

Solution Verification on a Grid

Christophe Picard, Venkat Subramaniam & Marc Garbey

Department of Computer Science
University of Houston, Houston, Texas

Agenda

- **Problem**
- Computational Complexity
- Parallelization Issues and Solutions
- System Architecture
- Speedup Expectations
- Conclusion



Motivations

Solution verification is becoming a critical issue because :

- The complexity of the problem overwhelm the domain of validity of the mathematics.
- Large scale hardware/software system are unreliable.
- Decisions are based on numerical simulation: nuclear waste, defense program, safety of the space shuttle, optimum design of civil air planes, etc...



Solution Verification in CFD

- A posteriori Estimators in the finite elements framework.
- Richardson extrapolation.
- Bayesian theory.
- Our method : treat the solution verification as an optimum design of the extrapolation problem based on few coarse grid solutions.



Least Square Extrapolation method

- Design of a method to deal with some of the deficiencies of RE.
- Easy to implement as a stand alone procedure.
- Can be used with existing computational software.
- Can be coupled with existing a posteriori error estimator method, but may work in much more general situations.



LSE for Navier-Stokes

- Given
 - a non-linear operator N
 - the numerical solutions u_i on several grids G_i where $i=1\dots3$
 - a fine grid G_0 that (supposedly) solves all scales
- Let U_i be the interpolation of u_i on this fine grid.
- LSE : *Find α, β such that*
$$N[\alpha U^1 + \beta U^2 + (1 - \alpha - \beta)U^3] \text{ is minimum in } L_2 \text{ norm}$$
- Use very few time steps as to filter out spurious frequency components of the projection.



The optimization problem

- This is a minimization problem of unknowns α and β .
- This optimization problem is characterized by :
 - The choice of the functional space of α and β .
 - The level of accuracy desired.
 - The multi-scales of the problem.



Agenda

- Problem
- **Computational Complexity**
- Parallelization Issues and Solutions
- System Architecture
- Speedup Expectations
- Conclusion



Numerical Algorithm

- Step 1 - Solve the problem with 2 or 3 different meshes.
- Step 2 - Interpolate these solutions on a fine grid G_0 , and form linear combinations with an initial guess on the weight functions.
- Step 3 – Generate a surface response for each point in the optimum design space :
 - Step 3.1 - Apply few relaxations steps on the fine grid.
 - Step 3.2 - Compute residual.



Sequential Programs

- Step 1: 3 executions of a Native Computational code : Adina - Nast2d – Nast3d.
- Step 2: C++ code to read files, compute interpolation and generate new input files.
- Step 3.1: $N_\alpha \times N_\beta$ executions of a Native Computational code.
- Step 3.2: C++ code to read files and compute residual and error of solutions.



Main Idea

- Step 3 is the real time consuming step.
- It has a strong asynchronous parallelism.
- Ideal application for the grid.



Agenda

- Problem
- Computational Complexity
- **Parallelization Issues and Solutions**
- System Architecture
- Speedup Expectations
- Conclusion



Network Architecture

- We want to design a portable method that can be used with all the resources to our disposition.
- Architecture /OS are heterogenous
 - Dual AMD Athlon cluster running linux OS
 - 8 way Opteron running linux OS
 - G5 cluster running OSX
 - Dual opteron running Windows XP/2002
 - Pentium 4 running Windows XP/2002
 - Pentium 4 running Linux



Issue generated by the grid

- Native code is driving which architectures we can use.
- Communication interfaces should be portable.
- Each job is launched using a script.
- The LSE post-processing should be computationnaly efficient, and portable.



Asynchronous Parallelization

- Only two steps can be parallelized:
 - Step 1, since the three solutions from different grids.
 - Step 3, since it has there is no need of information exchanges during the solving process.
- Step 1 will not be parallelized.
 - The solution on a fine coarse grid will be used as an initial solution for the next grid.
- Step 3 will be parallelized.



