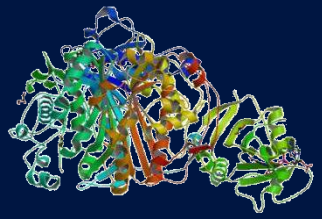
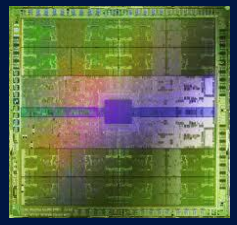
HTPMD

High Throughput Parallel Molecular Dynamics





renci

Steve Cox
RENCI Engagement

Overview

- High Throughput Parallel Computing
- Molecular Dynamics
- First User
- Solution
- Bigger Challenges
- Workflow and Hybrid Computing

High Throughput Parallel Computing (HTPC)

- Objectives
  - Exploit parallel processing OSG resources
  - Simplify submission to hide details (RSL/targeting)
  - Integrate with existing submission models
  - Explore MPI delivery and execution
- Status
  - 8-way jobs are the practical upper bound
  - About a half dozen sites are HTPC enabled
  - Implementing discoverable GIP configuration

Molecular Dynamics (MD)

Molecular dynamics is computer simulation of physical movements by atoms and molecules.

- Wikipedia

"…everything that living things do can be understood in terms of the jigglings and wigglings of atoms."

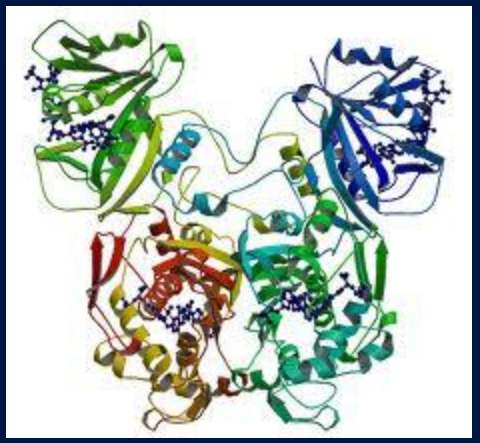
- Richard Feynman

Amber / PMEMD

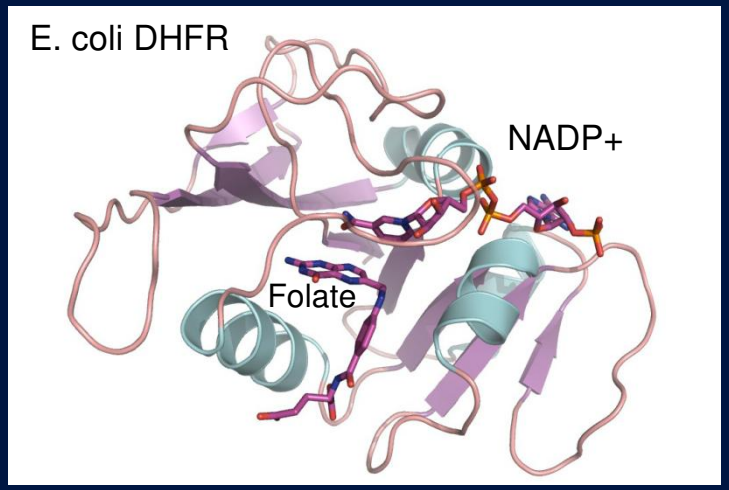
- Widely used for molecular simulation
- Atomic motion modeled at nanosecond granularity
- PMEMD: Particle Mesh Ewald Molecular Dynamics
- Heavily reliant on message passing interface (MPI)
- Works with MPICH / MPICH2 among others
- Can be statically linked for portability
- One researcher on Amber9, one on Amber10
- Amber11 PMEMD is GPGPU accelerated

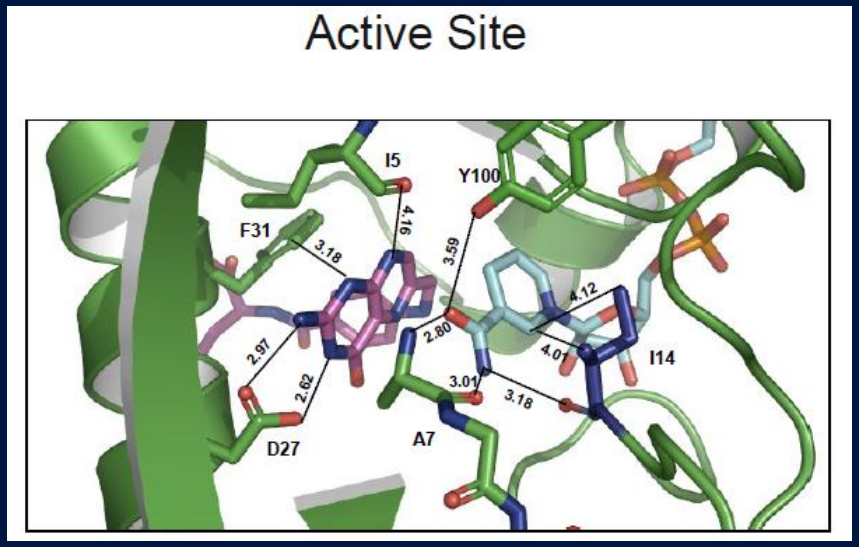
Case Study 1: DHFR Protein Dynamics & FDH

- Dr. **Laura** Perissinotti of U. Iowa
- Referral from SBGrid
- Studying
  - (1) Dihydrofolate Reductase
    - Found on chromosome 5
    - Required for manufacture of purines
    - Catalyzes DNA components
  - (2) Formate Dehydrogenase – instrumental in
    - E. coli anaerobic respiration
    - Decomposition of compounds like methanol



