch/



Zoltan

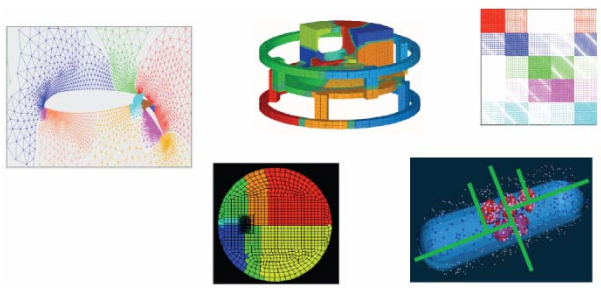
¶ Zoltan (version 3.1, parallel)

Zoltan, developed by Sandia National Laboratories in USA, is a family of software packages and libraries for parallel partitioning, load balancing and data-management services.

Math principle is: a graph G=(N, E, W_N, W_E) where N is nodes or vertices; E is edges; W_N is node weights and W_E is edge weights.

Graph partition is to choose the partition {N_1, N_2, ..., N_k} containing

1. the sum of node weights in each part N_j is close to equal.
2. the sum of edge weights connecting all different parts N_i and N_j is minimised.



http://www.cs.sandia.gov/~kddevin/Zoltan_html/Zoltan_FAQ.html

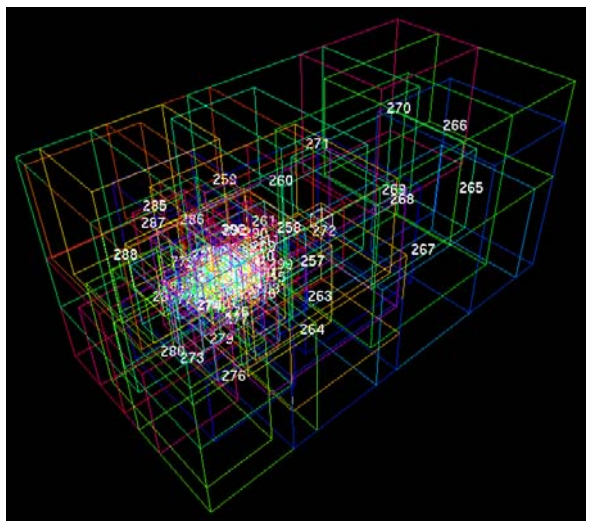
Visualization

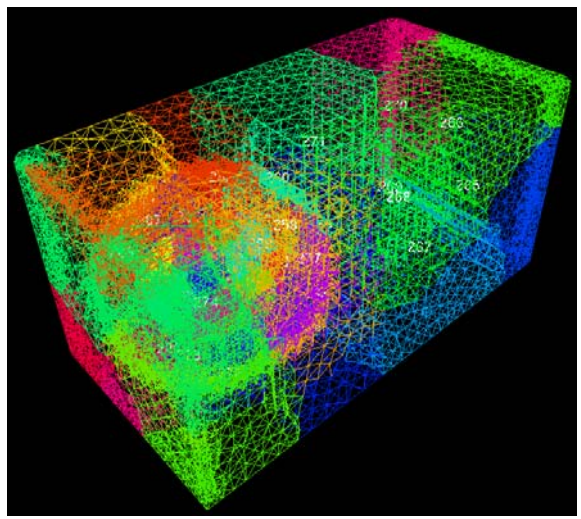
- PMVIS
- Paraview

PMVIS

- PMVIS (version 1.09)

  PMVIS, developed by University of Minnesota in USA, is software to visualize the results of the graph partitions by the graph partition software. It can help the user to examine and understand the partitions.



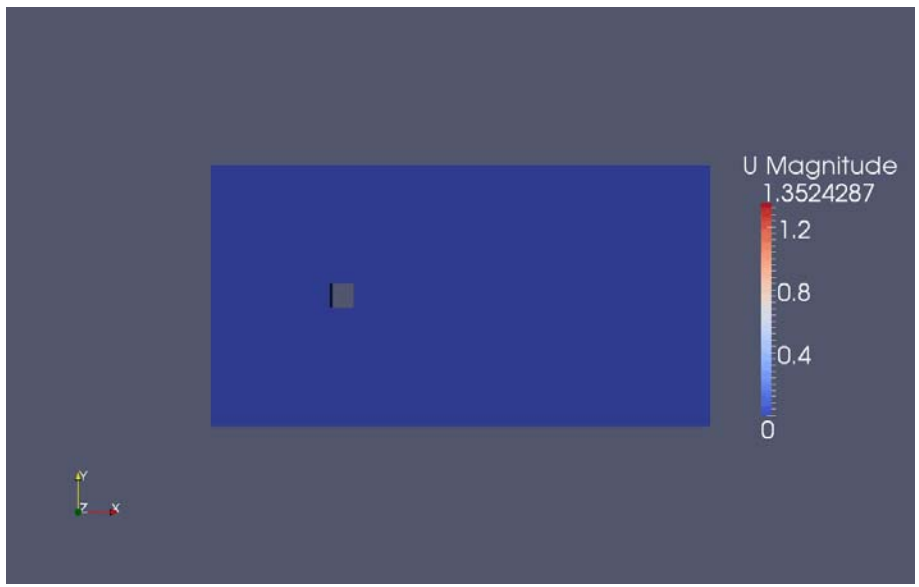


http://www-users.cs.umn.edu/~oztekin/pmvis/

Paraview

- Paraview (version 3.6.1)

  ParaView is an open-source, multi-platform application designed to visualize data sets of size varying from small to very large.



http://www.paraview.org/Wiki/ParaView

Partitioning Results

- Partition time is used to compare the partition speed
- Edge cuts is used to compare the partition efficiency
- Capacity statistic: Mesh number in each partition domain is used to measure the load balance
- Neighbours statistic: Neighbour domain number is used to compare the future CFD calculation data communication among the multi processors.