Data Transfer

- The way to communicate between the different steps is done using files:
 - We cannot embed the native CFD code into our code.
 - Plotting the final results are done using external software.
 - Data generated by our code cannot be stored in memory without thrashing the disk.
- But create files on the master node is not efficient: each slave have a different input file since they are processing independent problems.



Dealing with data transfer

- Size of the file is critical for a slow network. We have to revisit the distribution of the work between the master and the slaves.
- We trade numerical redundancy for network bandwidth.
- Each slave node is interpolating the coarse grids solutions on the fine grid once and for all.



Agenda

- Problem
- Computational Complexity
- Parallelization Issues and Solutions
- **System Architecture**
- Speedup Expectations
- Conclusion



Mapping on to Grid



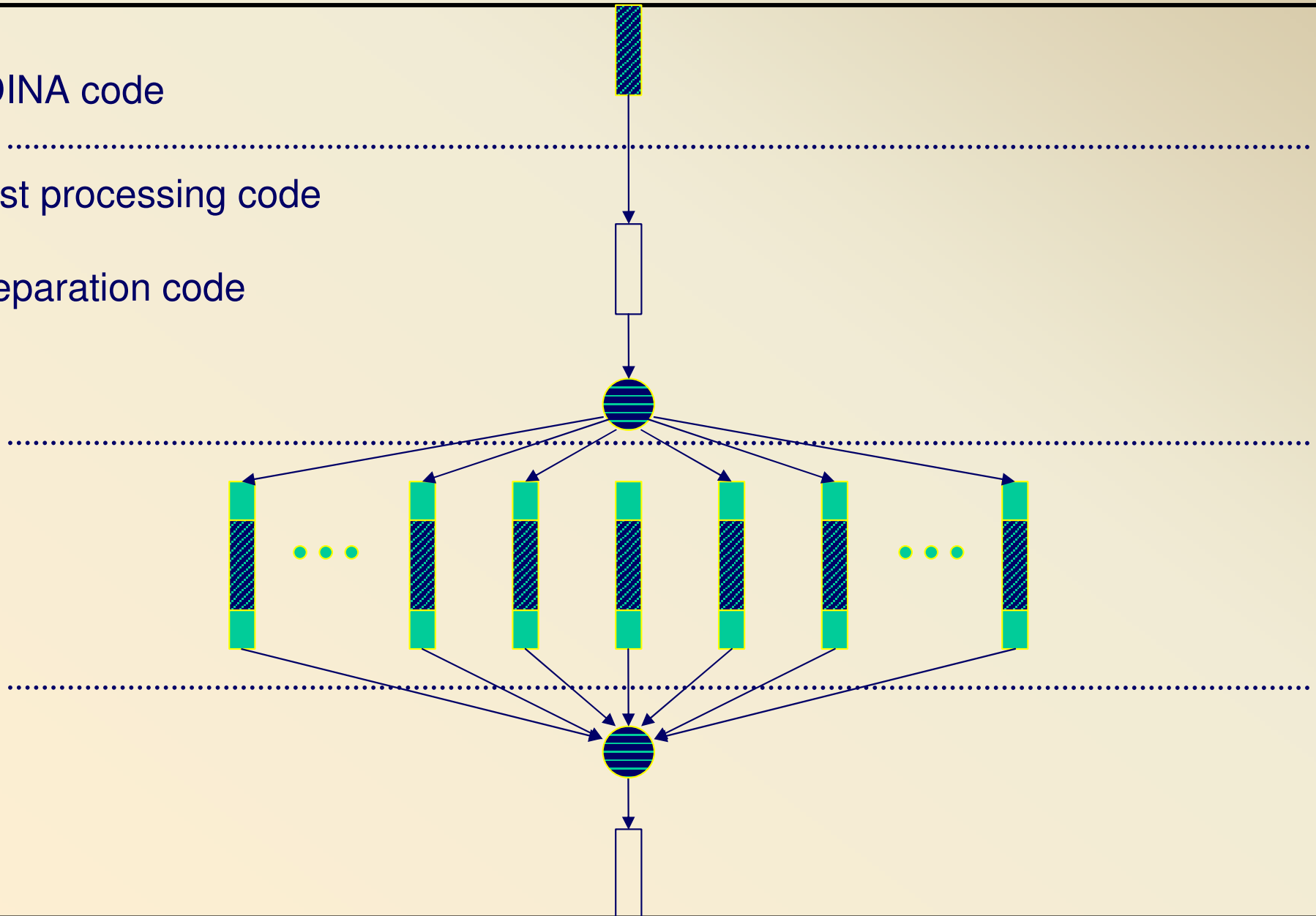
ADINA code



Post processing code



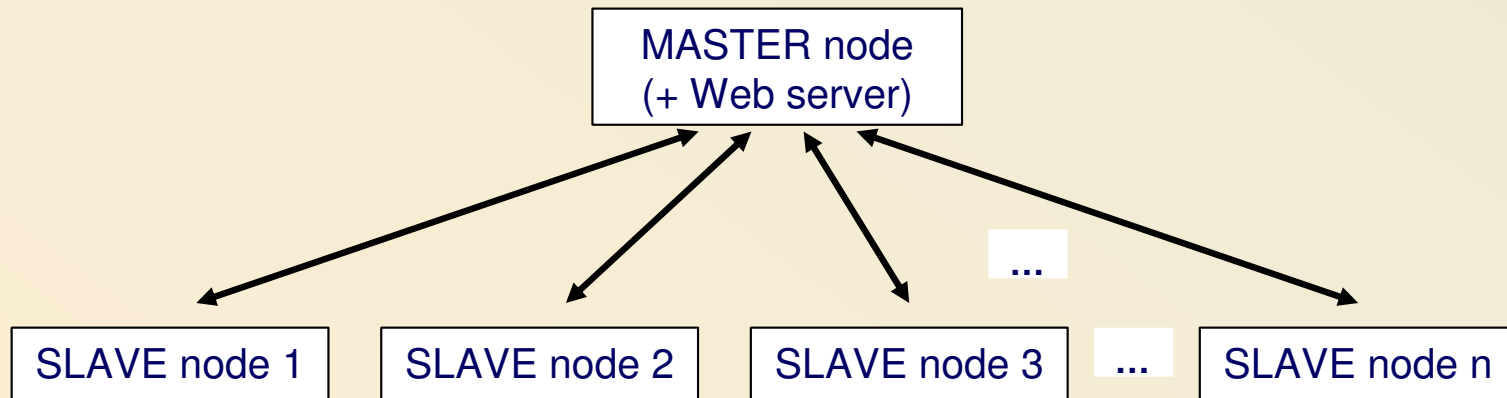
Preparation code



Node distribution

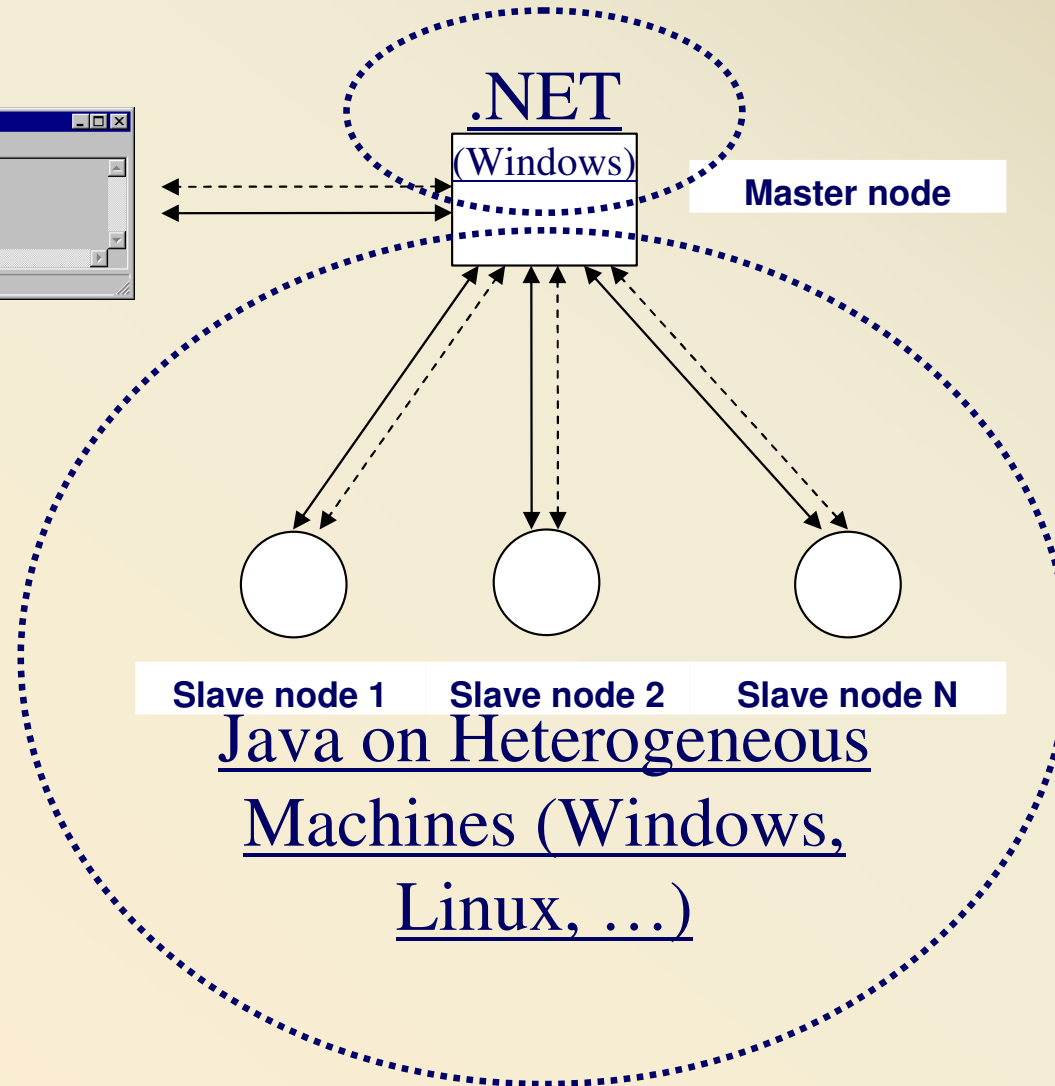
Dynamic scheduling instead of static scheduling

- Bird-feeding-chicks model
- Any available node is given task
- Leads to maximum utilization