Case Study 1: DHFR Protein Dynamics & FDH



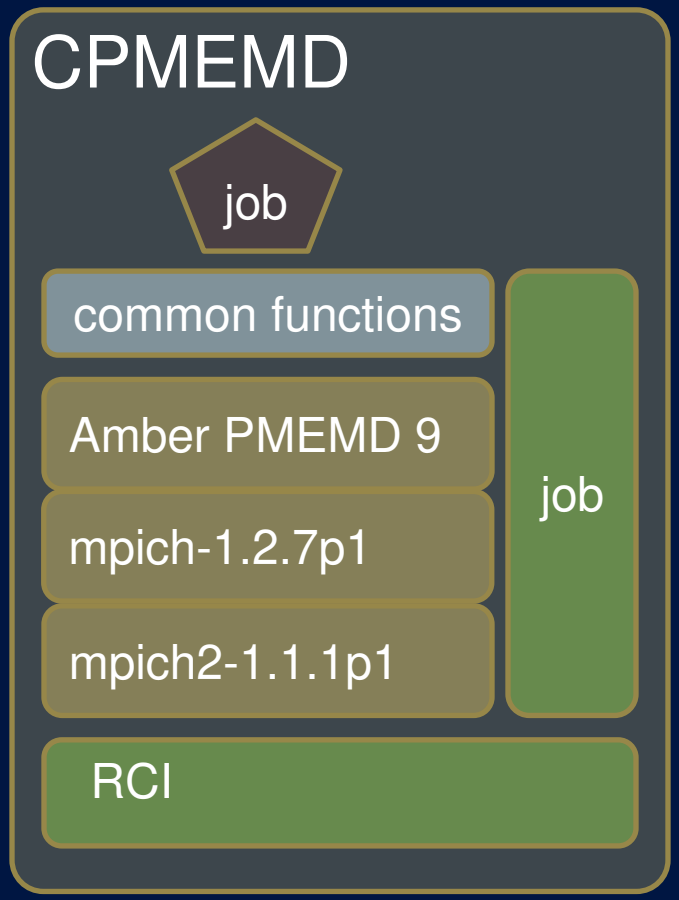


Low atom count relative to upcoming projects

Case Study 1: Simplify the Researcher-Grid Interface

CPMEMD packages

- Amber PMEMD
- MPI Libraries (MPICH, MPICH2)
- OSG Adapter Scripts
- RCI – Job Control