Information of 5.7M meshes

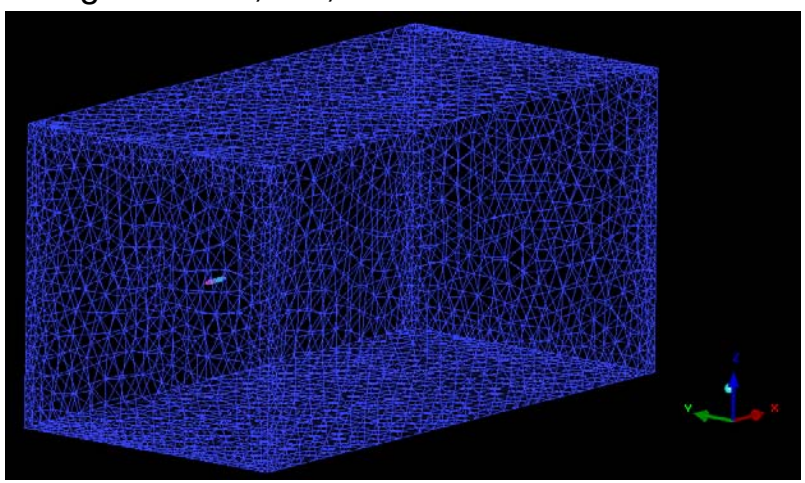
¶ Metis, ParMetis, PT-Scotch

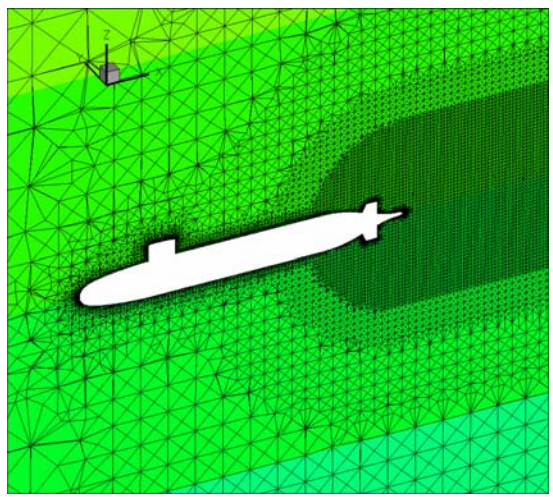
- A test case is already built up with 5.7 million unstructured meshes.

  Elements: 5,780,325

  Vertices: 983,576

  Edges: 11,532,131





Sequential results on PC

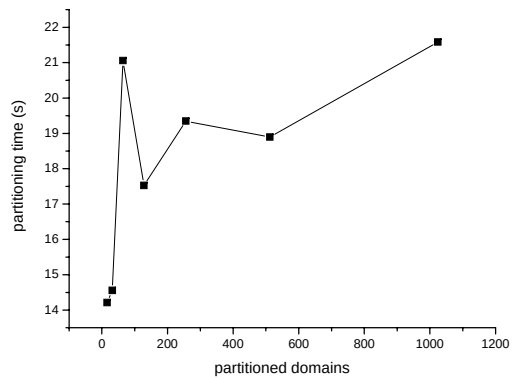
Metis4.0 and 5.0 are serial therefore processor number = 1

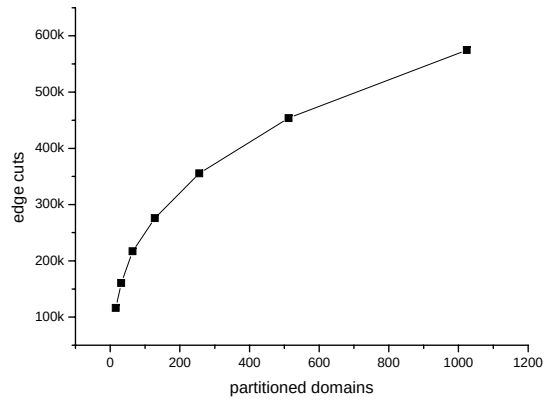
Parmetis3.1.1 is parallel but only use one processor

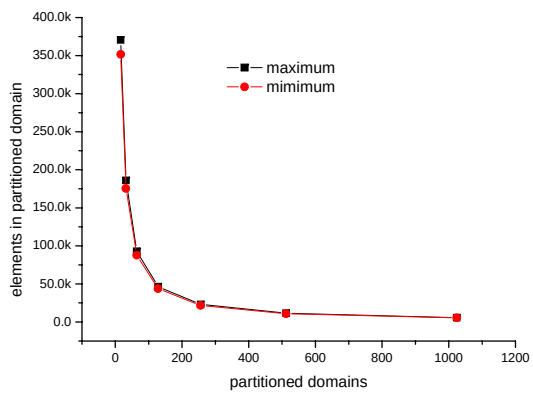
PT-Scotch5.1 is parallel but only use one processor

