GridSolve: A Seamless Bridge Between the Standard Programming Interfaces and Remote Resources

Jack Dongarra
University of Tennessee
and
Oak Ridge National Laboratory

Overview

♦ Grid/NetSolve
  ➢ Grid enabled software
  ➢ Allows easy access to remote resources

♦ No magic
  ➢ Someone has to write a program
  ➢ The program may run on a parallel computer

♦ NetSolve is a tool for distributed computing
What is NetSolve?

- **Client-server RPC-like system**
  - Designed for ease-of-use
- **Interactions mediated by an agent**
  - e.g. scheduling, tracking, fault tolerance
- **Dynamic service bindings**
  - Client does not need to have stubs for the services that it wishes to use
- **Multiple clients**
  - C, Fortran, Matlab, Java, Mathematica, Octave
- **Extended to support GridRPC API**
  - Part of GGF working group defining a standard API
NetSolve is an example of a Grid based hardware/software/data server.

Based on a Remote Procedure Call model but with …
- resource discovery, dynamic problem solving capabilities, load balancing, fault tolerance
- asynchronicity, security, ...

Easy-of-use paramount
- Its about providing transparent access to resources.
- Legacy codes easily wrapped into services

GridSolve Architecture

\[x, y, z, \text{info}] = \text{gridsolve('dgesv', A, B)}

Can be from Matlab, C, Fortran, Python, Java, Mathematica, Excel, ...
NetSolve Client

♦ Function Based Interface.
♦ Client program embeds call from NetSolve’s API to access additional resources.
♦ Interface available to C, Fortran, Matlab, and Mathematica.
♦ Opaque networking interactions.
♦ NetSolve can be invoked using a variety of methods: blocking, non-blocking, task farms, ...

NetSolve Client

♦ Intuitive and easy to use.
♦ Matlab Matrix multiply e.g.:
  ➢ $A = \text{matmul}(B, C);$

  $A = \text{netsolve}('matmul', B, C);$

  • Possible parallelisms hidden.
NetSolve Client

i. Client makes request to agent.

ii. Agent returns list of servers.

iii. Client tries first one to solve problem.

NetSolve Agent

- Name server for the NetSolve system.
- Information Service
  - client users and administrators can query the hardware and software services available.
- Resource scheduler
  - maintains both static and dynamic information regarding the NetSolve server components to use for the allocation of resources.
NetSolve Agent

♦ Resource Scheduling (cont’d):
  ➢ CPU Performance.
  ➢ Network bandwidth, latency.
  ➢ Server workload.
  ➢ Problem size/algorithm complexity.
  ➢ Calculates a “Time to Compute.” for each appropriate server.
  ➢ Notifies client of most appropriate server.

Basic Usage Scenarios

♦ Grid based numerical library routines
  ➢ User doesn’t have to have software library on their machine, LAPACK, SuperLU, ScalAPACK, PETSc, ARPACK, ...

♦ Task farming applications
  ➢ “Pleasantly parallel” execution eg Parameter studies
  ➢ Scavenge cycles

♦ Remote application execution
  ➢ Complete applications with user specifying input parameters and receiving output

♦ “Blue Collar” Grid Based Computing
  ➢ Does not require deep knowledge of network programming
  ➢ Level of expressiveness right for many users
  ➢ User can set things up, no “su” required
  ➢ In use today, up to 130 servers on the experimental grid

♦ Can plug into Globus, Condor, NINF, ...
Task Farming - Multiple Requests To Single Problem

- **A Solution:**
  - Many calls to netslnb(); /* non-blocking */

- Farming Solution:
  - Single call to netsolve_farm();

- Request iterates over an "array of input parameters."

- Adaptive scheduling algorithm.

- Useful for parameter sweeping, and independently parallel applications.

Server Proxies – Hide Parallelism

NetSolve System

LFC (LAPACK for Clusters), Condor, ScaLAPACK, etc.

User maybe unaware of parallel processing
**GridSolve Usage with VGrADS**

- Simple-to-use access to complicated software libraries, with no knowledge of grid based computing.
- Selection of best machines in your grid to service user request
- Portability
  - Non-portable calls can be run from a client using RPC like mechanisms as long there is a server provisioned with the code
- Legacy codes easily wrapped into services

**Plug into VGrADS Framework**

- Using the vgES for resource selection and launching of application:
  - Integrated performance information
  - Integrated monitoring
  - Fault prediction
  - Integrating the software and resource information repositories

**Virtual Grid Execution System (vgES)**

- **A Virtual Grid (VG) takes**
  - Shared heterogeneous resources
  - Scalable information service
- **and provides**
  - An hierarchy of application-defined aggregations (e.g. ClusterOf) with constraints (e.g. processor type) and rankings

**Virtual Grid Execution System (vgES) implements VG**

- VG Definition Language (vgDL)
- VG Find And Bind (vgFAB)
- VG Monitor (vgMON)
- VG Application Launch (VgLAUNCH+DVCW)
- VG Resource Info (vgAgent)
VGrADS/GridSolve Architecture

Agent

Client

Service Catalog

query

software location

vgDL

Virtual Grid

query

Software Repository

Data Persistence

- Chain together a sequence of NetSolve requests.
- Analyze parameters to determine data dependencies. Essentially a DAG is created where nodes represent computational modules and arcs represent data flow.
- Transmit superset of all input/output parameters and make persistent near server(s) for duration of sequence execution.
- Schedule individual request modules for execution.
Data Persistence (cont’d)

\[ \text{netsl} \begin{array}{l}
\text{begin_sequence( );}
\text{netsl(“command1”, A, B, C);}
\text{netsl(“command2”, A, C, D);}
\text{netsl(“command3”, D, E, F);}
\text{netsl_end_sequence(C, D);}
\end{array} \]

Industry Partners:
Microsoft, Sun, Dell, Cisco,
Foundry, Dolphin, Myracom

Current SInRG Infrastructure

- Federated Ownership: CS, Chem Eng., Medical School,
  Computational Ecology, El. Eng.
- Real applications, middleware development,
  logistical networking
NetSolve- Things Not Touched On

♦ Integration with other NMI tools
  ➢ Globus, Condor, Network Weather Service
♦ Security
  ➢ Using Kerberos V5 for authentication.
♦ Monitor NetSolve Network
  ➢ Track and monitor usage
♦ Fault Tolerance
♦ Local / Global Configurations
♦ Dynamic Nature of Servers
♦ Automated Adaptive Algorithm Selection
  ➢ Dynamic determine the best algorithm based on system status and nature of user problem
♦ NetSolve evolving into GridRPC
  ➢ Being worked on under GGF with joint with NINF

Software at:
http://icl.cs.utk.edu/netsolve/

NetSolve Team

♦ Sudesh Agrawal
♦ Don Fike
♦ Eric Meek
♦ Keith Seymour
♦ Zhiao Shi
♦ Asim YarKhan