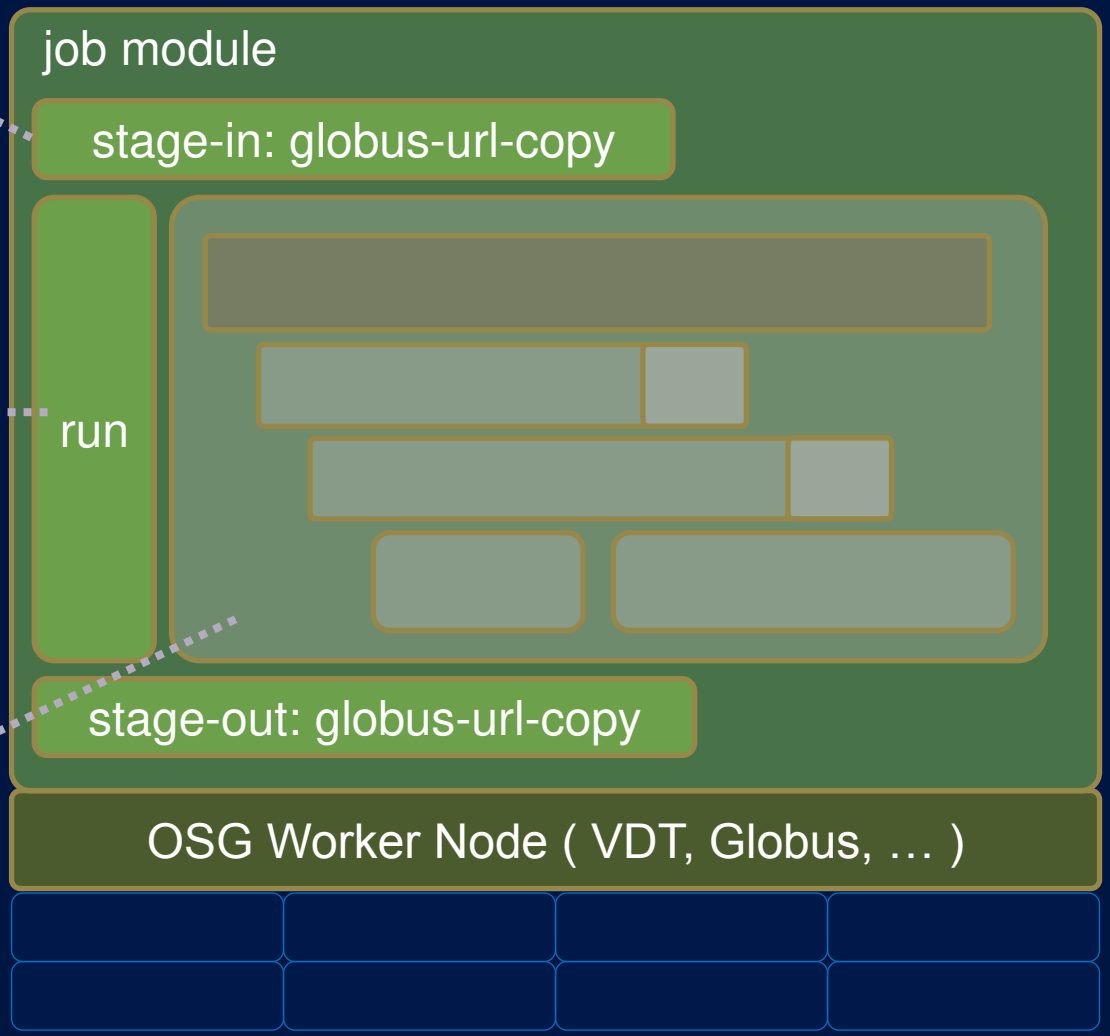


Case Study 1: Simplify the Researcher-Grid Interface

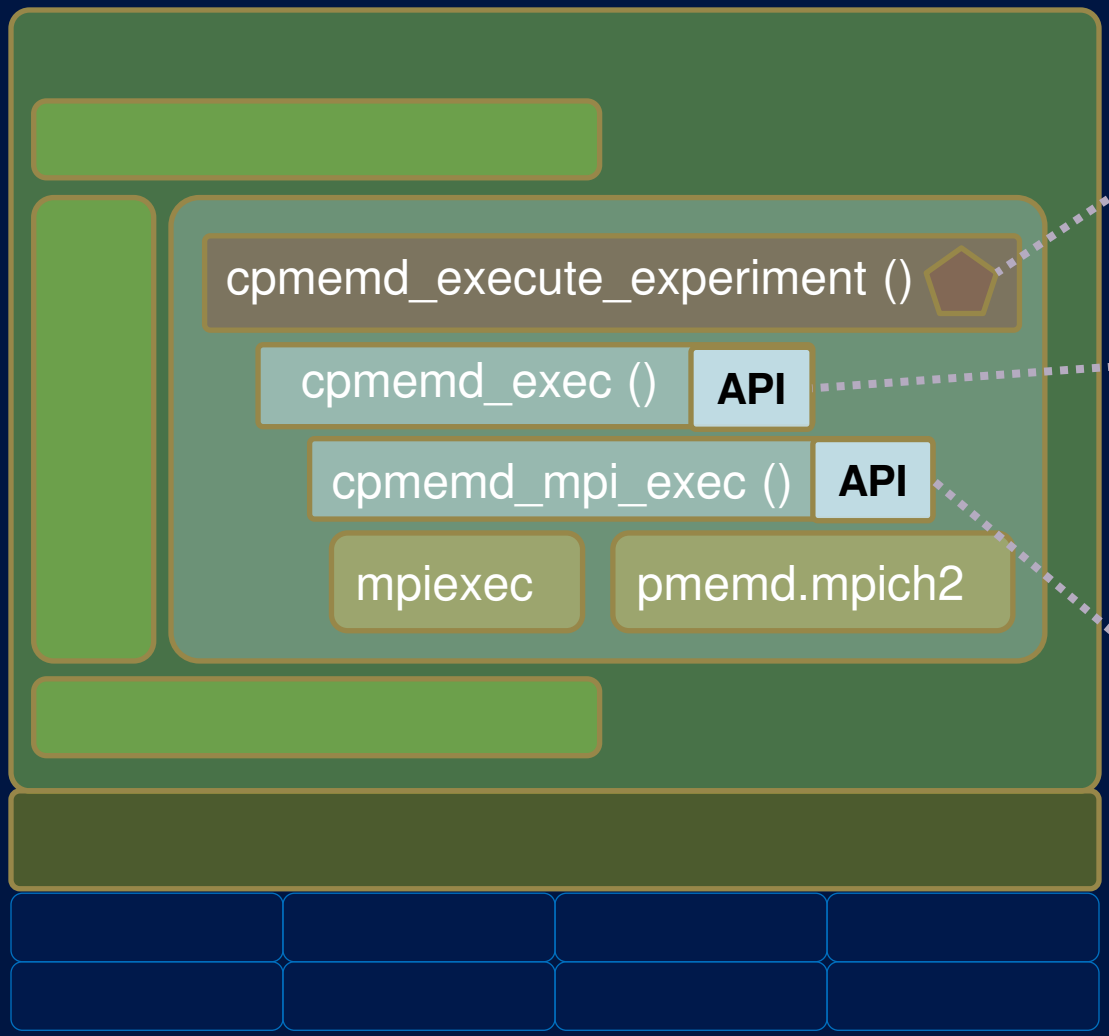
All files are staged in and out for the user

The framework provides static executables, runs the specified experiment and tracks and reports exit status

The framework provides an API to run PMEMD via MPI



Case Study 1: Simplify the Researcher-Grid Interface



Researchers focus on the experiment - implement a standard entry point.

Execute PMEMD with a template driven input file; inputs and outputs from and to standard locations

Execute PMEMD with complete control over all parameters while still allowing the framework to manage MPI launch

Case Study 1: Outcomes (a)

- Laura is using it in production
  - OSG is “approximately 4 to 8 times faster”
  - Able to execute and extend it independently
- Gratia statistics so far
  - WallDuration: 310,721
  - CpuDuration: 1,841,945
  - CpuSystemDuration: 19,645
- Anticipating
  - 100ns of DHFR simulation
  - FDH simulation
    - PAAD probe: 50ns
    - Mutants: 200ns
  - Approximately
    - 35 jobs
    - WallDuration: 1,500,000

Case Study 1: Outcomes (b)

- Shortcomings
  - Poor performance relative to (GPU) alternatives
  - Too much workflow management code
  - Too little platform independent meta-data
  - Experiments are monolithic programs
  - No abstract models of – well – anything, really
  - No semantic value without reading all the code
  - Wont scale to UNC CSB's larger problems

Case Study 2: UNC Center for Structural Biology

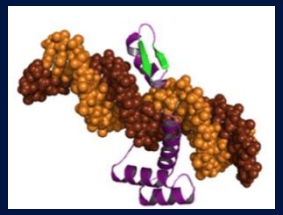
- Brenda Temple, PhD
  - Executive Director of the UNC CSB
  - Provides MD expertise to researchers
  - Uses Amber PMEMD extensively
  - Manages a variety of simultaneous MD projects
  - Projects are of widely varying complexity
  - Regularly runs 128-way jobs on a UNC cluster

