Grid Computing: Application to Science

Aiichiro Nakano

Collaboratory for Advanced Computing & Simulations
Dept. of Computer Science, Dept. of Physics & Astronomy,
Dept. of Chemical Engineering & Materials Science
Department of Biological Sciences
University of Southern California

Email: anakano@usc.edu
Grid Computing

- **World Wide Web:** Universal interface to digital library on the Internet
- **Information Grid:** Pervasive (from any place in the world at any time) access to everything (computing, mass storage, experimental equipments, distributed sensors, etc., on high-speed networks)
Scientific Grid Applications

1. Distributed supercomputing (metacomputing): Uses geographically distributed multiple supercomputers to tackle problems that cannot be solved on a single platform.

2. Data-intensive science: Synthesizes new knowledge from massive data maintained in geographically distributed repositories, digital libraries & databases (Google science).


- Virtual community science—democratization of science “Do I really need all that infrastructure to do science?”
Application-Level Grid Tools

Grid programming models

- Message passing: MPICH-G2
- Remote procedure call: Ninf-G

Grid application types

- Metacomputing
- Parameter-sweep (high throughput) applications
- Workflow applications
- Portals: Thin-client, graphical user interfaces to the Grid
1. Grid programming
   > Metacomputing—multiscale MD/quantum-mechanical (QM) simulations:
     Grid-enabled MPI (MPI-G2)
   > Task farm: Grid remote procedure call (Ninf-G)
   > Sustainable & adaptive Grid supercomputing

2. Grid software
   > Globus toolkit
   > Open Grid Services Architecture (OGSA)
Multiscale FE/MD/QM Simulation

- Embed high-accuracy computations only when & where needed
- Train coarse simulations by fine simulations

Multiscale simulation to seamlessly couple:

- Finite-element (FE) calculation based on continuum elasticity
- Atomistic molecular-dynamics (MD) simulation
- Quantum-mechanical (QM) calculation based on the density functional theory (DFT)
Grid-Enabled MD/QM Algorithm

**Additive hybridization** (Morokuma et al., '96)

- **Extrapolation in meta-model space (accuracy vs. size)**

\[ E \approx E_{\text{system}}^{\text{MD}} + E_{\text{cluster}}^{\text{QM}} - E_{\text{cluster}}^{\text{MD}} \]

- **Modular**
  - Reuse of existing MD & QM codes
  - Minimal inter-model dependence/communication
Grid Enabling: Multiple QM Clustering

\[ E = E_{MD}^{system} + \sum_{\text{cluster}} [E_{QM}^{cluster}(\{r_{QM}\}, \{r_{HS}\}) - E_{MD}^{cluster}(\{r_{QM}\}, \{r_{HS}\})] \]

Divide-\&-conquer
Grid Implementation

- Task decomposition (MPI Communicator) + spatial decomposition
- Computation/communication overlapping to hide latency
- MPICH-G2 (www3.niu.edu/mpi)/Globus (www.globus.org)
Global Collaborative Simulation

Hybrid MD/QM simulation on a Grid of distributed PC clusters in the US & Japan

Japan: Yamaguchi — 65 P4 2.0GHz
Hiroshima, Okayama, Niigata — 3×24 P4 1.8GHz
US: Louisiana — 17 Athlon XP 1900+

MD — 91,256 atoms
QM (DFT) — 76n atoms on n clusters

- Scaled speedup, $P = 1$ (for MD) + 8n (for QM)
- Weak-scaling efficiency = 0.94 on 25 processors over 3 PC clusters

H. Kikuchi, et al., IEEE/ACM SC02
Grid-Enabled MD Algorithm

Grid MD algorithm:
1. asynchronous receive of cells to be cached `MPI_Irecv()`
2. send atomic coordinates in the boundary cells
3. compute forces for atoms in the inner cells
4. wait for the completion of the asynchronous receive `MPI_Wait()`
5. compute forces for atoms in the boundary cells

Renormalized Messages:
Latency can be reduced by composing a large cross-site message instead of sending all processor-to-processor messages
Promise of ultra-fast downloads

Soon you could be downloading an entire movie off the net far faster than you do now.