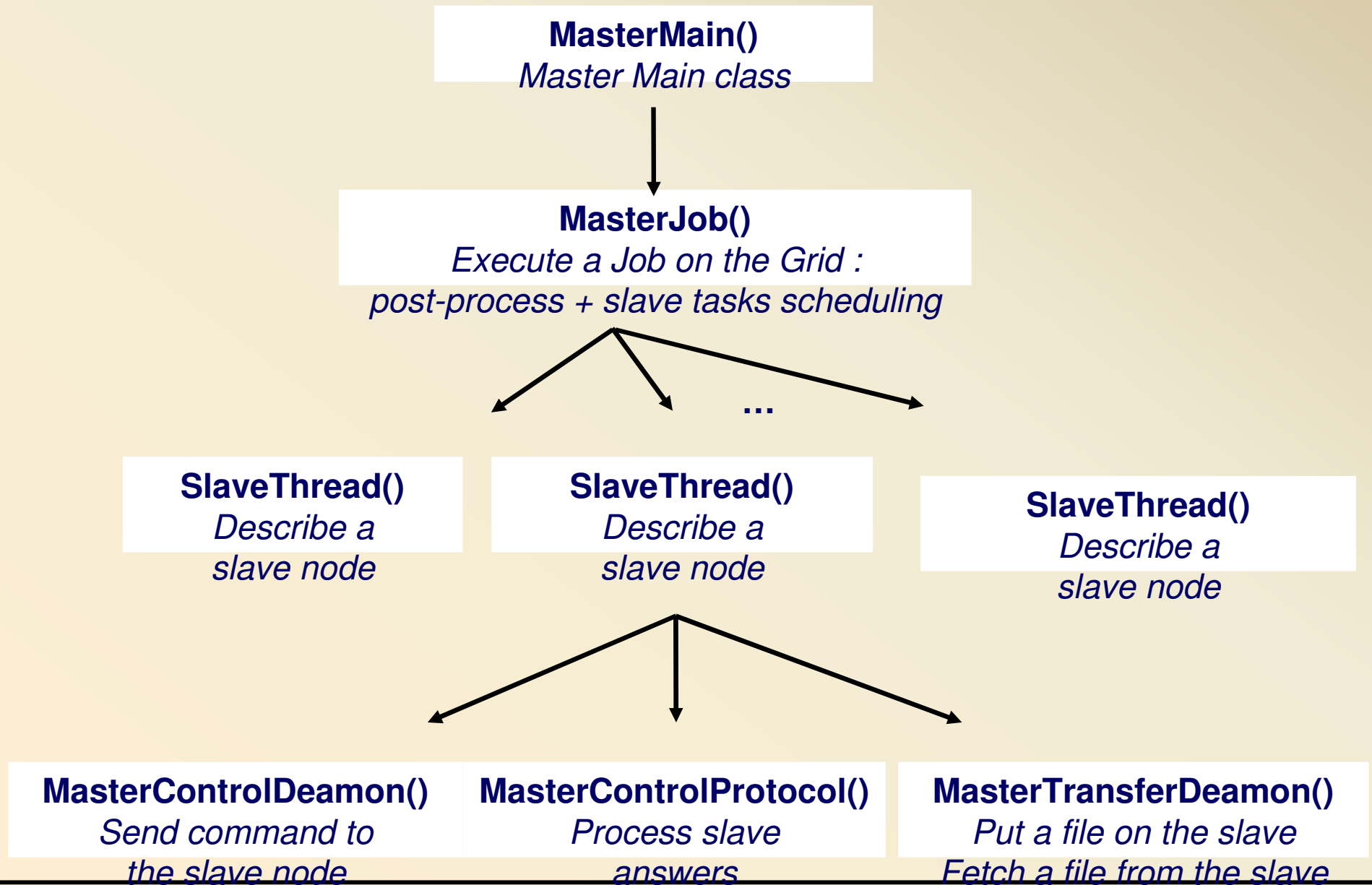


Software Technology

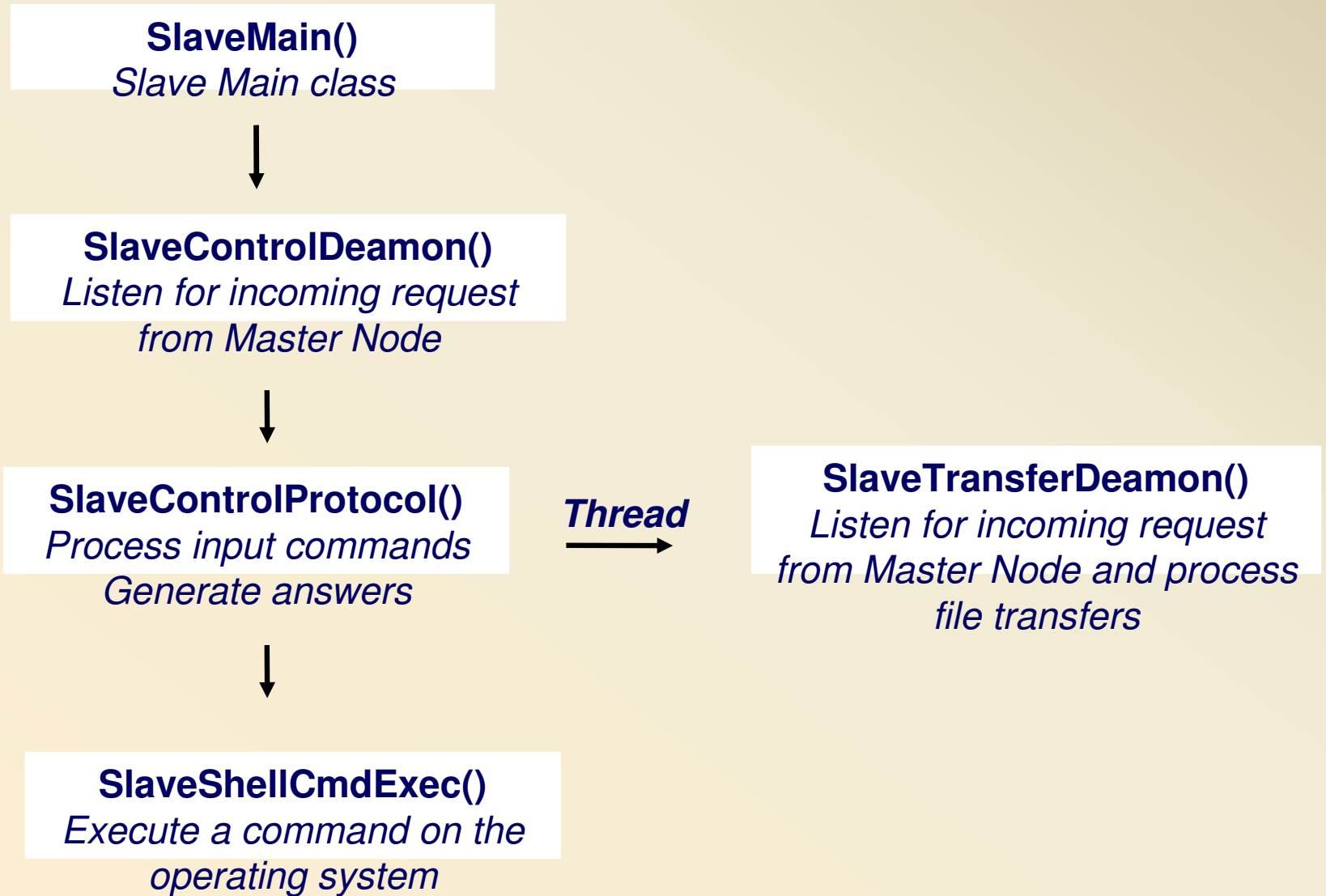
Web
Browser
(Windows,
Linux, Mac,...)



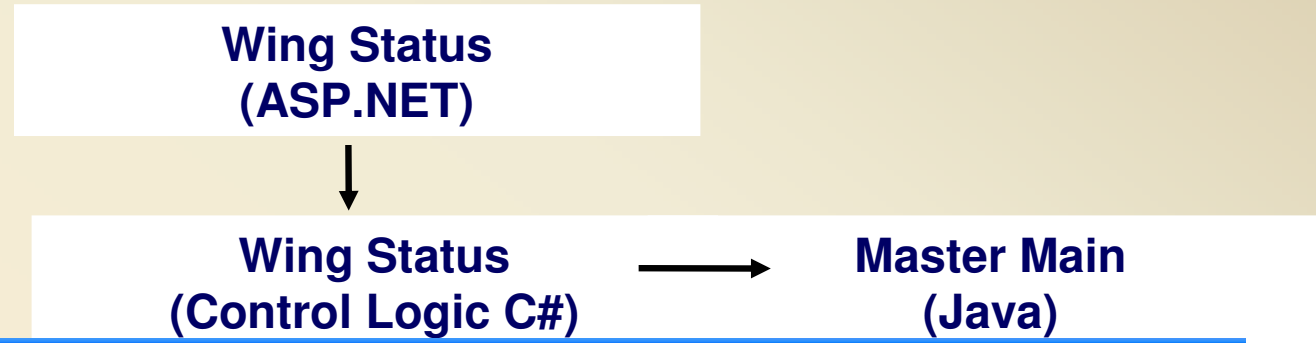
Master Node Java Module



Slave Node Java Module



Master Node .NET Module




nodesstatus - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites Media

Address <http://agility2/Wing/nodesstatus.aspx>

Wing NodesStatus | Start New Job | Job Status 

Nodes Status

Name	Status
Node 0	Busy
Node 1	Busy
Node 2	Busy
Node 3	Busy
Node 4	Free
Node 5	Busy
Node 6	Busy