Case Study 2: UNC Center for Structural Biology

CSB Center for Structural Biology

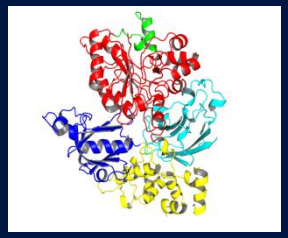
Brenda Temple, PhD.
Executive Director

Design of artificial transcription factors



Pilar Blanquefort's Lab

Regulation of PLC-b2 Activity by Conserved Motions of the X-Y Linker

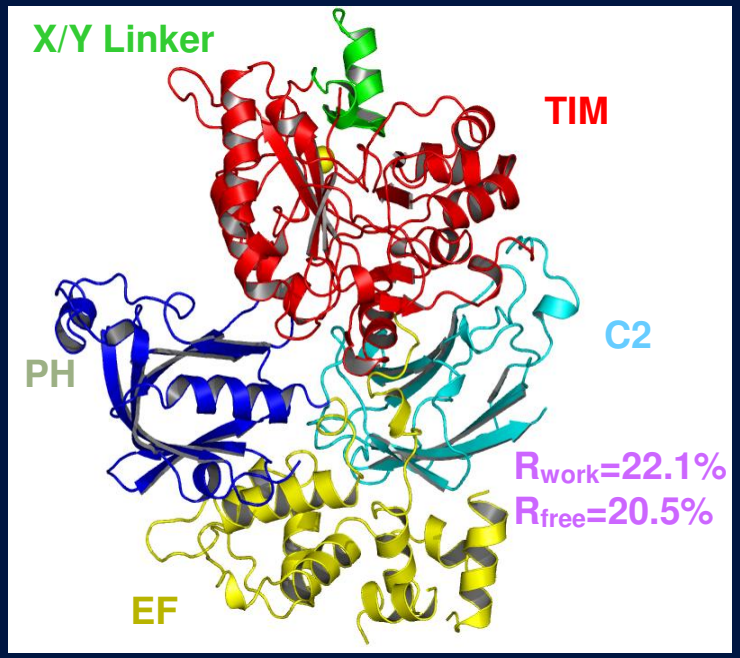


John Sondek's Lab

complexity

Case Study 2: CSB and the Sondek Lab

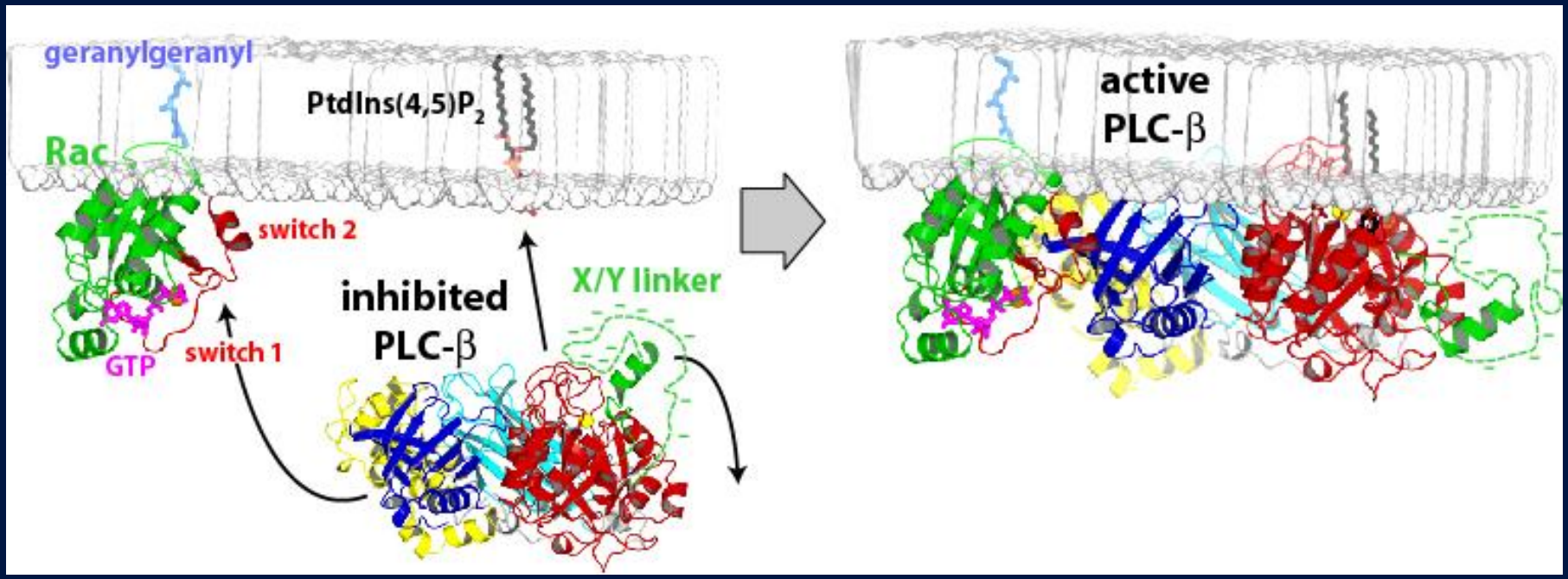
- Why Should We Use Molecular Dynamics to Study PLC-b2?
- **Working Hypothesis**: Negative charges in the linker are critical for auto-inhibition of PLC activity
- What is the Role of Electrostatics in X/Y Linker?
- How Does the Presence of a Membrane Influence the Motions of the Linker?
- Rate: 128 CPU/day x 1 ns/day = 65 ns / 65 days
- Our Goal is 200ns simulations