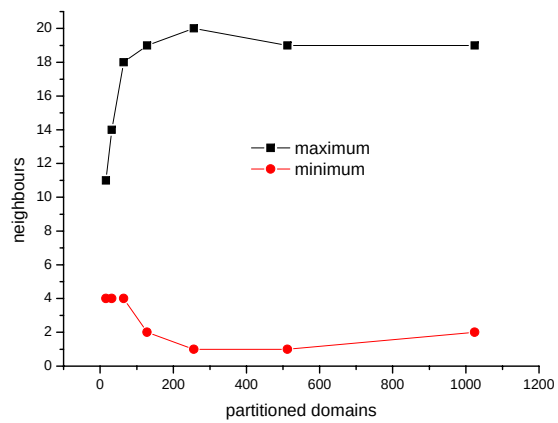
Results of Metis5.0

¶ Serial Partition under different domains



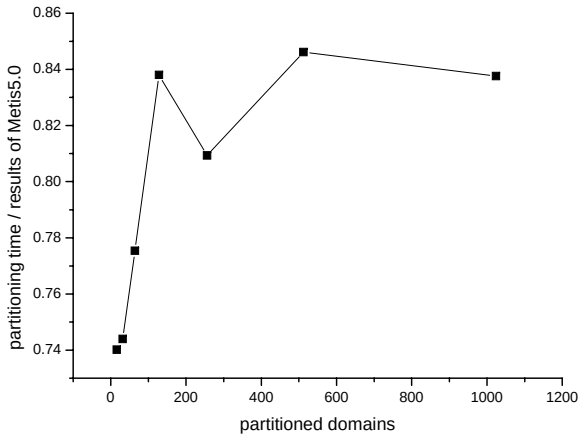


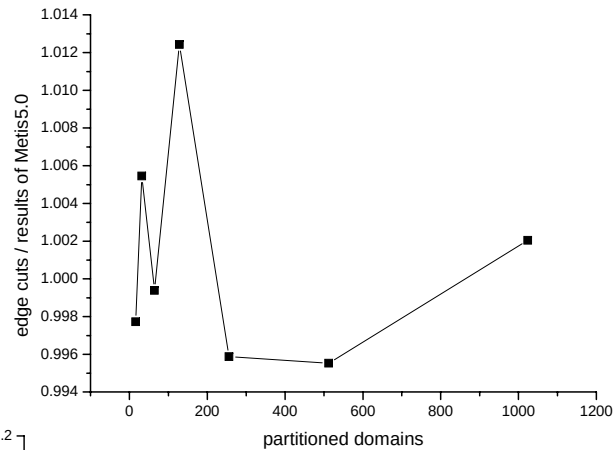


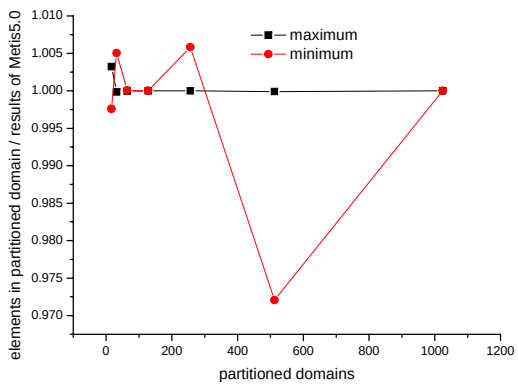


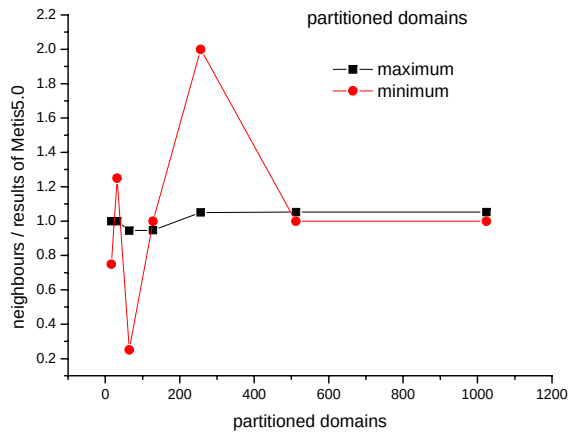
Results of Metis4.0

¶ Serial Partition under different domains/results of Metis5.0



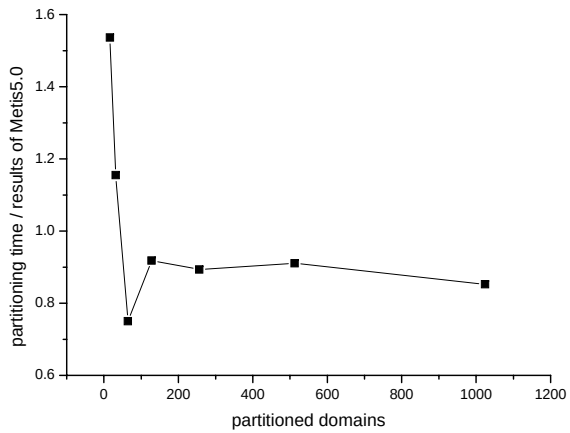