US researchers are working on ways to improve the way that net protocols decide how quickly data travels around the net.

Early tests of the new system show that it can triple data transmission speeds.

By linking lots of the faster systems together the researchers have produced data transfer speeds many times higher than is possible today.

Fast TCP: Achieved 8.6 Gb/s between Sunnyvale, CA & CERN, Switzerland

Steven Low (Caltech)

http://netlab.caltech.edu/FAST
Outline

1. Grid programming
   > Metacomputing—multiscale MD/quantum-mechanical (QM) simulations:
     Grid-enabled MPI (MPI–G2)
   > Task farm: Grid remote procedure call (Ninf–G)
   > Sustainable & adaptive Grid supercomputing

2. Grid software
   > Globus toolkit
   > Open Grid Services Architecture (OGSA)
Task Farm Applications

<table>
<thead>
<tr>
<th>Number CPUs</th>
<th>Number Active CPUs</th>
<th>Number Users</th>
<th>Number Teams</th>
<th>Last Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>423995</td>
<td>90438</td>
<td>210257</td>
<td>25971</td>
<td>2003-05-05</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>20:04:04</td>
</tr>
</tbody>
</table>

OS type | Active | Total
---|-------|-------
Windows | 86473  | 369859
Mac OS X | 2653  | 24129
Linux | 1294  | 29931
Other | 0     | 13
Total | 90420  | 423932

Screen saver (cf. OpenGL idle event handler)

http://folding.stanford.edu
World Wide Distributed Computing

Folding@home

http://folding.stanford.edu
Enabling Science by Online Game

Predicting protein structures with a multiplayer online game

Seth Cooper¹, Firas Khatib², Adrien Treuille¹,³, Janos Barbero¹, Jeehyung Lee³, Michael Beenen¹, Andrew Leaver-Fay²†, David Baker²,⁴, Zoran Popović¹ & Foldit players
Quake-Catcher Network

- Network of accelerometer-equipped laptops-desktops for early earthquake warning & research
- Clustering accelerometer time series data to detect earthquakes

http://qcn.stanford.edu
Harvard Clean Energy Project

http://cleanenergy.moleculespace.org

world community grid

Who We Are
World Community Grid brings together people from across the globe to benefit humanity by creating the world’s largest non-profit computing grid. We do this by pooling surplus processing power from volunteers’ devices. We believe that innovation combined with visionary scientific research and large-scale volunteerism can help make the planet smarter. Our success depends on like-minded individuals - like you.

How You Can Help
Download and install secure, free software that captures the spare processing power of your computer, smartphone or tablet, and harnesses it for scientific research.

http://www.worldcommunitygrid.org
Schrödinger Partners with Cycle Computing to Accelerate Materials Simulation

NEW YORK, N.Y., March 28 — Schrödinger, LLC and Cycle Computing, LLC announced today a partnership that will allow customers to run Schrödinger’s Materials Science Suite on the Cloud and elastic resources worldwide using Cycle Computing’s CycleCloud™ orchestration software. Cloud Computing provides users timely access to scalable computational resources as needed, without prohibitive upfront capital investment in infrastructure. Cycle Computing and Schrödinger have worked together on enabling many customer production workloads in the cloud, including the world’s largest and fastest cloud computing run of more than 156,000 cores called the MegaRun in late 2013.

During the record-breaking MegaRun, Professor Mark Thompson at the University of Southern California (USC) completed the largest cloud-computing run in the world, using Schrödinger’s software running on the CycleCloud platform. Professor Thompson calculated the optoelectronic properties for 205,000 materials with potential application in organic photovoltaic devices. The run used a maximum of 156,000+ CPU-cores completing 264 CPU-years of simulation in less than 18 hours.
Parallel History Matching

- **Inverse problem**: Calibration of reservoir simulation models to the observed production data

- Real field—offshore Africa
  - 32 wells
  - 30 years production history
  - 30 years production forecast

