Ural Federal University
named after the first President of Russia B.N.Yeltsin

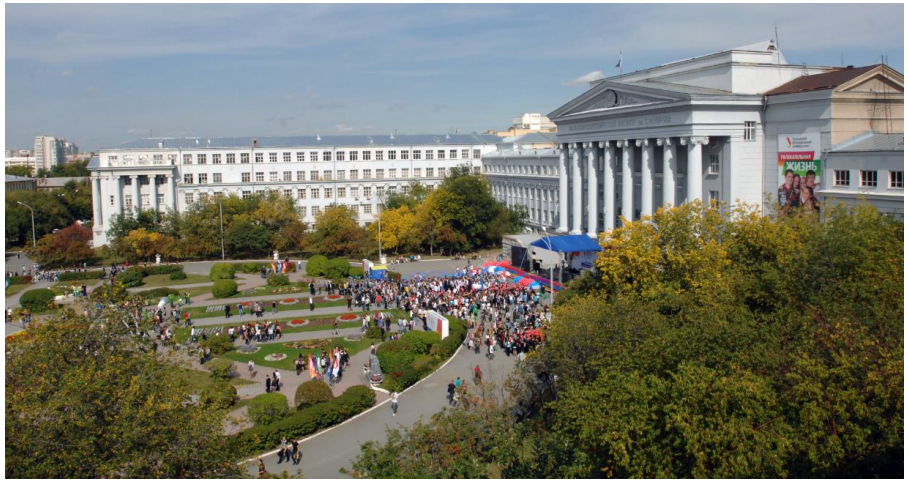
Teaching Heart Modeling and Simulation on Parallel Computing Systems

Andrey Sozykin, Mikhail Chernoskutov, Anton Koshelev, Vladimir Zverev, Konstantin Ushenin, and Olga Solovyova

Ural Federal University, Ekaterinburg, Russia
Institute of Mathematics and Mechanics Ural Brach of Russian Academy of Science
Institute of Immunology and Physiology Ural Brach of Russian Academy of Science

1st European Workshop on Parallel and Distributed Computing Education for Undergraduate Students (Euro-EDUPAR), Vienna, August 24, 2015

Ural Federal University



Ekaterinburg, Russia

One of the largest universities in Russia:

- 30 000 students from 60 countries
- 2000 faculty members
- 13 Institutes

IT and Computer Science Activities:

- 2014 ACM ICPC Global Programming Championship World Final
- Best Cisco Academy in Europe, 2013
- IT security Hackerdom team won the Nuit du Hack (Paris) CTF competition, 2012 and 2013

High Performance Computing Department

Jointly organized in 1997 by:

- Ural State University (now Ural Federal University)
- Institute of Mathematics and Mechanics UrB RAS

The department was created specifically to teach Parallel Computing

Computational resources:

- Supercomputer "URAN" of Institute of Mathematics and Mechanics UrB RAS, 240 TFLOPS, 370 GPU NVIDIA Tesla, 800 Intel Xeon CPU Cores
- Educational cluster of Ural Federal University, 24 TFlops, 12xIntel Xeon Phi, 12xGPU NVIDIA Tesla, 144 CPU Cores

HPC Courses

Parallel and distributed computing:

- Introductory course
- Architecture of parallel computing systems
- Multithreading programming
- Vectorization (SIMD)
- OpenMP
- MPI
- MapReduce

Advanced courses:

- GPU Programming
- Xeon Phi Programming

Target audience:

- Master students in Computer Science

The problem

Students think that parallel programming is easy!