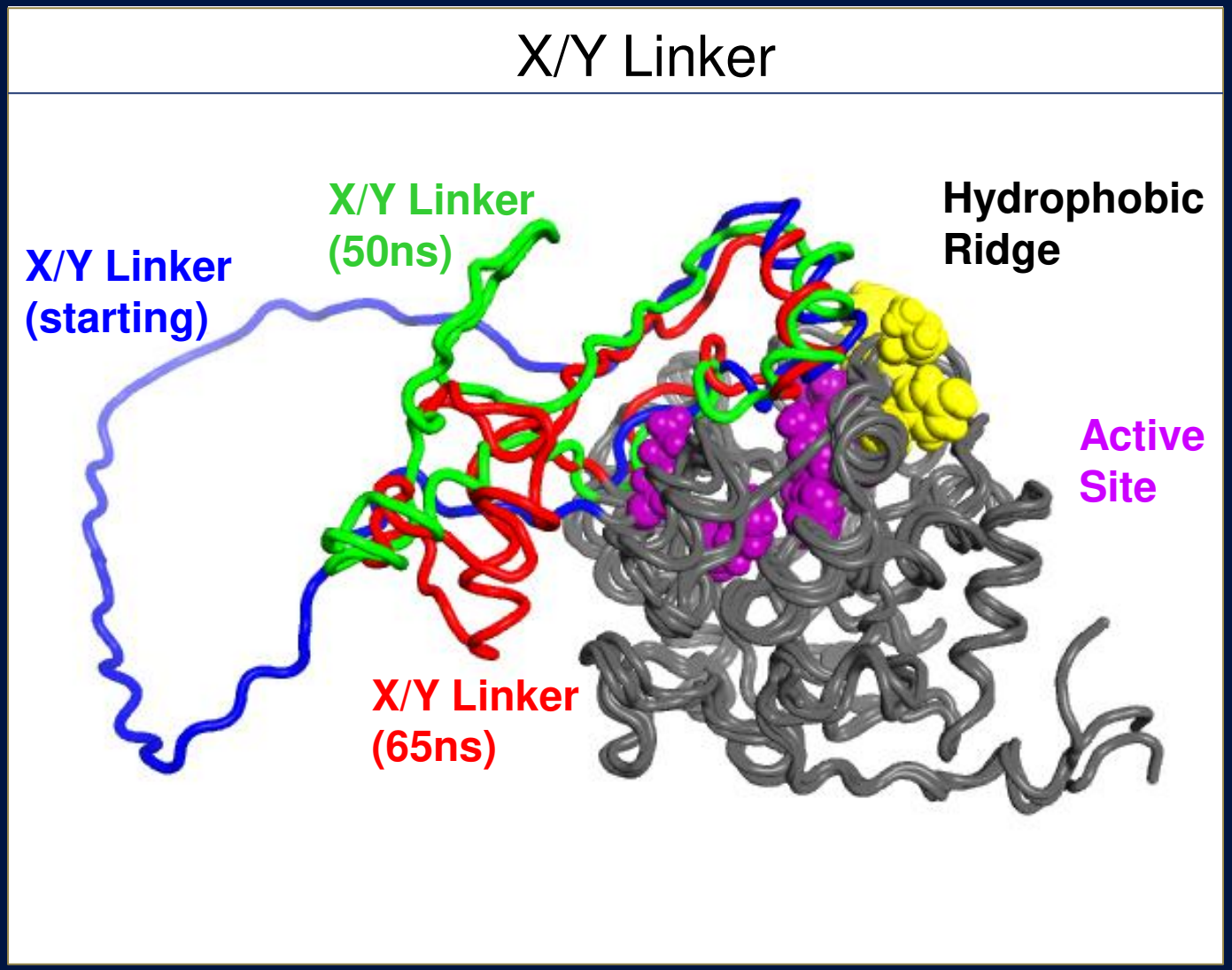
Phospholipase C-b2 Active Site is Occluded by X/Y Linker

Case Study 2: CSB and the Sondek Lab

Proposed Mechanism for Release of Auto-inhibition of PLC



Case Study 2: CSB and the Sondek Lab



Case Study 2: CSB and the Sondek Lab

- Mechanism of Collapse
  - Run longer simulations with wt, K475M, and G530P PLC-b2 mutants to evaluate collapse of linker
  - Run simulations with linker mutated to Gln and Ala to further investigate importance of negative charge in motions of X/Y linker
- Scope of Mechanism: Simulate X/Y linker motion for PLC-d
- Experimentally address MD insights
  - Mutate K475 to eliminate/reverse charge and evaluate *in vivo* effects
    - Met, Ala, Ser, Asp
  - Mutate Glu & Asp residues in X/Y linker to Gln, Asn, or Gly and Ser
- Historical note on in-silico molecular dynamics at the CSB:
  - 3-5 years ago: 10 ns of simulation was average
  - Now: 50 ns of simulation is about average

