Grid-Aware Numerical Libraries

To enable the use of the Grid as a seamless computing environment

Early Focus of GrADS

- **To create an execution environment that supports reliable performance for numerical libraries on Grid computing platforms.**
- **Interested in structuring and optimizing an application and its environment for execution on target computing platforms whose nature is not known until just before run time.**

Milestones from GrADS

- **Library interface for numerical libraries**
- **Infrastructure for defining and validating performance contracts**
- **Adaptive GrADS runtime interface for scheduling and forecasting systems in the Grid**

ScaLAPACK

ScaLAPACK
A Software Library for Linear Algebra Computations on Distributed-Memory

- **ScaLAPACK is a portable distributed memory numerical library**
- **Complete numerical library for dense matrix computations**
- **Designed for distributed parallel computing (MPP & Clusters) using MPI**
- **One of the first math software packages to do this**
- **Numerical software that will work on a heterogeneous platform**
- **In use today by IBM, HP-Convex, Fujitsu, NEC, Sun, SGI, Cray, NAG, IMSL, …**
  - **Tailor performance & provide support**

ScaLAPACK Demo

- **Implement a version of a ScaLAPACK library routine that runs on the Grid.**
  - **Make use of resources at the user's disposal**
  - **Provide the best time to solution**
  - **Proceed without the user's involvement**
- **Make as few changes as possible to the numerical software.**
- **Assumption is that the user is already "Grid enabled" and runs a program that contacts the execution environment to determine where the execution should take place**

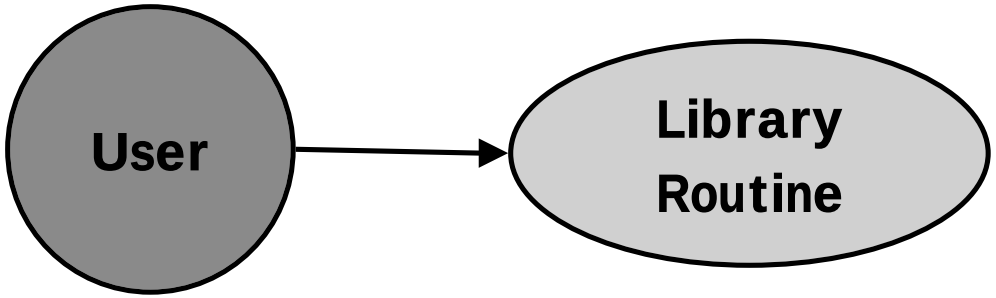
How ScaLAPACK Works

- **To use ScaLAPACK a user must:**
  - **Download the package and auxiliary packages to the machines**
  - **Write a SPMD program which**
    - **Sets up the logical process grid**
    - **Places the data on the logical process grid**
    - **Calls the library routine in a SPMD fashion**
    - **Collects the solution after the library routine finishes**
  - **The user must allocate the processors and decide the number of processors the application will run on**
  - **The user must start the application**
    - **"mpirun –np *N* user_app"**
      - The number of processors is fixed at run time
  - **Upon completion, return the processors to the pool of**

GrADS Numerical Library

- **Want to relieve the user of some of the tasks**
- **Make decisions on which machines to use based on the user's problem and the state of the system**
  - **Optimize for the best time to solution**
  - **Distribute the data on the processors and collections of results**
  - **Start the SPMD library routine on all the platforms**
  - **Check to see if the computation is proceeding as planned**
    - **If not perhaps migrate application**

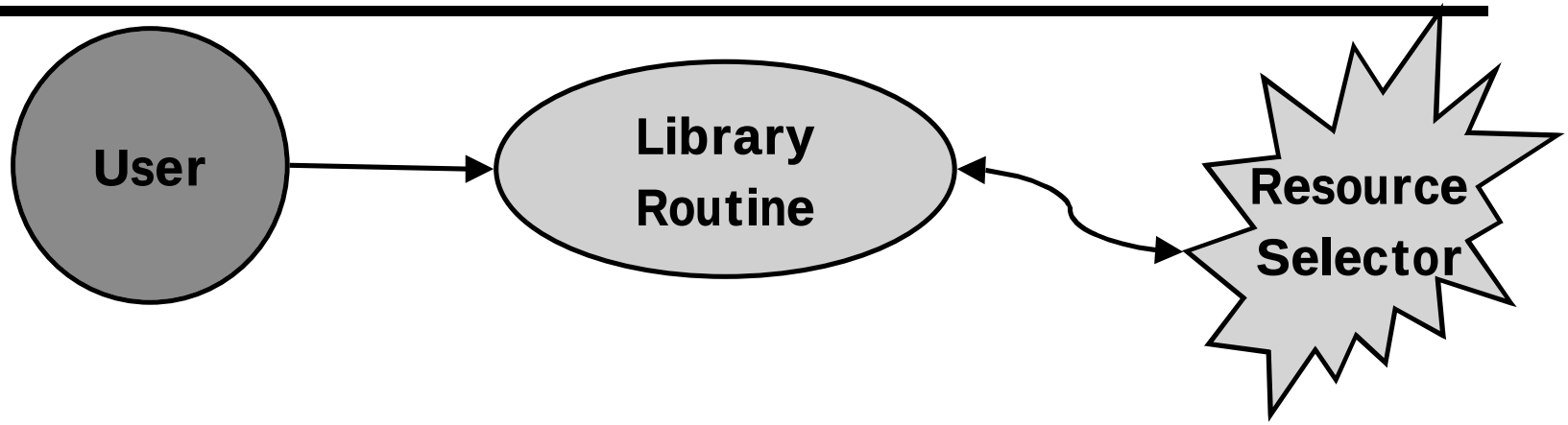
GrADS Library Sequence



User makes a sequential call
to a numerical library routine.
The Library Routine has “crafted code”
which invokes other components.

Assumption is that Autopilot Manager has been started and Globus is there.