Agenda

- Problem
- Computational Complexity
- Parallelization Issues and Solutions
- System Architecture
- **Speedup Expectations**
- Conclusion



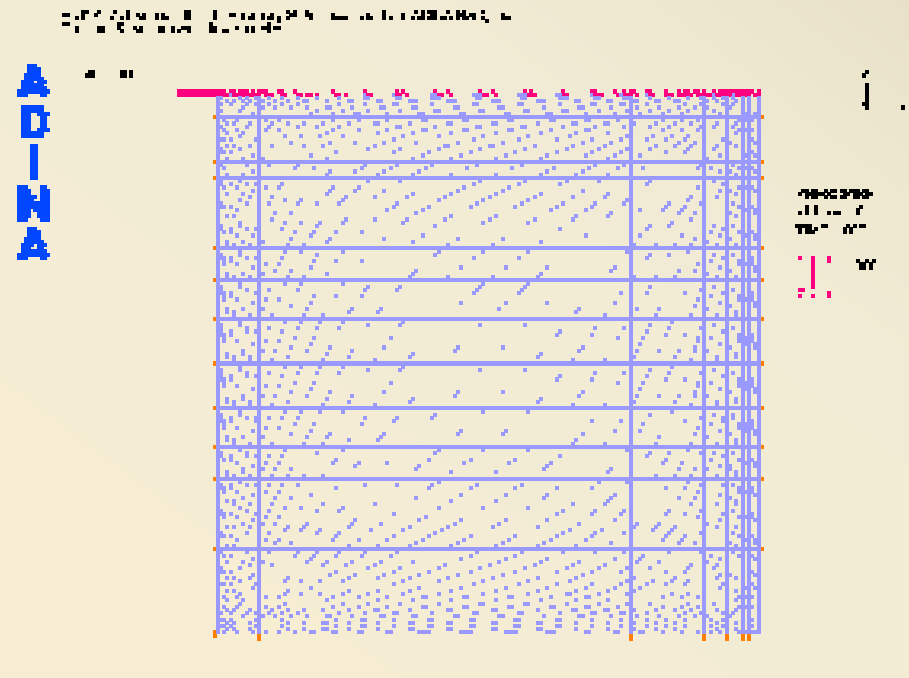
A example with ADINA

Driven cavity flow test case :