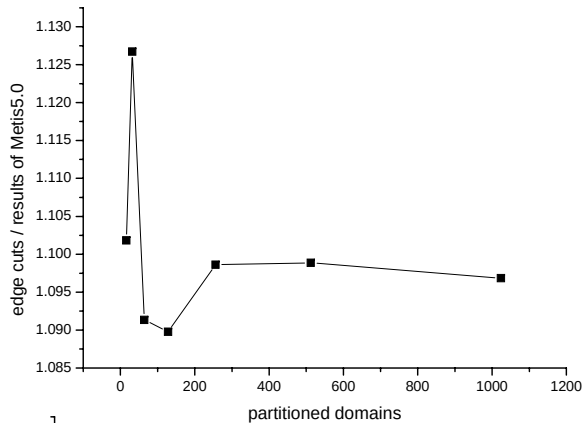


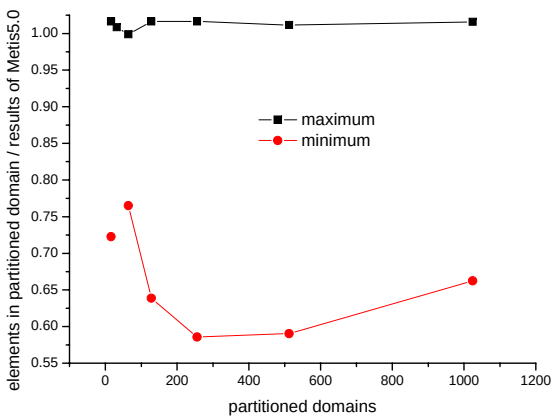


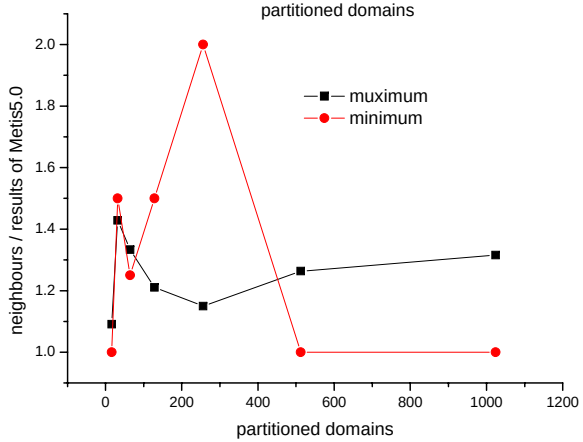
Results of ParMetis3.1.1

¶ One processor Partition under different domains/results of Metis5.0



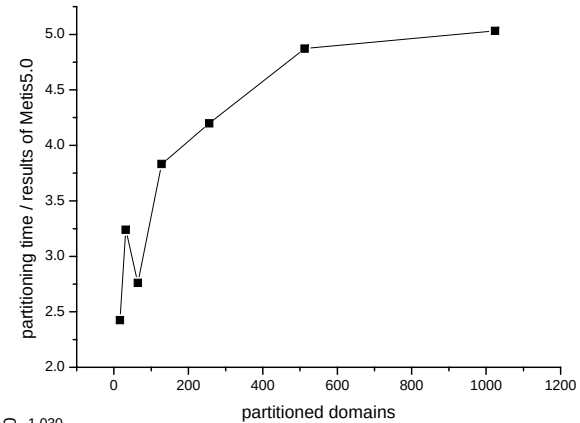


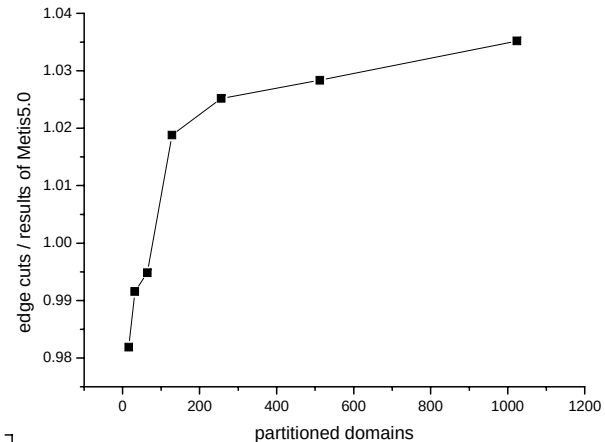


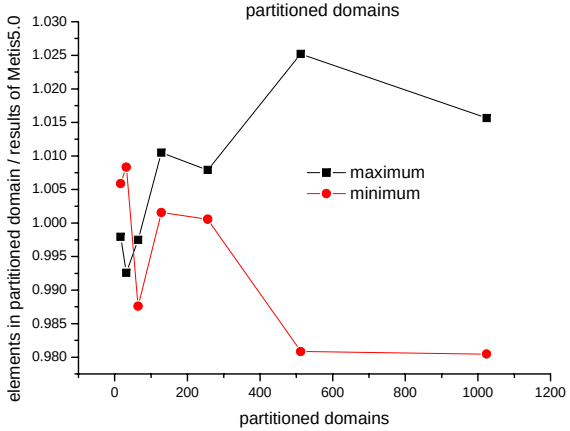


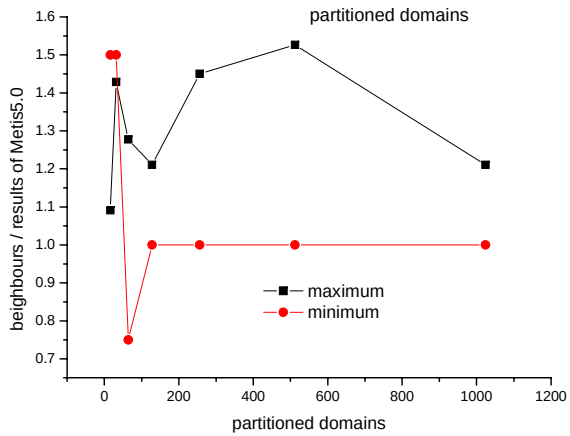
Results of PT-Scotch5.1