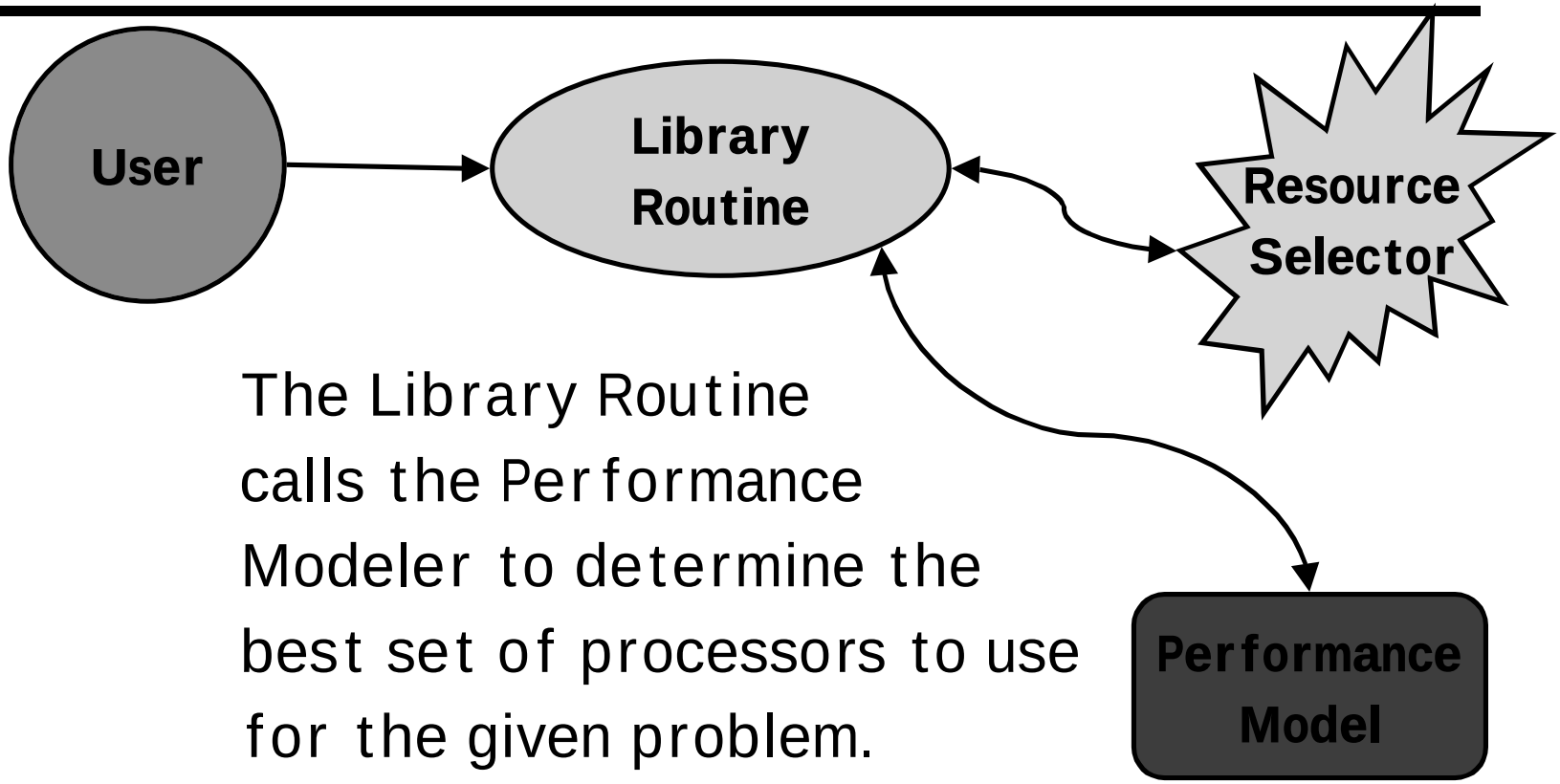
GrADS Library Sequence



The Library Routine calls a grid based routine to determine which resources are possible for use.

The Resource Selector returns a "bag of processors" (coarse grid) that are available.

GrADS Library Sequence

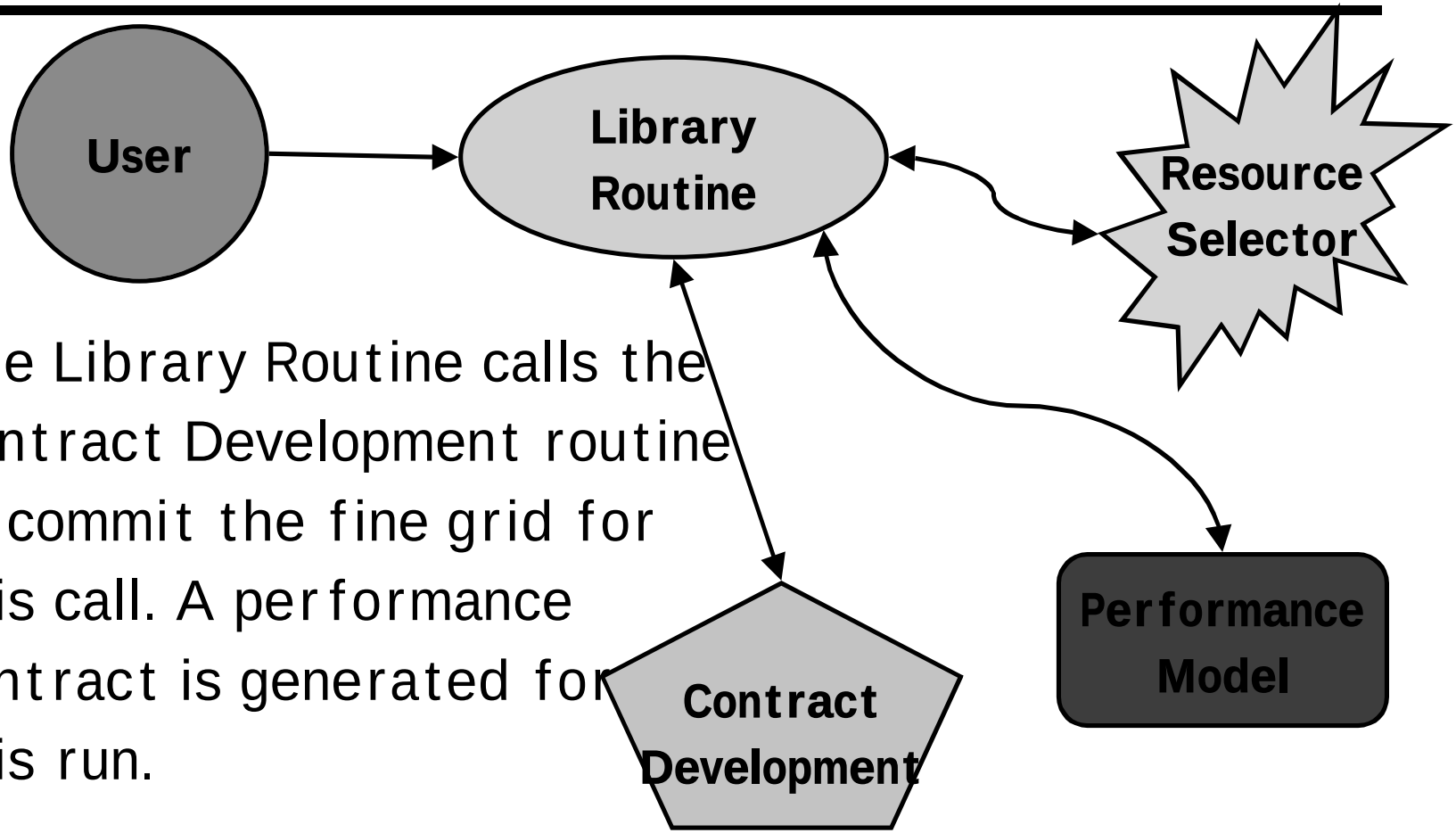


The Library Routine calls the Performance Modeler to determine the best set of processors to use for the given problem.

May be done by evaluating a formula or running a simulation.