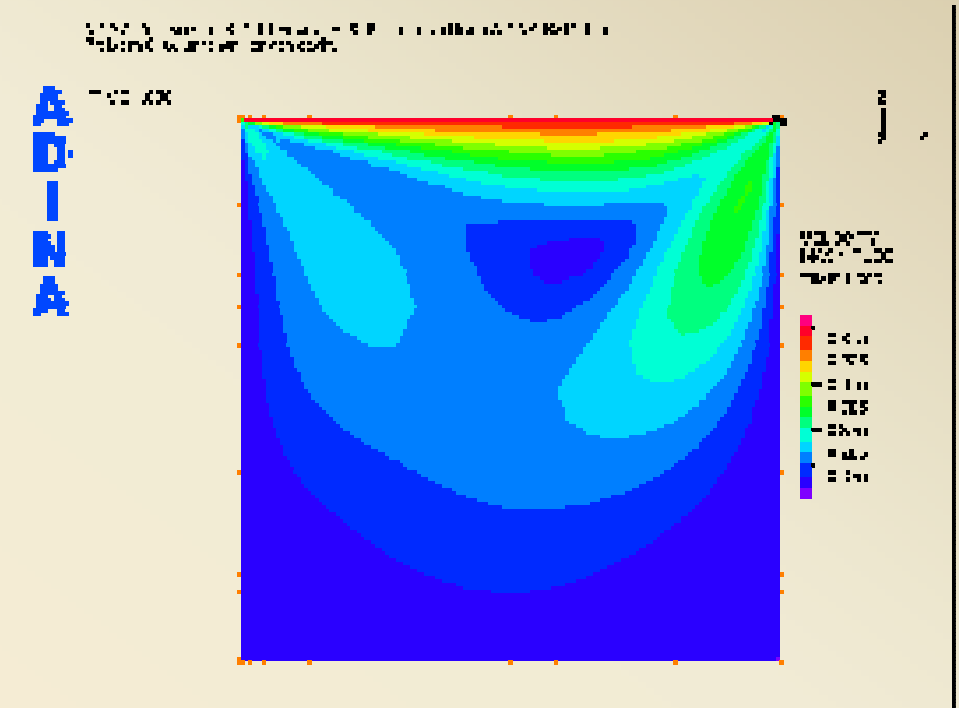
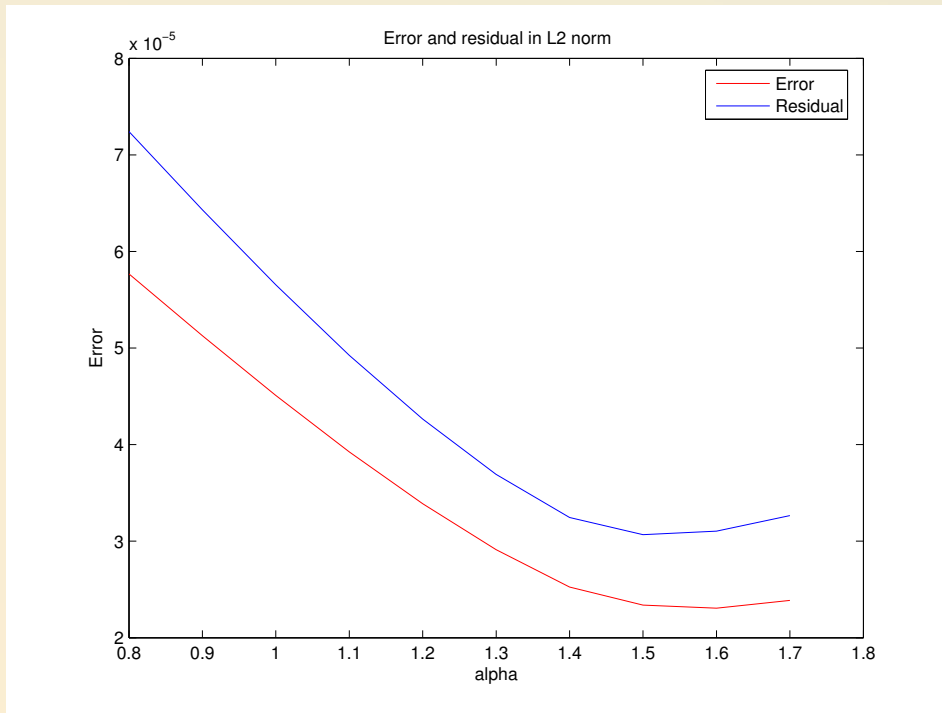
Reynolds : 100

Grids : G1 = 30x30, G2 = 40x40, G3 = 50x50

G0 = 100x100



Results for driven cavity flow



Overall time 12 minutes:

80s for the Step 1, 16s for the Step 2 620s for the Step 3

Expecting time : 62s for Step 3, giving an overall time of 2'30



Agenda

- Problem
- Computational Complexity
- Parallelization Issues and Solutions
- System Architecture
- Speedup Expectations
- **Conclusion**



Conclusion and Objective

- Different pieces needs to be put together
- Extends the functional space for the surface response
- Choose more software to interface with
 - NAST3D, NAST2D