¶ One processor Partition under different domains/results of Metis5.0









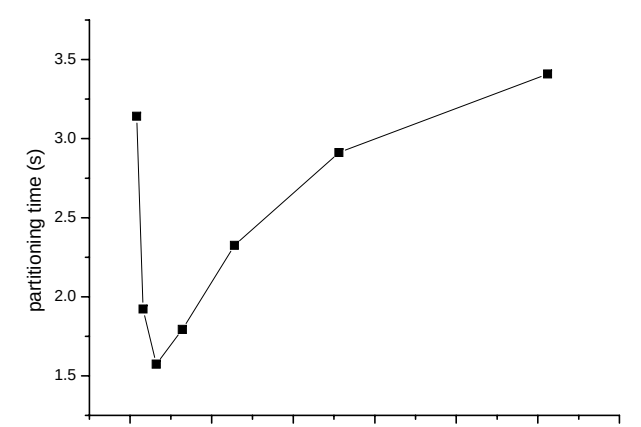
Multi-processor results on HECToR

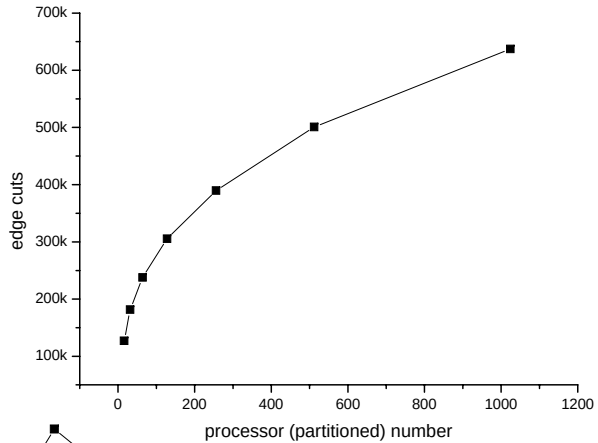
Processor number 16 – 1024 for Parmetis3.1.1 and the partitioning domain number = processor number

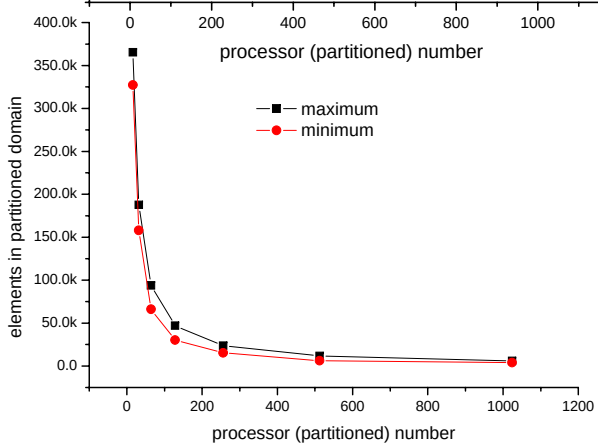
Processor number 16 – 1024 for PT-Scotch5.1 and the partitioning domain number = processor number