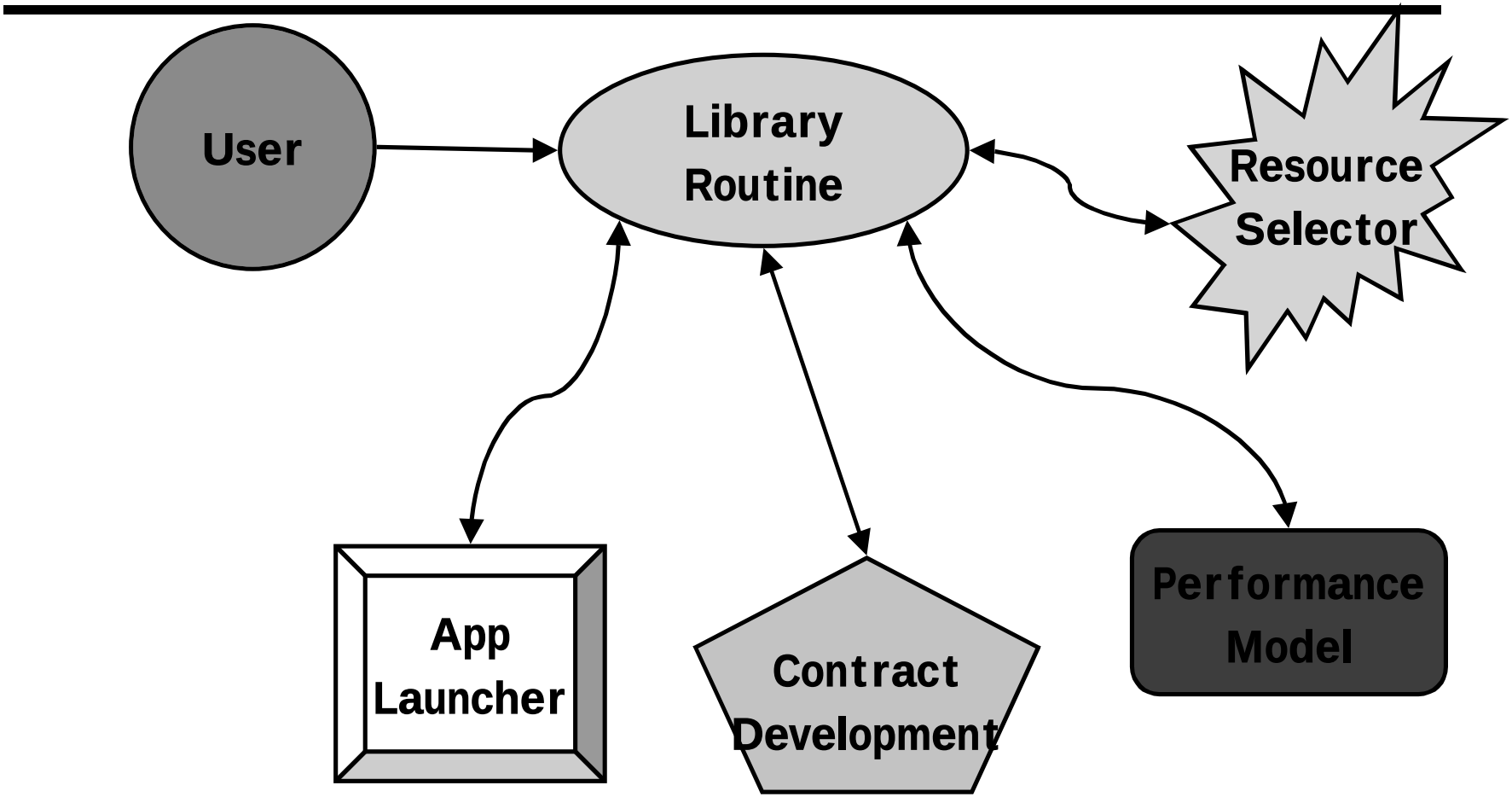
May assign a number of processes to a processor. At this point have a fine grid.

GrADS Library Sequence



The Library Routine calls the Contract Development routine to commit the fine grid for this call. A performance contract is generated for this run.

GrADS Library Sequence



“mpirun –machinefile fine_grid grid_linear_solve”

Grid Environment for this Experiment

4 cliques
43 boxes

UIUC
amajor-dmajor
PII 266Mhz 100Mb sw
opus0, opus13-opus16
PII 450Mhz Myrinet

U Tennessee
torc0-torc8
Dual PIII 550Mhz
100 Mb sw

UCSD
Quidam, Mystere, Soleil
PII 400Mhz 100Mb sw
Dralion, Nouba
PIII 450Mhz 100Mb sw

U Tennessee
cypher01 – cypher16
dual PIII 500Mhz
1 Gb sw

ferret.usc.edu
64 Proc SGI 32-250mhz
32-195Mhz
jupiter.isi.edu
10 Proc SGI
lunar.uits.indiana.edu

Components Used

- **Globus version 1.1.3**
- **Autopilot version 2.3**
- **NWS version 2.0.pre2**
- **MPICH-G version 1.1.2**
- **ScaLAPACK version 1.6**
- **ATLAS/BLAS version 3.0.2**
- **BLACS version 1.1**
- **PAPI version 1.1.5**
- **GrADS' "Crafted code"**

Heterogeneous Grid

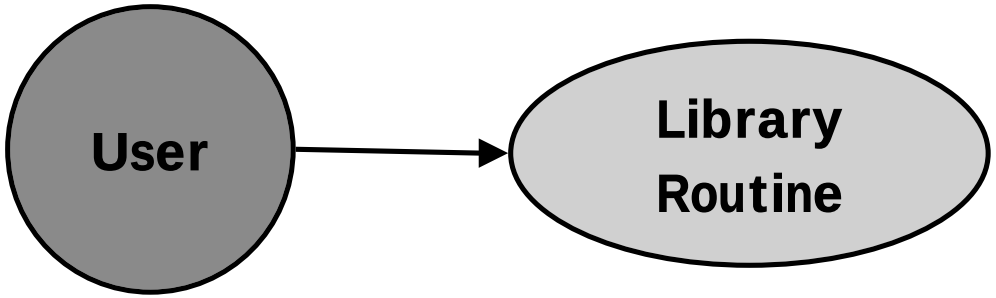
	TORC	**CYPHER**	**OPUS**
Type	Cluster 8 Dual Pentium III	Cluster 16 Dual Pentium III	Cluster 8 Pentium II
OS	Red Hat Linux 2.2.15 SMP	Debian Linux 2.2.17 SMP	Red Hat Linux 2.2.16
Memory	512 MB	512 MB	128 or 256 MB
CPU speed	550 MHz	500 MHz	265 – 448 MHz
Network	Fast Ethernet (100 Mbit/s) (3Com 3C905B) and switch (BayStack 350T) with 16 ports	Gigabit Ethernet (SK-9843) and switch (Foundry FastIron II) with 24 ports	IP over Myrinet (LANai 4.3) + Fast Ethernet (3Com 3C905B) and switch (M2M-SW16 & Cisco Catalyst 2924 XL) with 16 ports each

The Grads_lib_linear_solve Routine Performs the Following Operations:

- **Gets information on the user's problem**
- **Creates the "coarse grid" of processors and their NWS statistics by calling the resource selector.**
- **Refines the "coarse grid" into a "fine grid" by calling the performance modeler.**
- **Invokes the contract developer to commit the resources in the "fine grid" for the problem.**

  Repeat Steps 2-4 until the "fine grid" is committed for the problem.
- **Launches the application to execute on the committed "fine grid".**

