

# A Scalable Domain Decomposition Method for Ultra-Parallel Arterial Flow Simulations

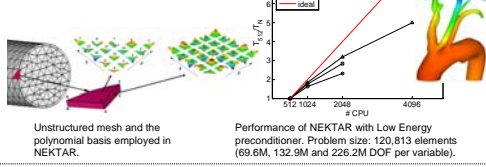
## Background

The human vascular system is very complex, with five liters of blood traveling 60,000 miles in just one minute. Interactions of blood flow in the human body occur between different scales with large-scale flow features coupled to cellular and sub-cellular biology, or at similar scales in different regions of the vascular system. Numerical simulations of such blood flow in different regions of the vascular system require multiscale modeling across many orders of magnitude on spatial and temporal scales.

Blood flow simulation in the entire vascular system is an immense computational and algorithmic challenge. Solving 3D flow equations with billions of degrees of freedom in a reasonable amount of time requires the use of parallel solvers running on tens of thousands of processors. From the parallel computing standpoint, optimizing communication between thousands of processors on multicore computers is a great challenge.

## Nektar

Our numerical software NEKTAR is designed for the solution of computational fluid dynamic problems. NEKTAR is a parallel numerical library based on a high-order spectral/*hp* element method and employs an unstructured finite element mesh with a spectral expansion within each element. NEKTAR has exhibited very good parallel efficiency on BlueGene and the CRAY XT3.



## Methods

Ultra-parallel flow simulations on thousands of processors require new multi-level domain decomposition methods. We have developed such a new two-level method that has features of discontinuous and continuous Galerkin formulations. At the coarse level, the domain is subdivided into several big patches; a spectral element discretization (fine level) is employed within each patch. New interface conditions for the Navier-Stokes equations aiming to minimize data transfer are developed to connect the patches.

The two-level domain decomposition (2DD) requires multi-level partitioning of a global communicator. Hence, we have developed the Multilayer Communicating Interface (MCI), the primary purpose of which is to: 1) regroup available processors with respect to the hardware topology and the distinct computational tasks and 2) establish a communication pathway between processors from different groups based on multi-level parallelism. For communication between distributed computers, MCI uses the MPIg or MPICH-G2 library developed at ANL.

### Two-level Domain Decomposition

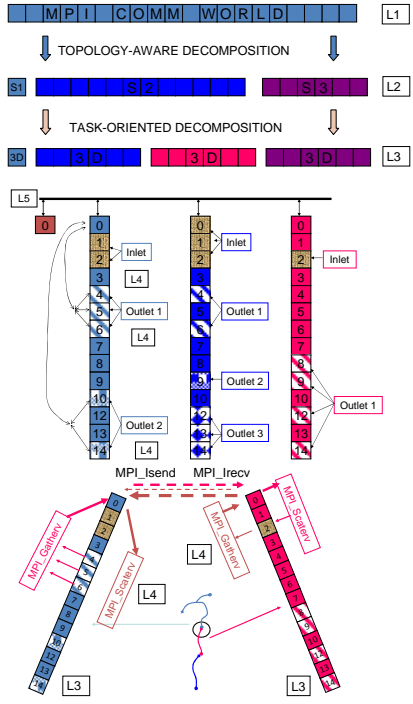
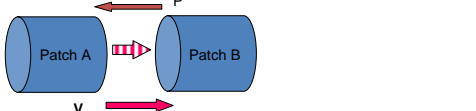
- Solution of the Navier-Stokes equations done on two levels:
  - Inner level: solve tightly-coupled problem with semi-implicit solver using standard, one-level domain decomposition (1DD).
  - outer level: solve loosely-coupled problem explicitly.
- Two types of communication:
  - local or intra-patch communications (processes from same group communicate)
  - global or inter-patch communications (processes from different groups communicate)

### Multilayer Communicating Interface (MCI)

- L1: is composed of all available processes (i.e., MPI COMM WORLD).
- L2: is derived from the first one using topology-aware splitting of MPI\_COMM\_WORLD, where processors from the same computer (site) are grouped in non-overlapping manner (groups S1, S2 and S3 in figure 2).
- L3: is derived from L2 using task oriented decomposition, each processor sub-group is dedicated for parallel solution of one tightly coupled problem (groups T1, T2, T3 and T4 in figure 2).
- L4: Solution of the tightly coupled problem may also involve specific tasks executed by some processes, hence the fourth layer (L4) of MCI is created. In the arterial flow simulation such tasks involve treatment of boundary conditions at the arterial inlets and outlets.
- L5: Data exchange between processes of all tasks is performed through the root processes of each task, grouped together on the fifth layer of MCI.

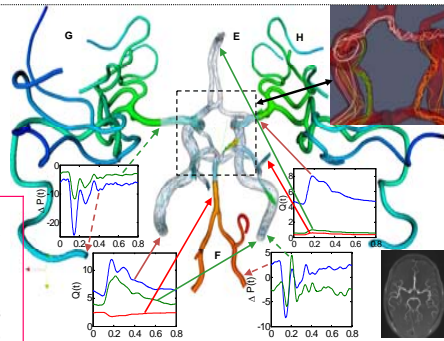
### Interface Boundary Condition

- Velocity computed at outlet; imposed as Dirichlet Boundary Condition at inlet.
- Pressure computed at inlet; imposed as Dirichlet Boundary Condition at outlet.
- Velocity flux from inlet averaged with velocity flux computed at outlet and imposed as Neuman Boundary Condition at outlet.