One OpenMP directive:

- #pragma omp parallel for
- Similar OpenACC directive for GPU

Six MPI functions:

- MPI_Init, MPI_Finalize, MPI_Comm_size, MPI_Comm_rank, MPI_Send, MPI_Recv (or MPI_Bcast, MPI_Reduce)

By the end of the traditional HPC courses students:

- Know the basic parallel computing technologies
- Able to solve simple problems using HPC
- Not able to tackle real-world scientific problems

Programming technology progress

Serial Code

Single CPU Core Performance: ***1x***

```
.
.
.
.
for(int j=1;j<ny-1;j++) {
 for(int k=i1;k<nz-1;k++) {
  for(int i=1;i<nx-1;i++) {
   Anext[Index3D (nx,ny,i,j,k)] =
    (A0[Index3D (nx,ny,i,j,k+1)] +
       A0[Index3D (nx,ny,i,j,k-1)] +
       A0[Index3D (nx,ny,i,j+1,k)] +
       A0[Index3D (nx,ny,i,j-1,k)] +
       A0[Index3D (nx,ny,i+1,j,k)] +
       A0[Index3D (nx,ny,i-1,j,k)])*c1
       -A0[Index3D (nx,ny,i,j,k)]*c0;
   }
  }
}
```

Parallel Code for GPU

Add One OpenACC Directive
Tesla K40 Perf: ***13.6x***

```
.
.
.
#pragma acc parallel loop collapse(3)
for(int j=1;j<ny-1;j++) {
 for(int k=i1;k<nz-1;k++) {
  for(int i=1;i<nx-1;i++) {
   Anext[Index3D (nx,ny,i,j,k)] =
    (A0[Index3D (nx,ny,i,j,k+1)] +
       A0[Index3D (nx,ny,i,j,k-1)] +
       A0[Index3D (nx,ny,i,j+1,k)] +
       A0[Index3D (nx,ny,i,j-1,k)] +
       A0[Index3D (nx,ny,i+1,j,k)] +
       A0[Index3D (nx,ny,i-1,j,k)])*c1
       -A0[Index3D (nx,ny,i,j,k)]*c0;
   }
  }
}
```

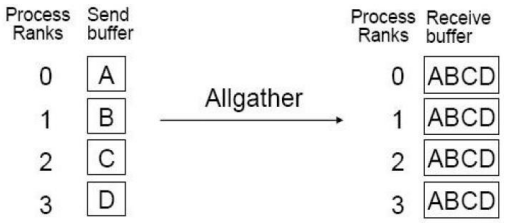
Dual socket E5-2698 v3 @2.3GHz (Haswell), 16 cores per socket, 256 GB memory, 1x Tesla K40
Benchmark: Parboil Stencil from University of Illinois with 1000 iterations
Source code for Parboil: http://impact.crhc.illinois.edu/Parboil/parboil.aspx

https://developer.nvidia.com/openacc

The problem

Introductory course exercises:

- Computing PI, Monte-Carlo Simulation, Matrix Multiplication, Mandelbrot Set, Conwey's Game of Life, k-Means Clustering, Canny Edge Detection, Jacoby Simulation
- Could be solved using simple techniques
- Short simulation time (seconds, minutes, seldom hours)



Real-world application requires much more:

- Advanced MPI: non-blocking point-to-point communication, advanced collective communication, overlapping computation with sending/receiving data, virtual topologies
- Hybrid parallelization: MPI+OpenMP+SIMD (+GPU, +Xeon Phi, +FPGA)
- Optimal domain decomposition: 1D, 2D, 3D, etc.
- Data locality optimizations

First solution: advanced exercises

Include advanced exercises into the introductory courses:

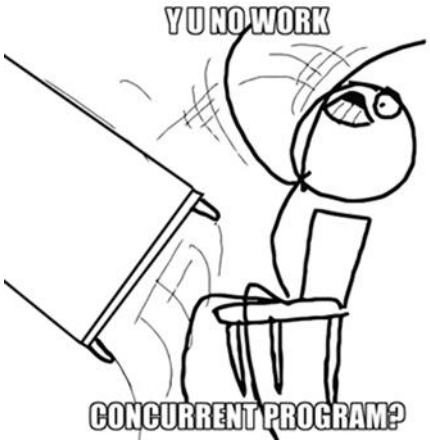
- Optional
- Mandatary

Negative results:

- Almost no one chose the optional advanced exercises
- Almost everyone failed the mandatary advanced exercises

Reasons of failure:

- Technical difficulties: advanced exercises requires much more studying and practicing than the basic ones
- Lack of students' motivation
- Lack of math or domain area knowledge



Second solution: real-world tasks

Give the students big real-world scientific tasks to solve

High motivation

- Students solve real problems instead of boring exercises
- Parallel programing is studied not for its own sake, but as a useful tool for research

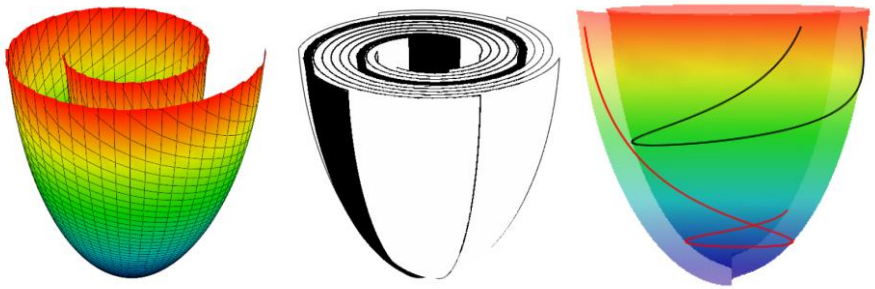
Teach parallel computing together with domain science:

- Introductory course on parallel computing
- Theoretical course on domain science
- Advanced course on using parallel computing in the domain science
- Big projects in domain science (real-world problems)

Heart modeling

Cooperation with the laboratory "Modeling of living systems on supercomputers"

- The laboratory develops various heart models on different levels (cell – tissue - organ)
- Distinguished feature: anatomical models of heart left ventricle (LV) with an analytic description of its shape and fibre orientation
- Simulation of 1 heart bit (1 second of physical time) requires 3 day on personal computer



Existing theoretical course:

- Simulation of Living Systems
- Olga E. Solovyova, Head of the laboratory

Proposed course:

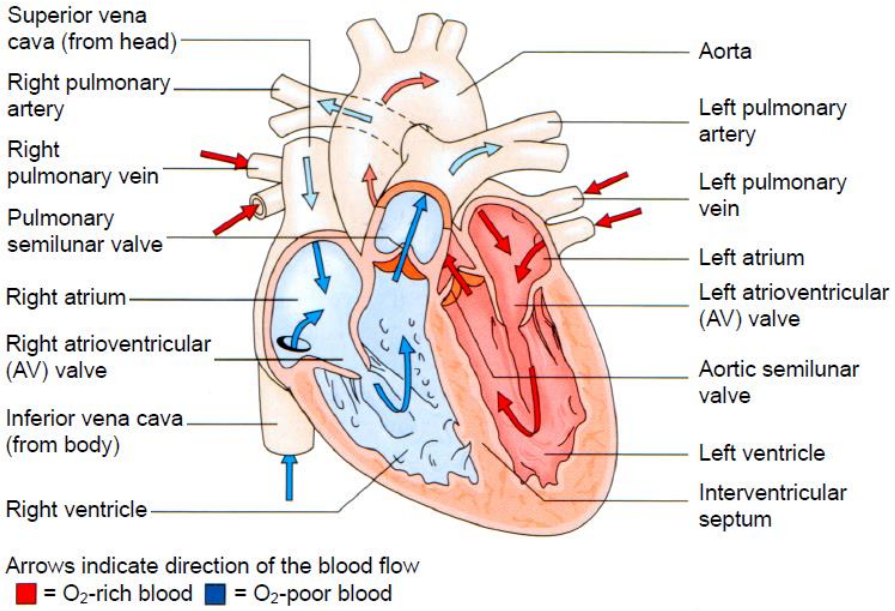
- Modeling Heart Dynamics on Parallel Computing Systems

Heart modeling on PCS course

Simulation of electromechanical function of the mammal heart LV on parallel computing systems

Lecture topics:

- Structure and physiology of the mammal heart
- Various methods and models of heart LV digital representation
- Numerical methods: mass-spring system, finite element method (FEM)
- Simulation of the heart LV electrodiusion
- Simulation of heart LV mechanical contraction

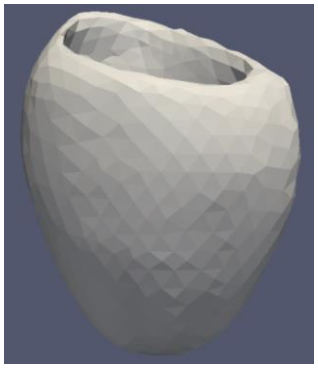


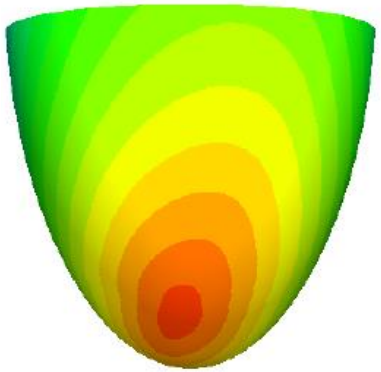
L. Sherwood, Human physiology : from cells to systems. Australia ; United Kingdom: Thomson Brooks/Cole, 2004.

Heart modeling on PCS course

Workshop topics:

- Building the model of LV using the tetrahedral 3D-network with the help of “GMsh” library
- Studying the FEM implementation for MPI clusters from the “FEniCS” library
- Optimization of the FEM application on parallel computing systems
- Simulation of the muscle cube on MPI cluster using the “FEniCS” library
- Visualization of simulation results using Paraview

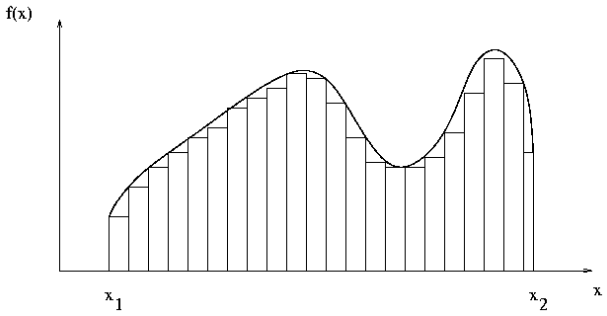




Parallel Numerical Methods Course

Compute-intensive tasks:

- Linear algebra
- Numerical integration and differentiation
- Numerical solving of mathematical physics equations
- Finite element method



Data-intensive tasks:

- Processing of large graphs

Techniques of effective parallelization:

- Efficient cache usage
- Data partitioning among the processes
- Reducing the inter-process communication

Parallel Numerical Methods Course

Course mission:

- Teach students how to choose algorithmic building blocks for scalable parallel application
- Explain to students how to avoid performance degradation of parallel programs

Home programming assignments requirements:

- Implement several version of parallel algorithms
- Carry out research on their performants and scalability
- Choose the most efficient algorithm and implementation

Positive outcome:

- Students started to learn advanced parallel computing techniques to create efficient implementation for home programming assignments