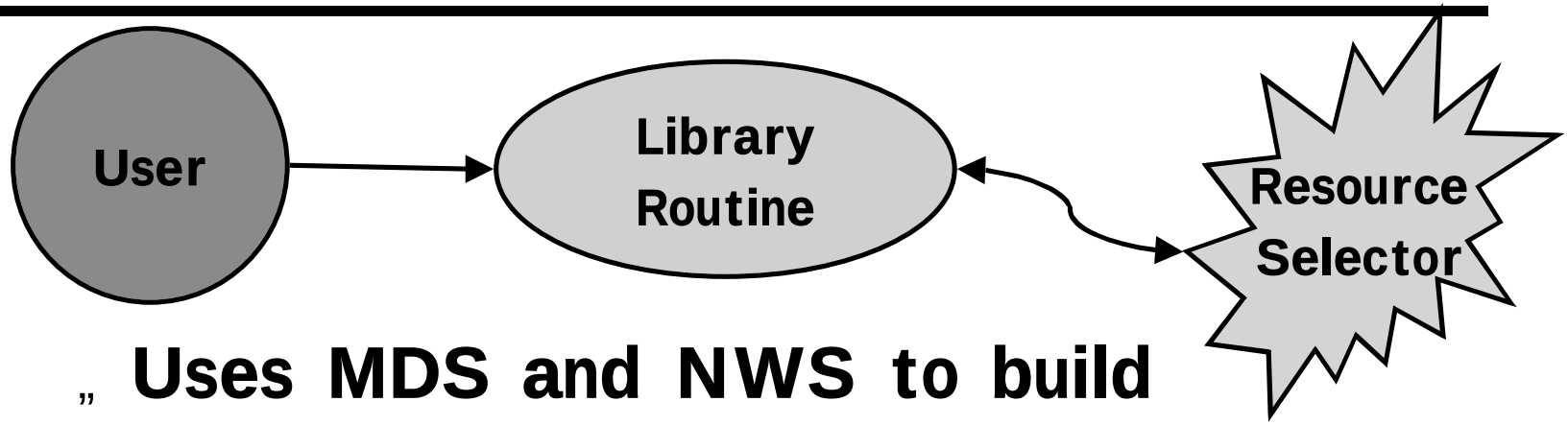
GrADS Library Sequence



- **Has "crafted code" to make things work correctly and together.**

Assumptions:
Autopilot Manager has been started
and
Globus is there.

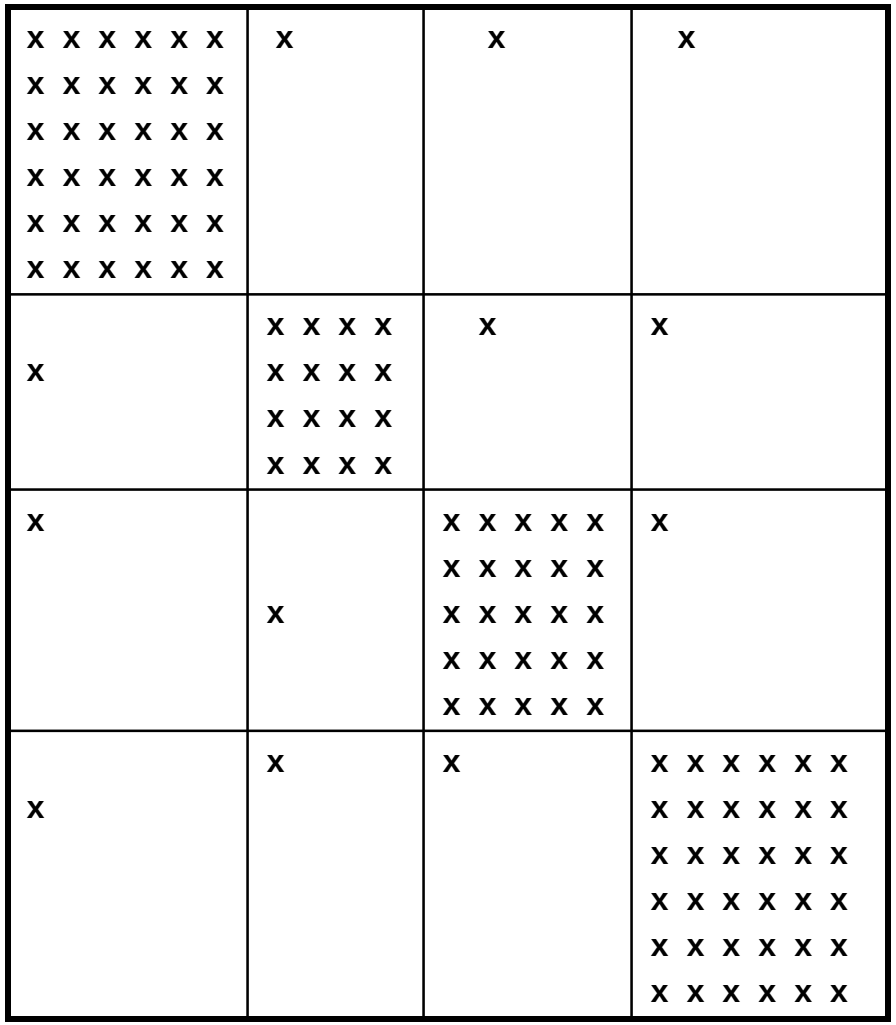
Resource Selector



- **Uses MDS and NWS to build an array of values**
  - **2 matrices (bw,lat) 2 arrays (cpu, memory available)**
  - **Matrix information is clique based**
- **On return from RS, Crafted Code filters information to use only machines that have the necessary software and are really eligible to be**

Arrays of Values Generated by Resource Selector

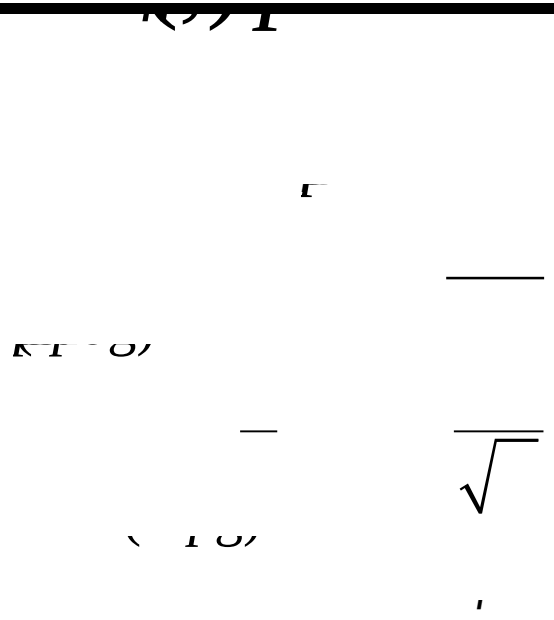
- **Clique based**
  - **2 @ UT, UCSD, UIUC**
  - **Full at the cluster level and the connections (clique leaders)**
  - **Bandwidth and Latency information looks like this.**
  - **Linear arrays for CPU and Memory**