Case Study 2: Observations

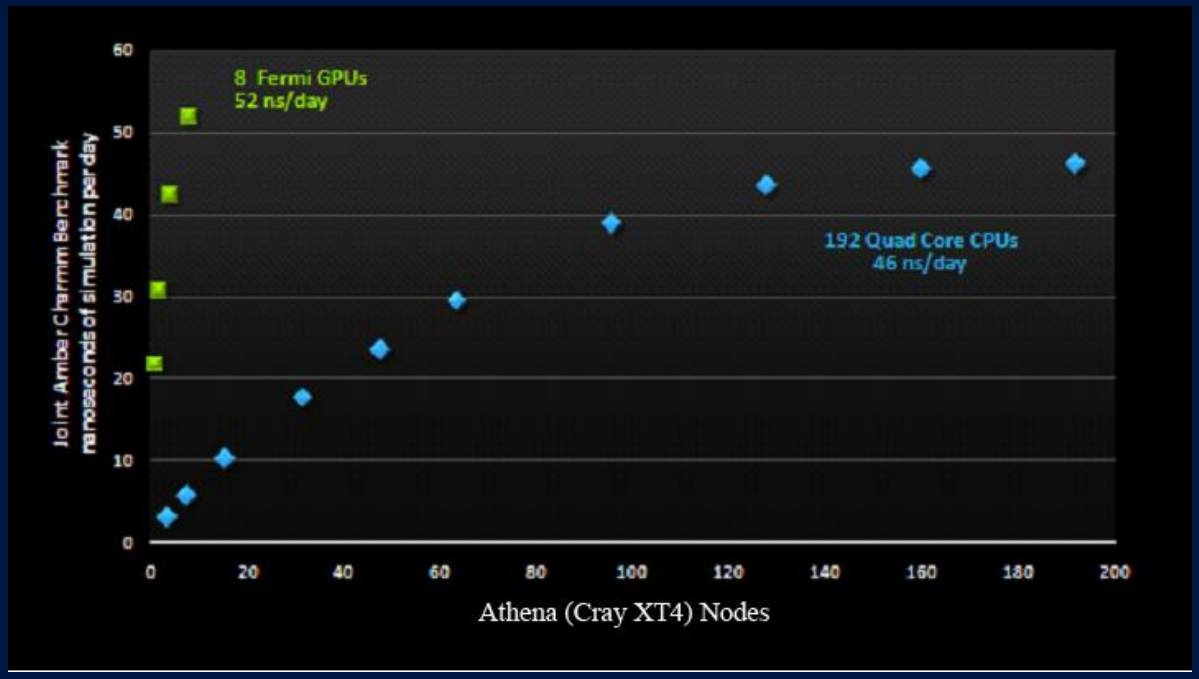
- Better performance is vital
- Current experiments
  - Have dozens of phases
  - Workflow semantics implemented as shell scripts
  - Structure is hidden from non-experts
  - Monolithic construction impedes reuse
- The future is
  - More complex workflow
  - Greater demand for compute power
- Scalable, semantically rich infrastructure needed

Second Generation: Performance and Workflow

- **GPGPU** improves performance dramatically
  - General Purpose Graphics Processing Units
  - Amber11 for GPU on RENCI-Blueridge
  - Available via Blueridge OSG CE interface
  - Extending GIP to model GPGPU-HTPC
  - Need to reflect the GPU difference in accounting
  - New FERMI GPUs a significant advance over Tesla

Are GPGPU's worth the effort?

Yes. The GPU architecture makes a critical difference in the performance of parallel molecular dynamics simulations



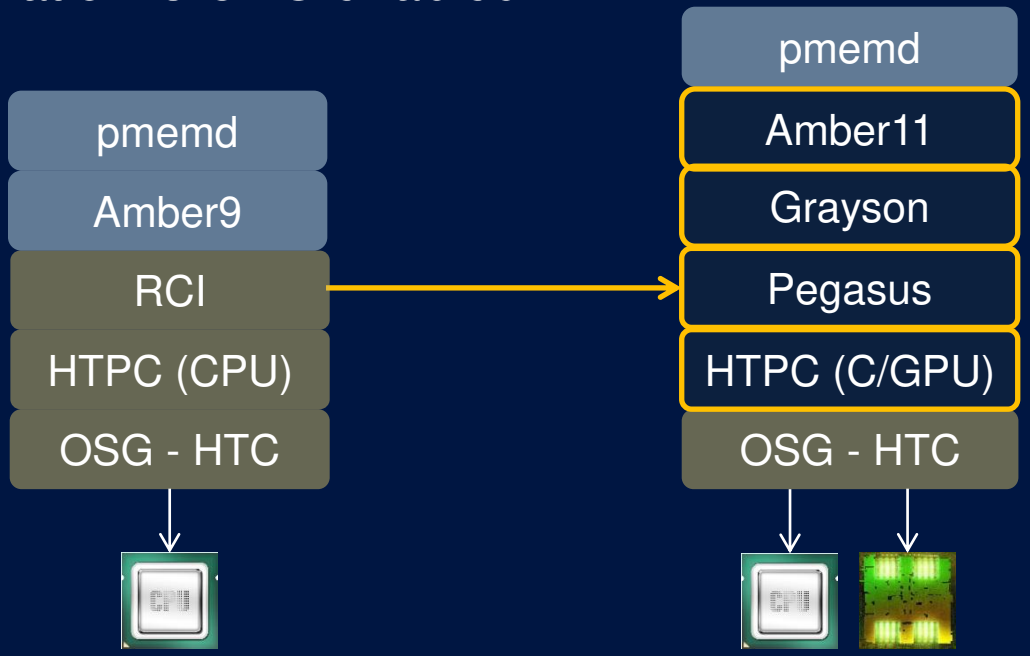
Amber11 PMEMD on FERMI

Second Generation: Performance and Workflow

- **Pegasus** for Workflow Management
  - An HTC differentiating advantage
  - Workflow framework simplifies development
  - Standards (XML) based workflow representation
  - Extensible via DAX APIs in Java, Python, Perl
  - Manages vital but tedious stage-in/out