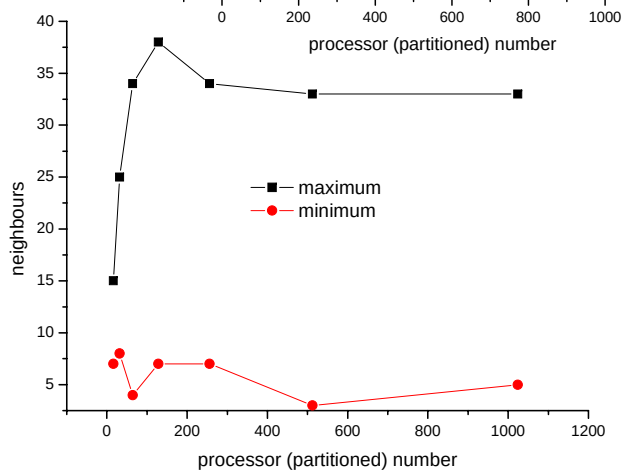
Results of ParMetis3.1.1

¶ Partition under different processors



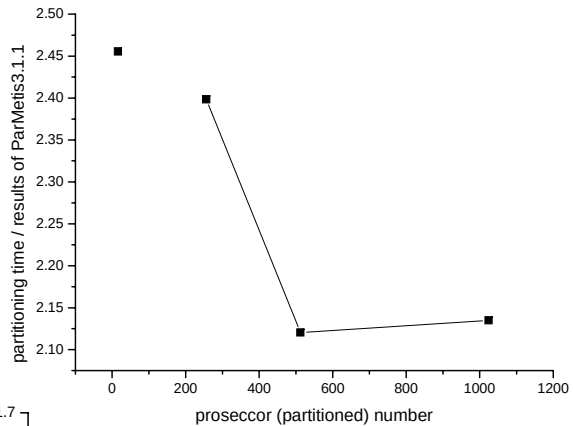


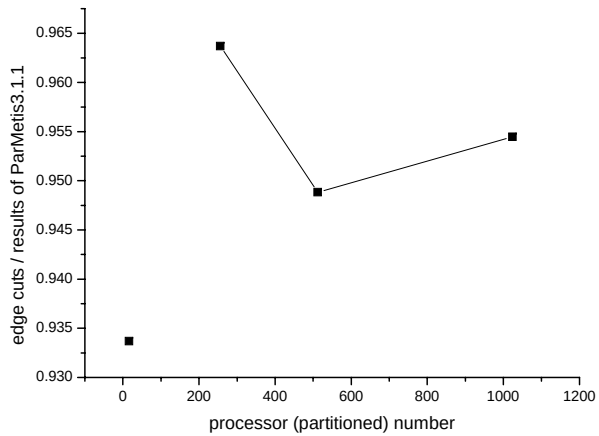


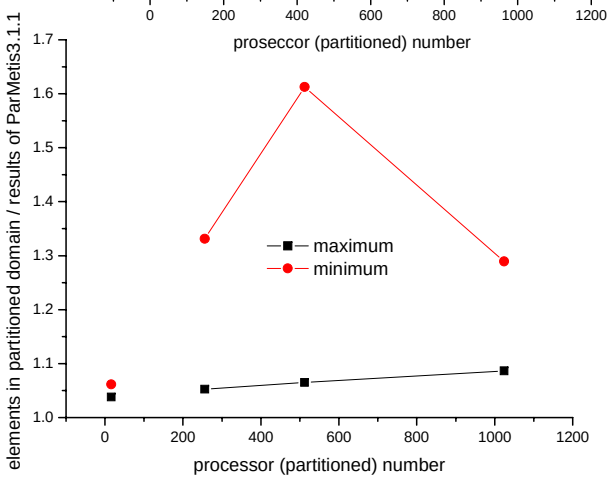


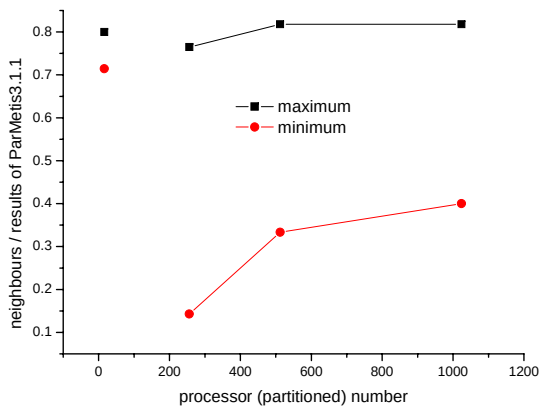
Results of PT-Scotch5.1

¶ Partition under different processors/results of ParMetis3.1.1



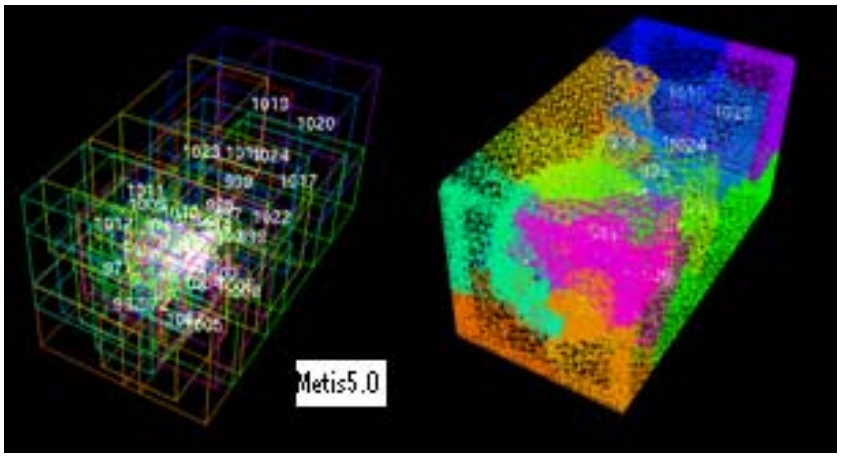


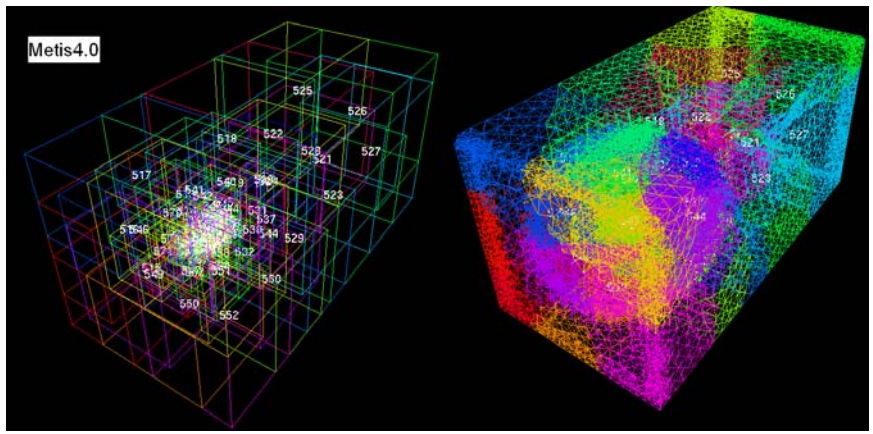


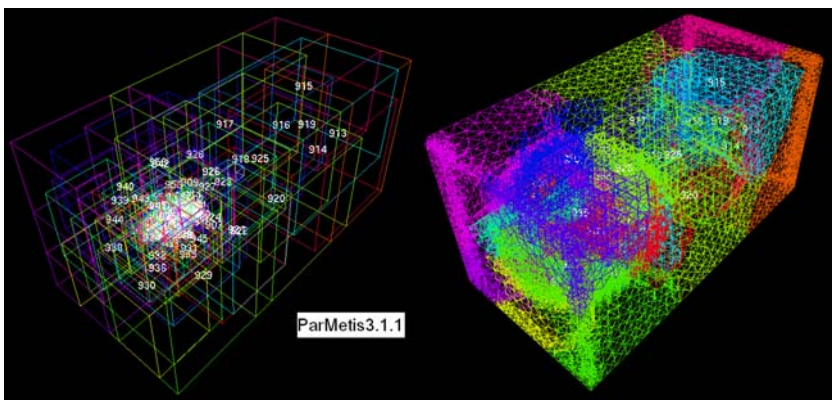