After the Resource Selector …

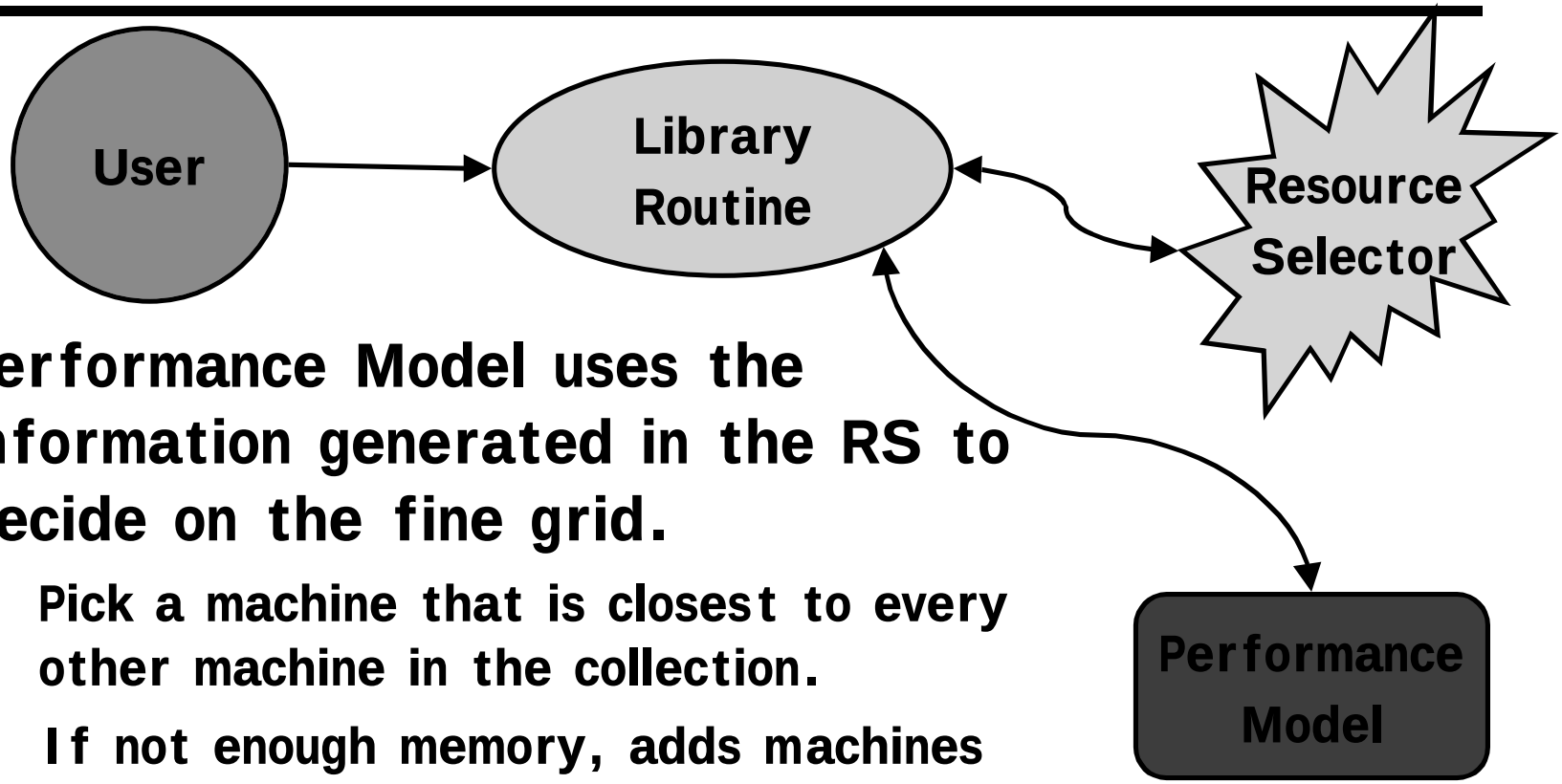
- **Matrix of values are filled out to generate a complete, dense, matrix of values.**
- **At this point have a workable coarse grid.**
  - **Workable in the sense that we know what is available, the connections, and the power of the machines.**

ScaLAPACK Performance Model



- **Total number of floating-point operations per processor**
- **Total number of data items communicated per processor**
- **Total number of messages**
- **Time per floating point operation**
- **Time per data item communicated**
- **Time per message**

Performance Model

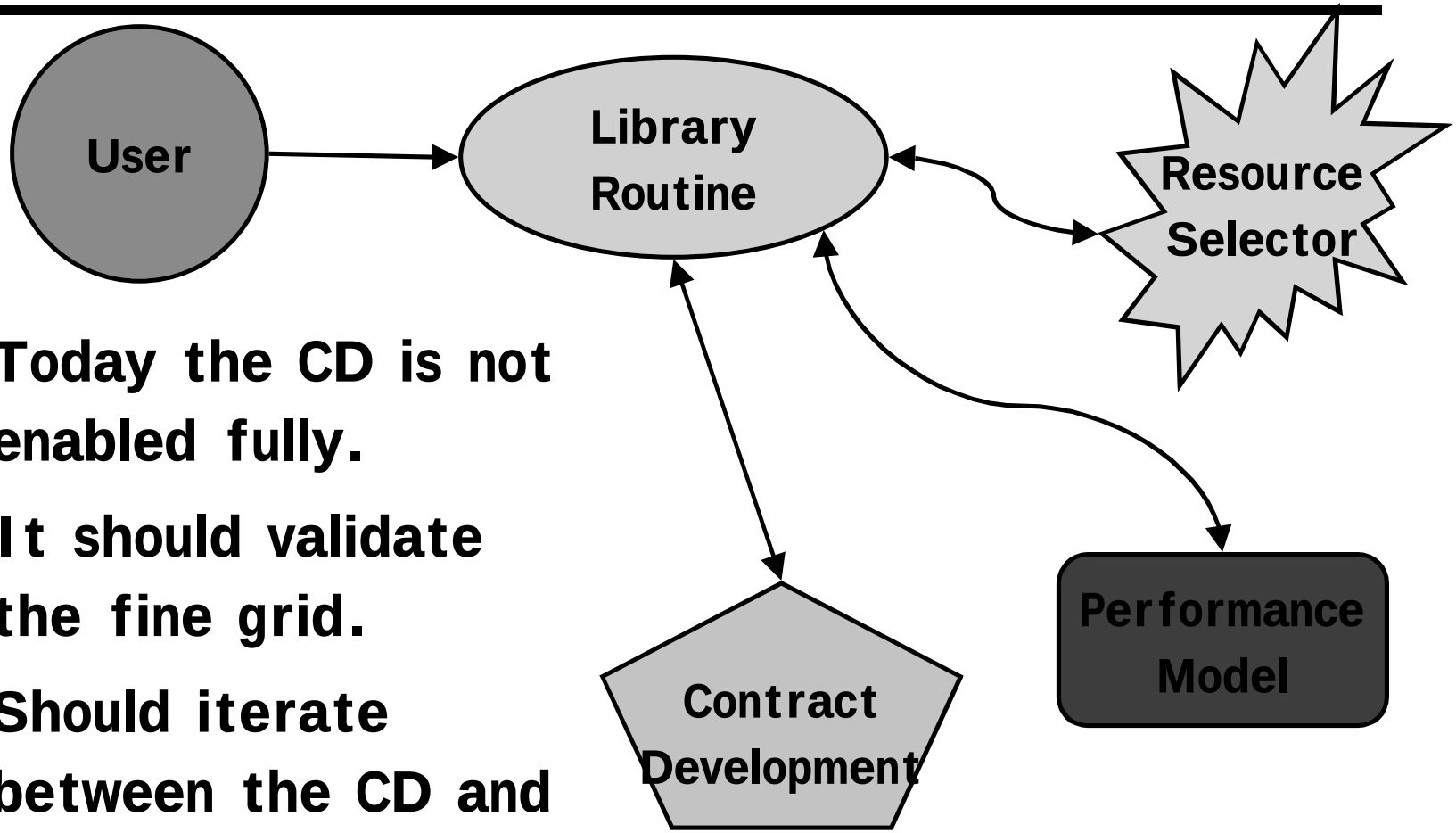


- **Performance Model uses the information generated in the RS to decide on the fine grid.**
  - **Pick a machine that is closest to every other machine in the collection.**
  - **If not enough memory, adds machines until it can solve problem.**
  - **Cost model is run on this set.**
  - **Process adds a machine to group and reruns cost model.**
  - **If "better", iterate last step, if not**