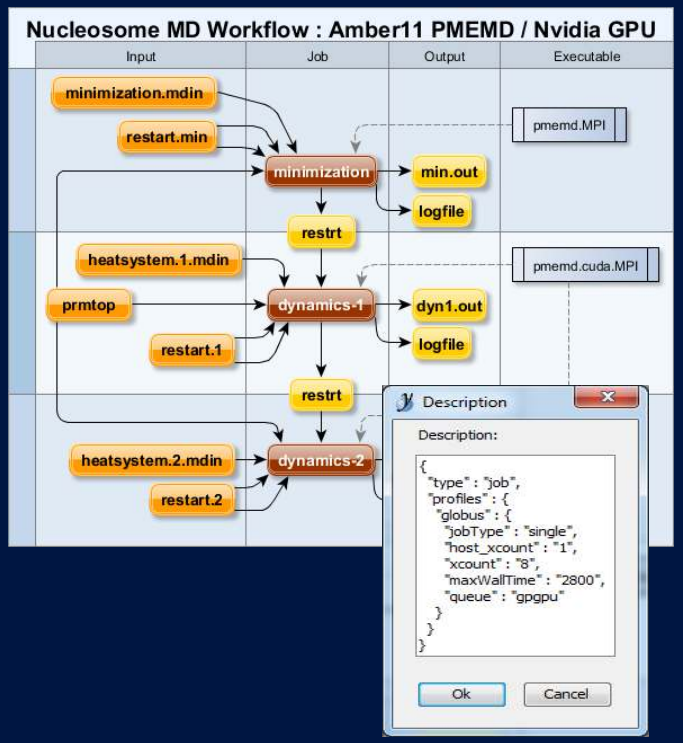
Second Generation: Changes to the Stack

- Amber11 provides GPGPU support for PMEMD
- Pegasus replaces various scripts (RCI)
- HTPC in a hybrid CPU/GPU architecture
  - PMEMD minimization calculation is CPU only
  - Dynamic calculation is GPU enabled



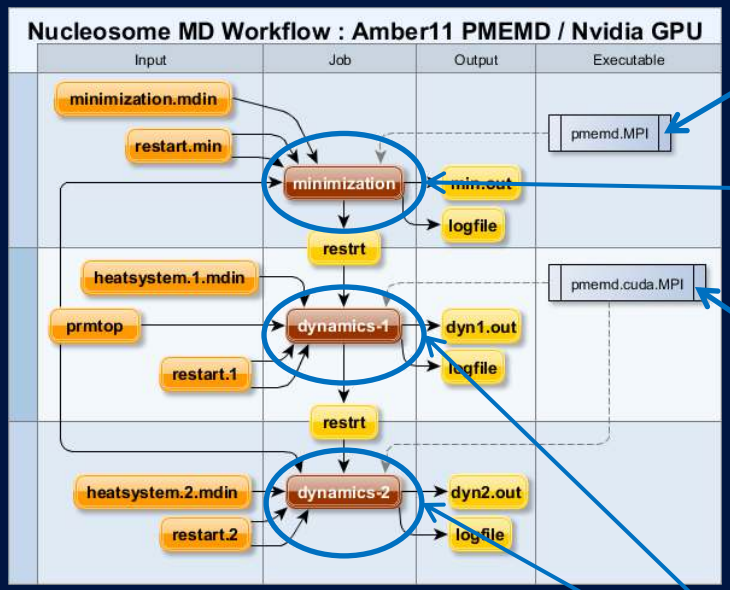
Introducing Grayson for Pegasus

Model Driven Architecture Applied to Workflow Management



- GraphML with JSON annotation
- Intuitive semantics with regard to
  - Input / Output
  - Order
  - Parallelism
  - Executable to job relationships
- Portable, open standard representation
- Execution environment independent
- Semantically rich meta-data with JSON
- Generates Pegasus DAX workflow format
- Produces information-rich visual artifacts

Grayson for Pegasus



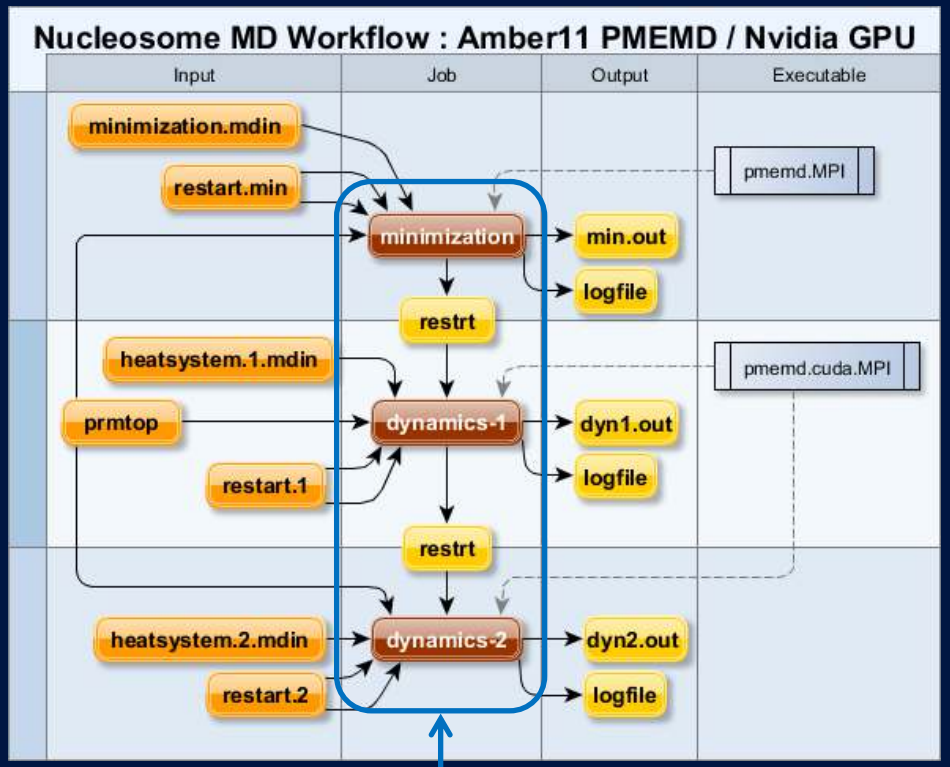

```
{
 "type" : "executable",
 "path":"/home/scox/gpu/bin/pmemd.MPI",
 "site" : "TestCluster"
}
```

```
{
 "type" : "job",
 "profiles" : {
  "globus" : {
   "jobType" : "single",
   "host_xcount" : "1",
   "xcount" : "8",
   "maxWallTime" : "2800"
  }
 }
}
```

```
{
 "type" : "executable",
 "path":"/home/scox/gpu/bin/pmemd.cuda.MPI",
 "site" : "TestCluster"
}
```

```
{
 "type" : "job",
 "profiles" : {
  "globus" : {
   "jobType" : "single",
   "host_xcount" : "1",
   "xcount" : "8",
   "maxWallTime" : "2800",
   "queue" : "gpgpu"
  }
 }
}
```