- CVX History Match & Associated Forecast (HMAF) framework: History matching & the assessment of uncertainties associated with flow prediction

HMAF: Landa & Guyaguler, SPE 84465 ('03)

Parallel HMAF: Landa, Kalia, Nakano, Nomura & Vashishta, IPTC 10751 ('05)
Overnight History Matching on a Grid

Nomura, Kalia, Nakano, Vashishta & Landa,
Journal of Supercomputing 41, 109 ('07)
Grid Remote Procedure Call (RPC)

- Simple RPC API (application program interface)
- Existing libraries & applications into Grid applications
- IDL (interface definition language) embodying call information, with minimal client-side management

```c
double A[n][n], B[n][n], C[n][n];  /* Data Declaration */
dmmul(n, A, B, C);                  /* Call local function */
grpc_function_handle_default(&hdl, "dmmul");
grpc_call(hdl, n, A, B, C);         /* Call server side routine */
```

- Ninf–G Grid RPC system
  http://ninf.apgrid.org
Outline

1. Grid programming
   > Metacomputing—multiscale MD/quantum-mechanical (QM) simulations:
     Grid-enabled MPI (MPI–G2)
   > Task farm: Grid remote procedure call (Ninf–G)
   > Sustainable & adaptive Grid supercomputing

2. Grid software
   > Globus toolkit
   > Open Grid Services Architecture (OGSA)
Combined GridRPC+MPI MD/QM

- **Original implementation (MPICH-G2)**

- **Hybrid implementation (GridRPC + MPI)**

- **Flexibility:** Dynamically add/subtract, allocate, & migrate resources

- **Fault tolerance:** Automatically detect & recover from explicit (OS down or disconnected networks) & implicit (job stuck in a queue) faults

- **Scalability:** Manage 1000’s of computing resources efficiently
Global Grid QM/MD

- Hybrid GridRPC (ninf.apgrid.org) + MPI (www.mcs.anl.gov/mpi) Grid computing
- 153,600 cpu-hrs metacomputing at 6 sites in the US (USC, PSC—Pittsburgh, NCSA—Illinois) & Japan (AIST, U Tokyo, TITech)

H. Takemiya et al., Proc. IEEE/ACM SC06; Y. Song et al., Int’l J. Comput. Sci. (’09); CCGrid09
Flow Chart of Grid MD/QM

Program status
- Waiting invocation
- Invocation
- Initialization
- Waiting QM simulation
- Executing QM simulation
- Updating data
- Terminating QM simulation

MD simulation
- Set task()
- Exec. init()
- Wait. init()
- Exec. sim()
- Wait. sim()

Scheduler
- Select
- Retain initial
- Retain previous result
- Read configuration file
- Check reservation schedule
- Change the No. of CPUs

QM simulation
- Invoke server
- Grpc. object. handle. init()
- Grpc. invoke. async("init")
- Grpc. wait()
- Grpc. invoke. async("QM")
- Grpc. wait()
- Grpc. object. handle. destruct()
- Terminate

Flow:
- Calculate classical forces in the system
- Calculate classical forces in the clusters
- Update atomic positions and velocities
- Calculate quantum mechanical forces in the clusters

Interface:
- Scheduler API
- GridRPC API
SIMOX (Separation by Implantation by Oxygen)

Red: quantum mechanically treated atoms $\sim O(N^3)$
Flexibility: Adaptive MD/QM

- Flexibility: Automated increase of the number of QM atoms on demand to maintain accuracy & associated dynamic re-allocation of CPUs
Fault Tolerance

- Automated migration in response to unexpected faults
1. Grid programming
   > Metacomputing—multiscale MD/quantum-mechanical (QM) simulations:
     Grid-enabled MPI (MPI–G2)
   > Task farm: Grid remote procedure call (Ninf–G)
   > Sustainable & adaptive Grid supercomputing

2. Grid software
   > Globus toolkit
   > Open Grid Services Architecture (OGSA)
Goal: Coordinated resource sharing & problem solving in dynamic, multi-institutional virtual organizations.