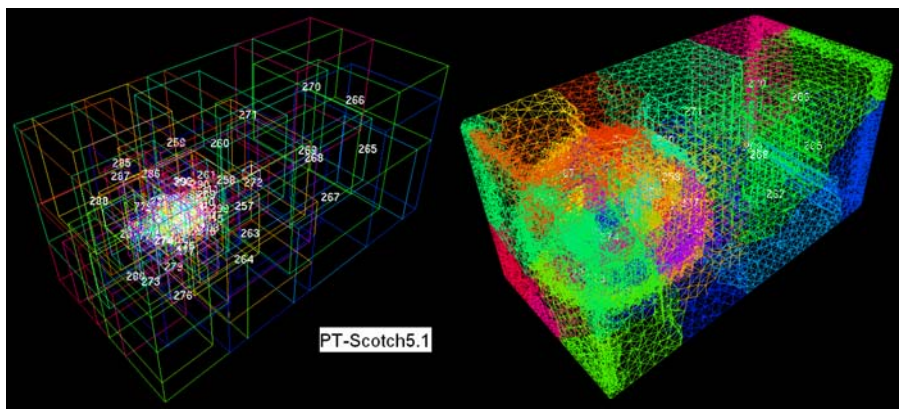
Partition visualizations

Comparisons of the partition results for 5.7M unstructured meshes of 1024 parts (domains)









Current status for large mesh

- Metis5.0
- ParMetis3.1.1
- PT-Scotch5.1
- Zoltan3.1

Information of large meshes

The test case is already built up with 108 million unstructured meshes.

Elements: 107,673,905

Vertices: 18,974,160

Edges: 213,553,779

The test case is already built up with 121 million unstructured meshes.