Performance Model

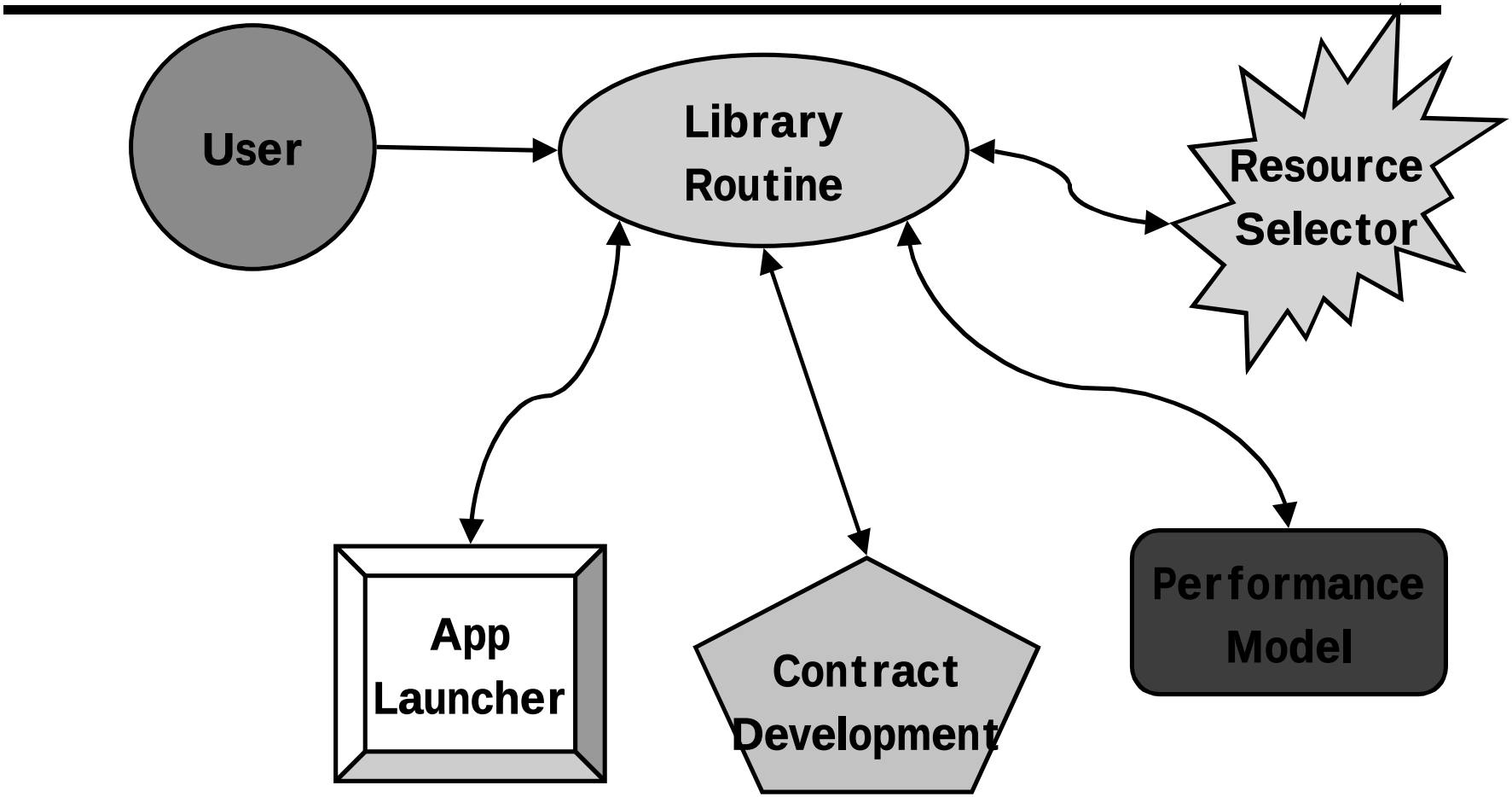
- **The PM does a simulation of the actual application using the information from the RS.**
  - **It literally runs the program without doing the computation or data movement.**
- **There is no backtracking implemented.**
  - **This is an area for enhancement and experimentation.**
  - **Only point to point information available for the cost model, ie don't have broadcast information between cliques.**
- **At this point we have a fine grid**

Contract Development



- **Today the CD is not enabled fully.**
- **It should validate the fine grid.**
- **Should iterate between the CD and PM phases to get a workable fine grid.**
- **In the futu**