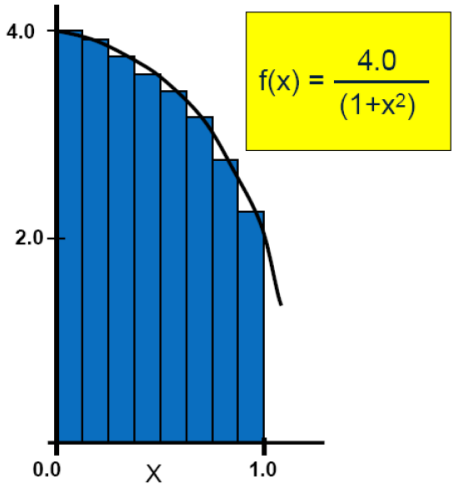
Parallel Numerical Methods Course

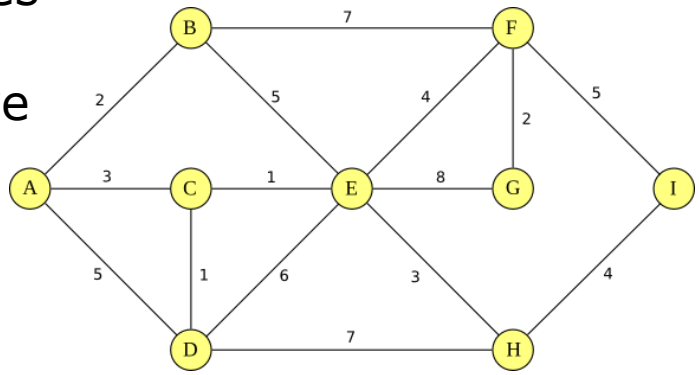
First programming assignment:

- Numerical integration
- Point-to-point communications (MPI_Send, MPI_Recv)

Advanced programming assignment:

- Single source shortest paths (SSSP) graph problem
- Point-to-point communications provides small speedup
- Students investigates various collective communications
- Efficient solution requires usage of MPI_Allgather (despite the big overhead of the function)



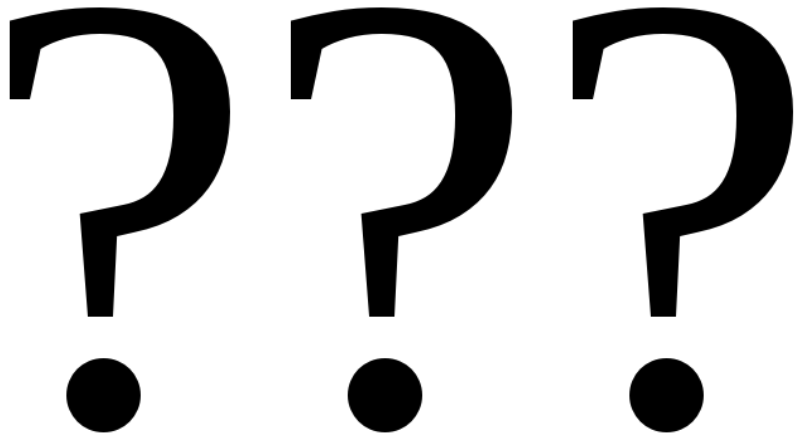


Modules of the course track

Three modules:

- Parallel and distributed computing
- Parallel numerical methods
- Heart modeling

Will the students be able to solve real-world scientific problems on parallel computer systems?



More time with teachers!

Three modules:

- Parallel and distributed computing
- Parallel numerical methods
- Heart modeling

Will the students be able to solve real-world scientific problems on parallel computer systems?

No!

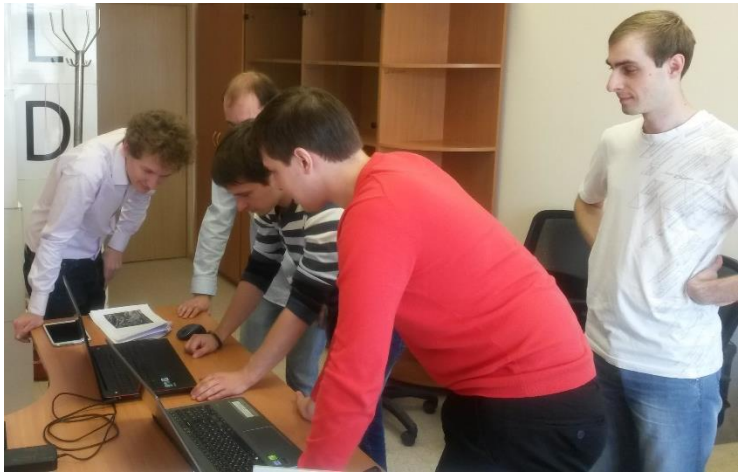
For complex projects students need much more assistance from the teachers than for introductory exercises:

- Complex projects require a broad knowledge of various parallel programming technologies
- Few information in the Internet about solving complex tasks on the parallel computing systems

Science Hackathon

Course aim:

- Provide the students with opportunity to work together with the tutor for a long time



Course format:

- Conducted on Saturdays
- Lasts all the day
- Group of students works on one project together with a tutor to produce usable software
- Students choose the topic of the projects

Examples of projects:

- Using auto-vectorization for Xeon Phi
- Evaluating Intel VML performance on Xeon Phi
- "FEniCS" library
- Simulating the heart muscle cube using FEM and mass-spring methods

Science Hackathon

Benefits for students:

- Tutor quickly corrects mistakes in the students code
- Tutor suggests the performance optimization techniques suitable for the projects
- Tutor provides quick feedback on the projects and further guidance

Positive outcome:

- Significant speedup of students' projects

Additional positive outcome for students:

- Team software development
- Problem solving skills
- Professional confidence
- Leadership (some students work as tutors with other students)

Heart modeling course track

No	Course name	Credits
1	***Parallel and distributed computing module***	
1.1	Parallel and distributed computing	4
1.2	GPU programming	2
1.3	Xeon Phi programming	2
2	***Numerical methods module***	
2.1	Parallel numerical methods	2
2.2	Science hackathon	4
3	***Heart modeling module***	
3.1	Simulation of living systems	4
3.2	Modeling heart dynamics on parallel computing systems	2

Heart modeling course track

Target audience

- Master students in Computer Science
- Bachelor Computer Science students and Master students in other majors are also eligible

Every course in the track is elective

- Students are not required to take all the courses
- Each student can choose courses according to his/he individual needs

Typical enrollment

- 10-15 students

Recommended sequence of study

Semester	Course name
1	Parallel and distributed computing Simulation of living systems
2	GPU programming Parallel numerical methods Modeling heart dynamics on parallel computing systems
3	Xeon Phi programming Science Hackathon

Positive results

Key success points

- Interesting real-world scientific tasks to solve
- The tasks simulation on personal computers takes long time
- Knowledge in the three area: parallel computing, numerical methods, and domain science
- Coherent courses programs
- Large amount of time with teachers/tutors

The reason of success

- High students' motivation
- Long simulation time force students to learn advanced parallel computing techniques in order to decrease it
- Students work in groups and share their knowledge
- Teachers/tutors provide quick feedback and project guidance

Student's success story

Second year Computer Science Master student

Project topics

- Building the analytical model of LV shape using the ultrasonic data (personalized model)
- Electrophysiological simulation of LV with the personalized analytical model

Student's achievements

- Ultrasonic data was imported into the LV simulation software
- Performance of the LV electrophysiological simulation software was improved
- Work with other students during Science hackathon

Student obtained research position at the laboratory "Modeling of living systems on supercomputers"

Future works

English version of the track courses

Short-term English-language internship program (1-3 months)

- Parallel computing internship program is already exists http://iie.urfu.ru/en/article/student-internships
- Include heart modeling courses into the internship program

MOOC “Modeling heart dynamics on parallel computing systems” on EdX.org

- Ural Federal University is the first Russian university that signed an agreement with EdX.org

Interdisciplinary courses in different domain areas:

- Molecular dynamics simulation
- Astronomy

Thank you!

Contacts:
Andrey Sozykin, Andrej.Sozykin@urfu.ru
Head of High Performance Computing Technologies Department
Institute of Mathematics and Computer Sciences
Ural Federal University
Ekaterinburg, Russia

Website
www.urfu.ru/en

Non Computer Science Students

Master students with non Computer Science Bachelor Degree are highly welcomed

- Physics, mathematics, biology
- Have broad domain knowledge
- Lack of the software development skills

Additional courses

- Scientific computing in C
- Software performance optimization

Best group of students for successful project

- One or two students with Computer Science Bachelor Degree
- One student with Domain Science Bachelor Degree