Elements: 121,989,150

Vertices: 21,367,880

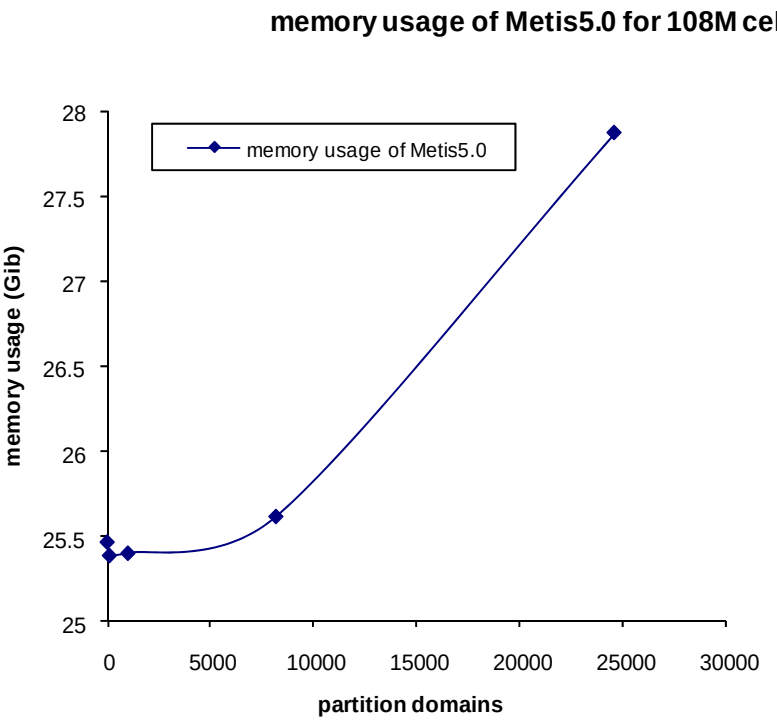
Edges: 242,184,039

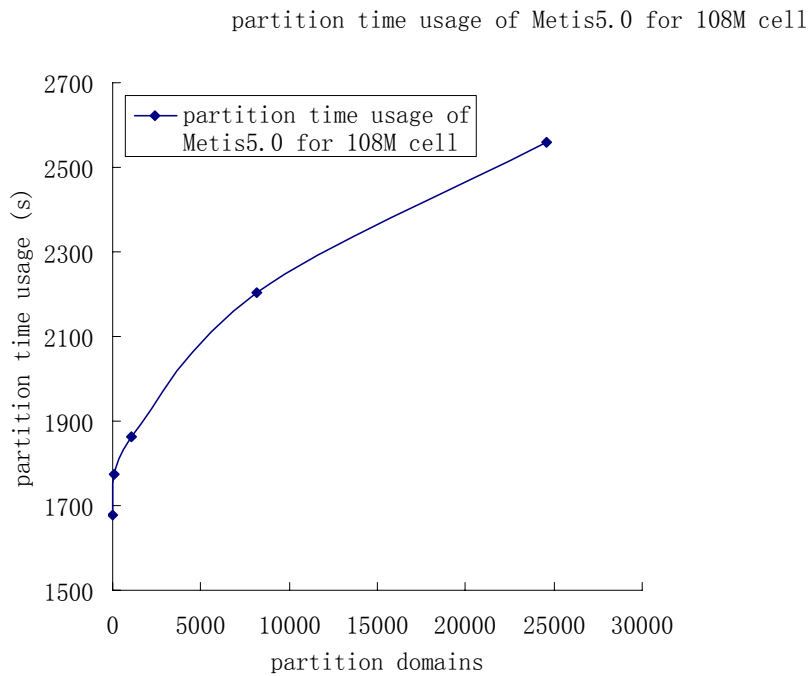
Metis5.0 performance

The partitioning is carried on machine SGI in Daresbury Lab. Metis4.0 has the same performance as 5.0

Memory usage for 107M cells: the sequential partition failed for domains larger than 32,000. On the positive side, only ~10% increase in memory requirements when the domains increasing 5 times.

Partition time usage for 107M cells: the time usage increases 1.5 times from 16 domains to 24588 domains.

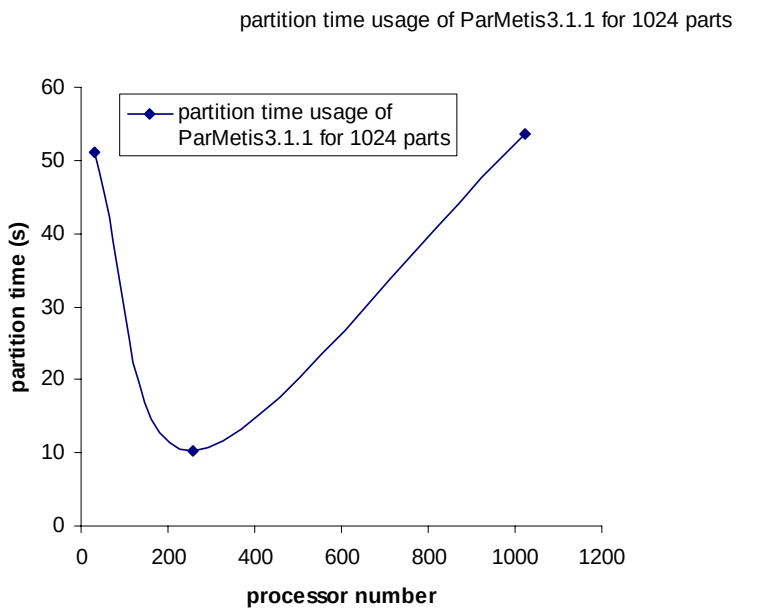




ParMetis3.1.1 performance

The partitioning is carried out on HECToR

Partition time usage for 121M cells under 1,024 parts

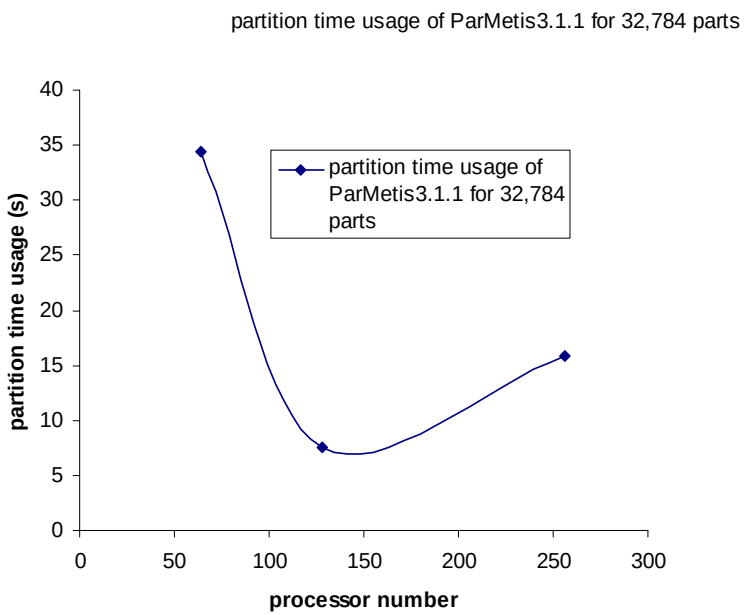


Metis5.0 (sequential) uses 2091.246 seconds for this partition on SGI

ParMetis3.1.1 performance

The partitioning is carried out on HECToR

Partition time usage for 121M cells under 32,784 parts



Metis5.0 (sequential) can not make the partition under 32,784 parts due to too many parts for Metis5.0

Metis4.0 has the same performance as 5.0

PT-Scotch5.1 and Zoltan3.1

Current status of Code_Saturne

- Starting
- Small mesh test (still doing)
- Large mesh test (still doing)
- Improvement (still doing)

Starting for Code_Saturne

Code_Saturne

- Getting familiar with Code_Saturne.
- Including as following issues:
- How to prepare a CFD case for Code_Saturne.
- How to set up the boundary conditions.
- How to set up the initial guess.
- How to set up the control parameters.
- How to output the simulation results.
- How to visualize the simulation results, etc.

Future work

- Incorporate the partition software into Code_Saturne
- Compare the efficiency of Code_Saturne with these partition software for large meshes (100-200 million cells, currently the largest we are able to generate with icemcfd)

The End