## Circle of Willis: Arterial Flow Simulation on CRAY XT3

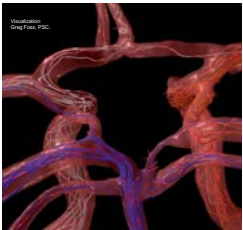
A total of 65 brain arteries with diameter of 1 to 4.5 mm were reconstructed from CT and MRI data. Velocity waveforms (measured with Doppler US) were used to impose inlet boundary conditions for arteries of the neck. Outflow boundary conditions at 31 outlets were imposed using RC boundary conditions (Grinberg & Karniadakis *Annl. Biomed. Eng.* 2008). High resolution simulation of one cardiac cycle on CRAY XT3 takes 80 hours on 3266 processors of CRAY XT3.



Patch	# of elements	Polynomial order	# of DOF per variable
Domain E	162,909	5	63,860,328
Domain F	44,632	5	17,495,744
Domain G	128,508	5	50,375,136
Domain H	123,201	5	48,294,792
<b>Total</b>	<b>459,250</b>	<b>5(6)</b>	<b>180,026,000 (264,528,000)</b>

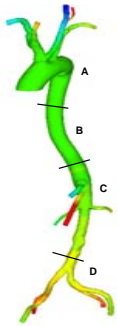
Total # of CPU	mean CPU-time / time step
2048	0.92 sec
3265	0.61 sec

Performance of Nektar on the CRAY XT3 of Pittsburgh Supercomputing Center.



Coarse level decomposition of the computational domain and number of Degrees of Freedom in each patch.

## Aorta: Simulation on CRAY XT3

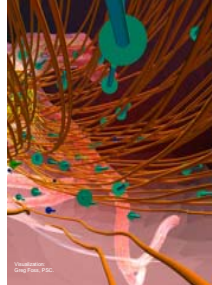
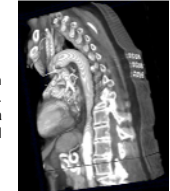


A total of 17 arteries including aorta and connecting vessels with diameter of 0.5 to 3.2 cm were reconstructed from CT and MRI data. The geometry was reconstructed from CT images using gOREK, a software package developed at Brown for reconstruction of arterial geometry from medical images.

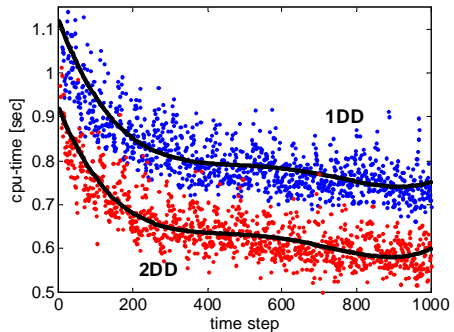
Domain	# of spectral elements	Polynomial order	# of DOF (per variable)
Domain A	120,813	6	69,588,288
Domain B	20,797	6	11,979,072
Domain C	106,219	6	61,182,144
Domain D	77,966	6	44,908,416
<b>Total</b>	<b>325,795</b>	<b>6</b>	<b>187,657,920</b>

Total # of CPU	256	994	1976
CPU time / time step	4.06 sec	1.24 sec	0.77 sec

Performance of Nektar on the CRAY XT3 of Pittsburgh Supercomputing Center.



## Large-Scale Flow Simulation on Ranger



Simulation with 1DD and 2DD on 1024 cores of Ranger. The domain is subdivided into two parts: patches C and D of the aorta (106,219 and 77,966 elements). Solution performed with P = 4. Dots – CPU-time per time step; solid lines – least-squares approximation.

The potentially great advantage of the 2DD approach is in simulating very large scale problems. Implementation of coarse partitioning of a computational domain into *M* patches leads to partitioning of the communicator into *M* non-overlapping groups of processes. The tight communication between processes is performed as intra-group message passing only. Minimization of inter-processor communication is particularly valuable on computers as Ranger where cores from the same node communicate significantly faster than cores from different racks.

In the figure on the left we compare performance of Nektar using standard and two-level domain decomposition. For simulation we use patches C and D of aorta shown above. Restricting communication in solution of tightly coupled problem to smaller group of processors enhances parallel efficiency.

The two-level domain decomposition approach was tested in simulation of a flow in the domain consisting of **147 arteries**. The domain consists of **1,244,295 spectral elements** and the total number of unknowns (for all four variables and P=6) was **2.87 billions**. Size of L3 communicators was 16 to 3072 processors and the total processors count was **18,576 CPUs**. The average CPU-time per time step was about 1.1 sec. The computation was performed on Ranger, TACC.

## Summary

- The potentially great advantage of the 2DD approach is in simulating very large scale problems. It addresses both:
- limited available memory per processor
  - tight communication between thousands of processors

### Team Members

- |   |   |   |
|---|---|---|
| <b>Brown University</b><br>Leopold Grinberg<br>Susanna Makela<br>George Karniadakis | <b>Children's Hospital Boston</b><br>Torner Anor<br>Joseph Madsen | <b>Northern Illinois University/ANL</b><br>Nicholas Karonis |
| <b>University of Pittsburgh</b>   | <b>Ben-Gurion University</b><br>Alexander Yakhot                  | <b>Texas Advanced Computing Center</b><br>John Cazes        |