1. Coordinate distributed resources.
2. Use standard, open, general-purpose protocols & interfaces.
3. Deliver nontrivial qualities of services (QoS).
Grid Architecture

Layered architecture

- **Hourglass model:** A small set of core protocols (e.g., TCP/IP) + various: (1) high-level behaviors & (2) underlying technologies.
Layered Grid Architecture

• Fabric: Introspection & management of local resources.
  > Computational resources: Start programs & monitor/control the execution of the resulting processes.
  > Storage: Put & get files (e.g., disk space allocation).
  > Network: Control network transfers (e.g., prioritization).

• Connectivity: Define communication & authentication protocols.

• Resource: Define protocols for negotiation, initiation, monitoring, control, accounting & payment of sharing operations on individual resources.

• Collective: Capture interactions across collection of resources, e.g., directory services, co-allocation & data replication.
Globus Toolkit

- Globus Toolkit version 2 (GT2): Open source, de facto standard of Grid computing middleware to construct interoperable Grid applications (’97)
  > Define & implement protocols, application program interfaces (APIs) & services
  > Provide solutions to authentication, resource discovery & resource access
- Globus Toolkit version 3 (GT3): OGSA-compliant standard (’02)
- Download Globus Toolkit version 6 (GT 6.0) at http://www.globus.org/toolkit
GT2: Globus Toolkit 2

- **Fabric**
  - General purpose architecture for reservation & allocation (GARA)

- **Connectivity**
  - Grid security infrastructure (GSI)

- **Resource**
  - Grid resource allocation & management (GRAM) protocol
1. Grid programming
   > Metacomputing—multiscale MD/quantum-mechanical (QM) simulations:
     Grid-enabled MPI (MPI–G2)
   > Task farm: Grid remote procedure call (Ninf–G)
   > Sustainable & adaptive Grid supercomputing

2. Grid software
   > Globus toolkit
   > Open Grid Services Architecture (OGSA)
Open Grid Services Architecture (OGSA)

- **OGSA = Definition of a service-oriented infrastructure**
- **Service:** A network-enabled entity with a well-defined interface that provides some capability

1. Align Grid computing with industrial initiatives in service-oriented architecture & Web services
2. Provide a framework within which to define interoperable & portable services
3. Define a core set of standard interfaces & behaviors
4. Implemented in the OGSA-based Globus Toolkit 3 (currently GT 6.0):
   http://www.globus.org
Open Grid Services Architecture (OGSA)

1. Web services description language (WSDL): An interface definition language describing services (or software components) independent of platforms.

2. Open grid services infrastructure (OGSI): A set of WSDL interfaces & associated conventions, extensions & refinements to Web services standards to support basic Grid.
OGSI Functionalities

- Grid service description & instances: Definition & execution
- Service state, metadata & introspection
- Naming & name resolution: Universal resource identifier (URI)
- Service life cycle: Instantiate & destruct
- Fault type: Standard base type for all fault messages
- Service groups: Represent & manage groups of services
OGSA Services

• Core services
  > Name resolution & discovery
  > Security
  > Policy
  > Messaging, queuing & logging
  > Events
  > Metering & accounting: Resource usage & charges

• Data & information services
  > Data management & access
  > Replication
  > Metadata & provenance

• Resource & service management
  > Provisioning & resource management
  > Service orchestration
Virtualization-aware Application Framework

Atomistic materials simulation methods

Molecular Dynamics (MD)

Quantum Mechanics (QM)

Atom

Electron density

Virtualized applications

MD/QM

Grid

Computing resources

- Scalability
- Portable performance
- Adaptation

Data-locality principles
Divide-conquer-recombine

1. **Computational complexity:** Computation time, $T$, as a function of the problem size, $N$

\[
O(N) \quad < \quad O(N^m) \quad < \quad O(C^N)
\]

Easy \quad \leftarrow \quad \text{Hard}

2. **Scalability:** Parallel efficiency

\[
\eta = \frac{T_1}{(T_p p)} \sim 1
\]

for a large number, $p$, of processors

3. **Fault resilience**