Application Launcher

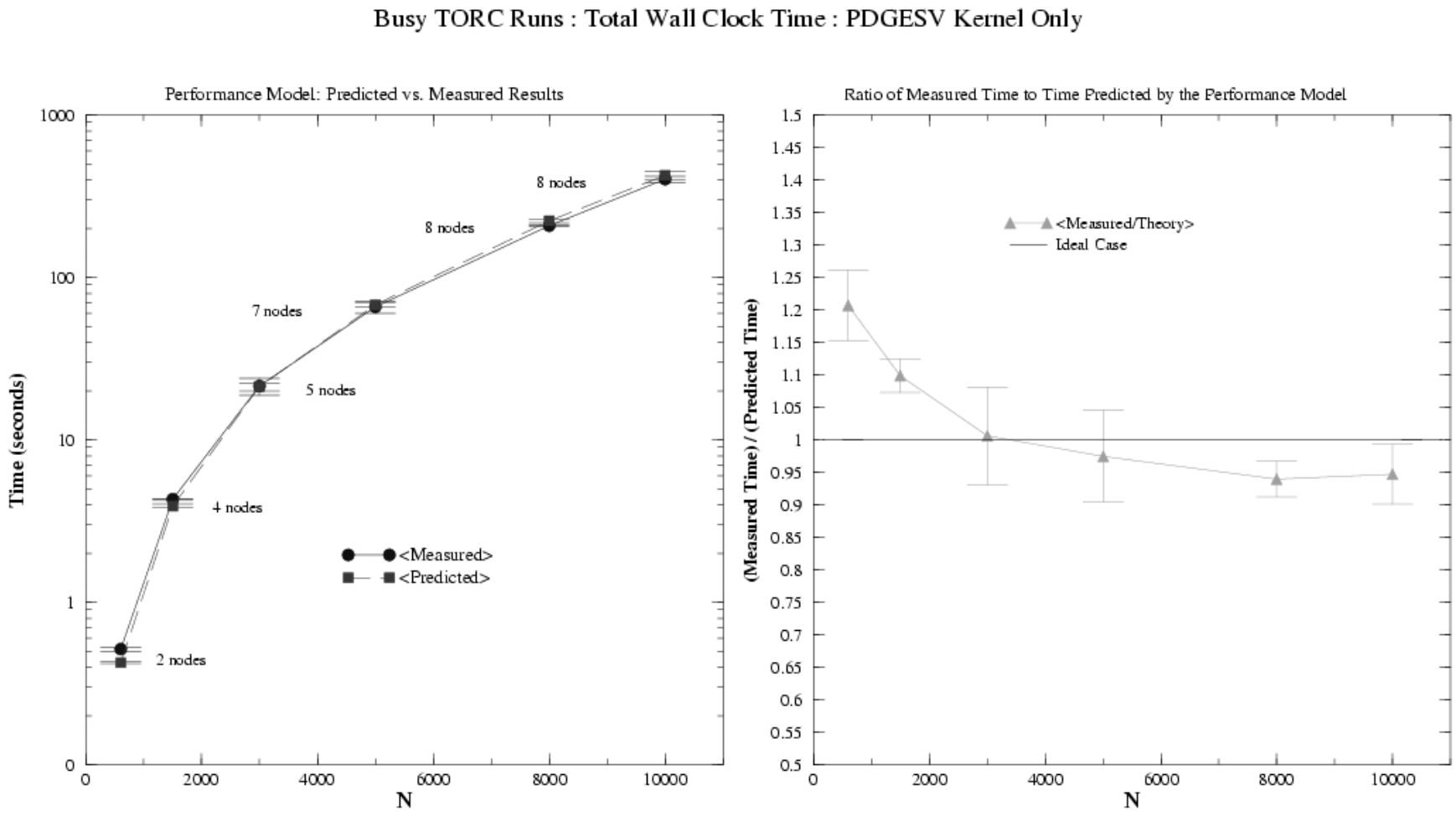


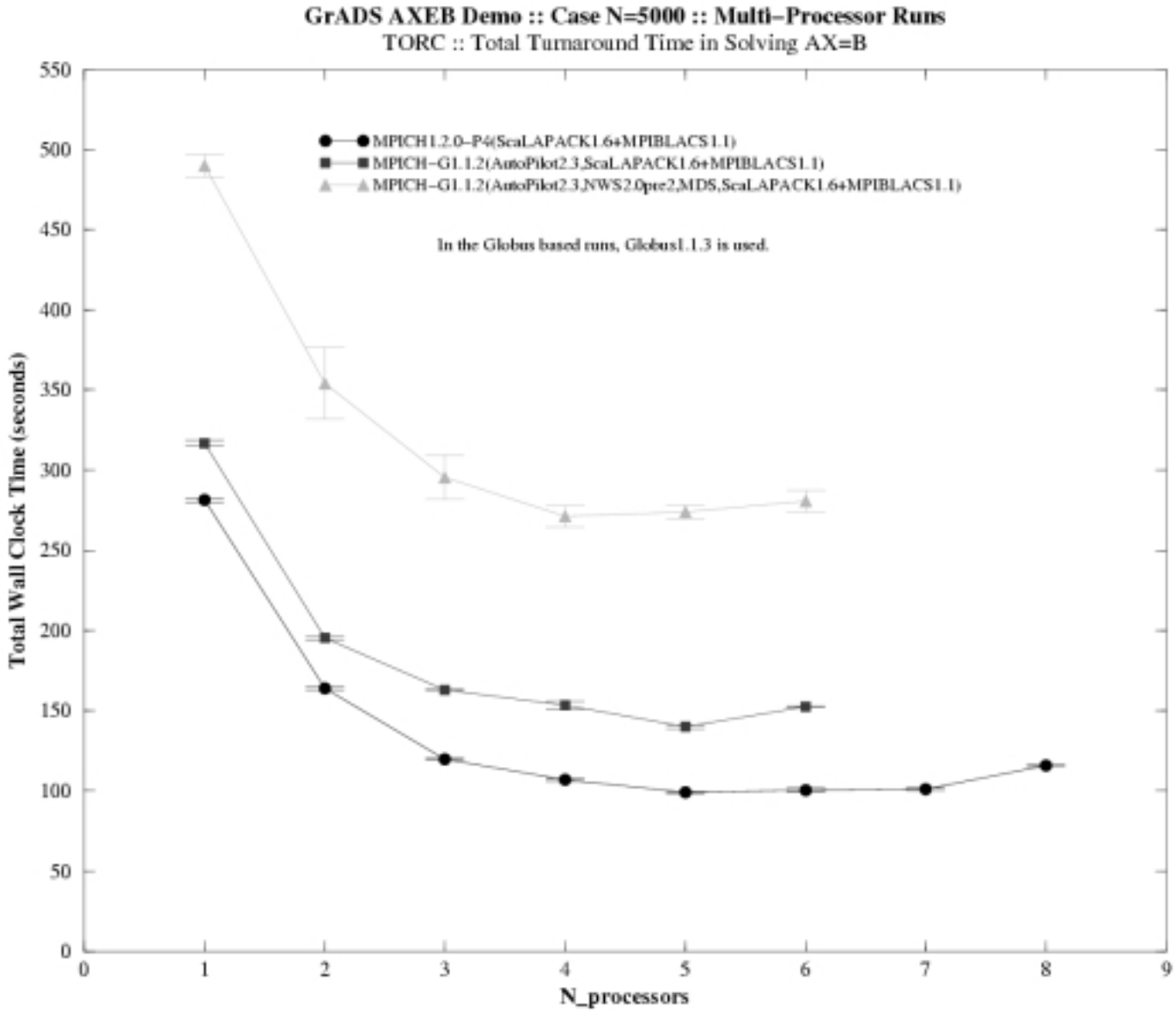
“mpirun –machinefile –globusrsl fine_grid grid_linear_solve”

Things to Keep in Mind About the Results

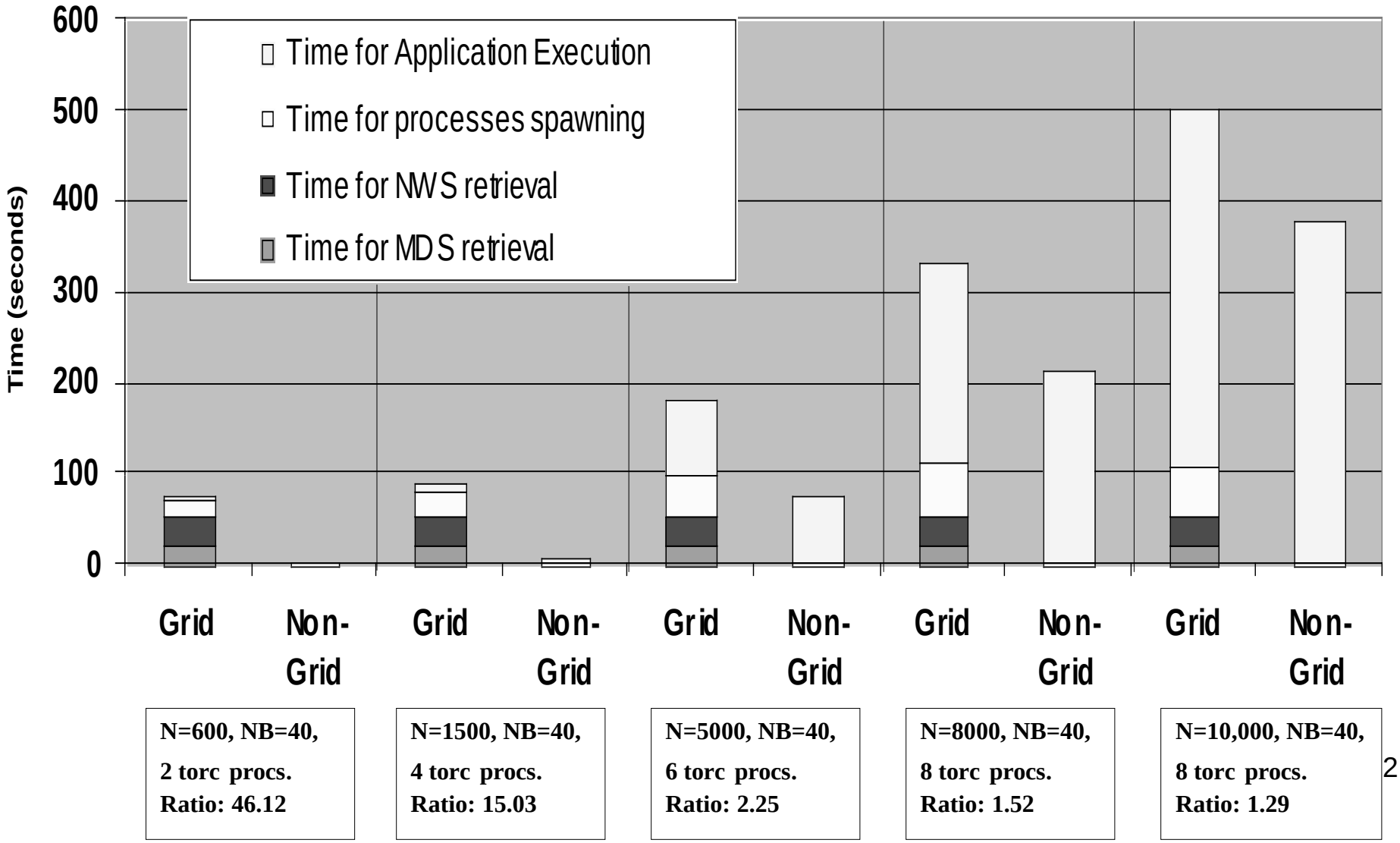
- **MPICH-G is not thread safe, so only one processor can be used of the dual machines.**
  - **This is really a problem with MPI in general.**
- **For large problems with on a well connected cluster ScaLAPACK gets ~3/4 of the matrix multiply exec rate and matrix multiply using ATLAS on a Pentium processor gets ~3/4 of peak. So we would expect roughly 50% of peak for ScaLAPACK in the best situation.**

Performance Model vs Runs

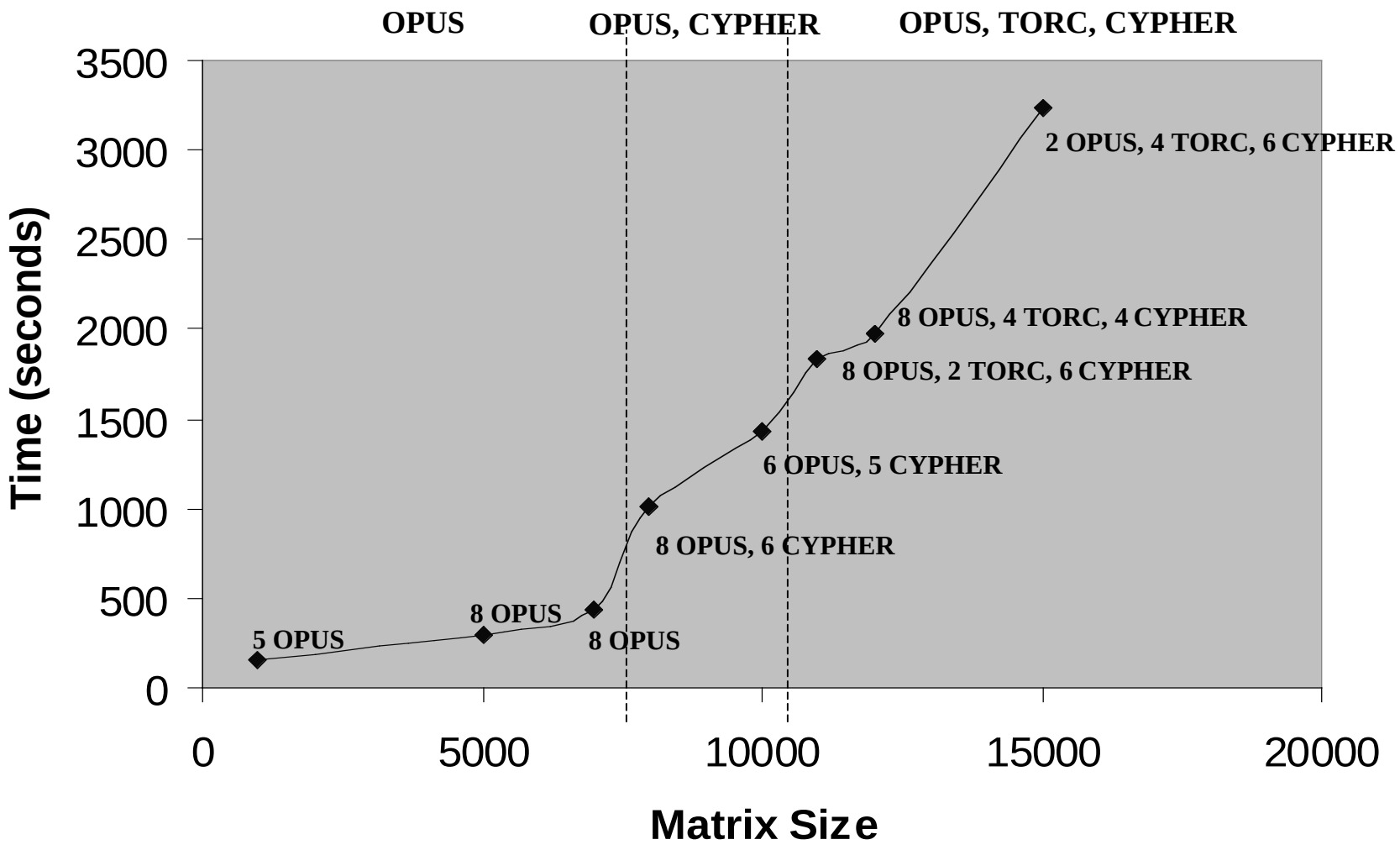



GrADS AXEB Demo :: Case N=5000 :: Multi-Processor Runs
TORC :: Total Turnaround Time in Solving AX=B
MPICH1.2.0-P4(ScaLAPACK1.6+MPIBLACS1.1)
MPICH-G1.1.2(AutoPilot2.3,ScaLAPACK1.6+MPIBLACS1.1)
MPICH-G1.1.2(AutoPilot2.3,NWS2.0pre2,MDS,ScaLAPACK1.6+MPIBLACS1.1)
In the Globus based runs, Globus1.1.3 is used.
Total Wall Clock Time (seconds)
N_processors
0
50
100
150
200
250
300
350
400
450
500
550
0
1
2
3
4
5
6
7
8
9

Grid ScaLAPACK vs Non-Grid ScaLAPACK, Dedicated Torc machines



ScaLAPACK across 3 Clusters



Largest Problem Solved

- **Matrix of size 30,000**
  - **7.2 GB for the data**
  - **32 processors to choose from UIUC and UT**
    - **Not all machines have 512 MBs, some little as 128 MBs**
  - **PM chose 17 machines in 2 clusters from UT**
  - **Computation took 84 minutes**
    - **3.6 Gflop/s total**
    - **210 Mflop/s per processor**
    - **ScaLAPACK on a cluster of 17 processors would get about 50% of peak**
    - **Processors are 500 MHz or 500 Mflop/s peak**

Futures (1)

- **Activate the sensors in the code to verify that the application is doing what the performance model predicted**
- **Enable Contract enforcement**
  - **If the "contract" is violated want to migrate application dynamically.**
- **Develop a better strategy in choosing the best set of machines.**
- **Implement fault tolerance and migration**
- **Use the compiler efforts to instrument code for contract monitoring and performance model development.**
- **Results are non-deterministic, need some way to**

Futures (2)

- **Would like to be in a position to make decisions about which software to run depending on the configuration and problem.**
  - **Dynamically choose the algorithm to fit the situation.**
- **Develop into a general numerical library framework**
- **Work on iterative solvers**
- **Latency tolerant algorithms in general**
  - **Overlap communication/computation**