Grayson for Pegasus




```
{
  "type" : "job",
  "profiles" : {
    "globus" : {
      "jobType" : "single",
      "host_xcount" : "1",
      "xcount" : "8",
      "maxWallTime" : "2800",
      "queue" : "gpgpu"
    }
  }
}
```

Job input, outputs, executables and profile information are all translated to Pegasus DAX form.

```
        <job id="ID0000003" namespace="gpmemd" name="dynamics-2" version="4.0">
                <argument>-t <file name="prmtop"/> -i <file name="heatsystem.2.mdin"/>
                <profile namespace="globus" key="queue">gpgpu</profile>
                <profile namespace="globus" key="xcount">8</profile>
                <profile namespace="globus" key="maxWallTime">2800</profile>
                <profile namespace="globus" key="jobType">single</profile>
                <profile namespace="globus" key="host_xcount">1</profile>
                <uses name="prmtop" link="input"/>
                <uses name="heatsystem.2.mdin" link="input"/>
                <uses name="restrt" link="input"/>
                <uses name="restart.2" link="input"/>
                <uses name="restart.2" link="input"/>
                <uses name="logfile" link="output"/>
                <uses name="dyn2.out" link="output"/>
        </job>

        <!-- part 4: List of control-flow dependencies (may be empty) -->
        <child ref="ID0000002">
                <parent ref="ID0000001"/>
        </child>
        <child ref="ID0000003">
                <parent ref="ID0000002"/>
        </child>
</adag>
```

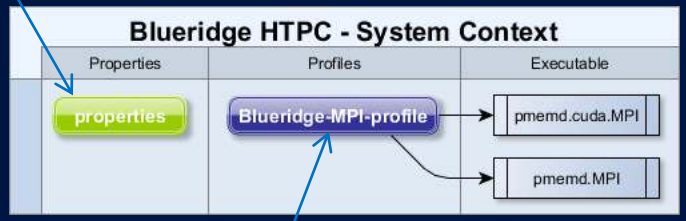
Input and output chains model DAX parent->child relationships.

Grayson for Pegasus

- Grayson 0.2
  - Increase workflow reusability
  - Simplify workflow creation
  - Remove execution environment details from the workflow model
  - Develop a context specific model reusable across workflows
  - Make workflows abstract process models
- Key features
  - Model by Reference: refer to a component in another model
  - Properties: compose the application flexibly
  - Inheritance: sort of - technically closer to aggregation
  - Profile Inheritance: jobs aggregate executable requirements
  - Separate Compilation: compose systems from separate models

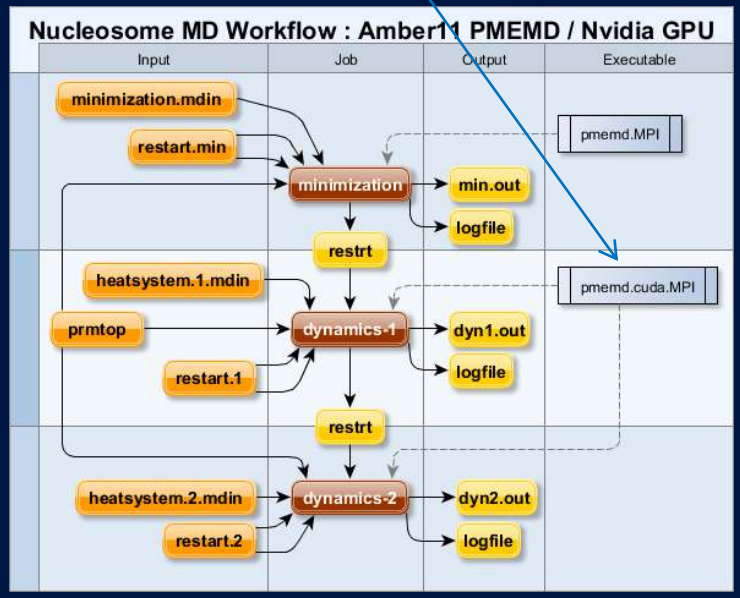
Grayson for Pegasus 0.2

```
{
  "type" : "properties",
  "map" : {
    "pmemdMPI" : "/home/scox/gpu/bin/pmemd.MPI",
    "pmemdCudaMPI" : "/home/scox/gpu/bin/pmemd.cuda.MPI",
    "clusterId" : "TestCluster"
  }
}
```




```
{
  "type" : "abstract",
  "profiles" : {
    "globus" : {
      "jobType" : "single",
      "host_xcount" : "1",
      "xcount" : "8",
      "maxWallTime" : "2800"
    }
  },
  "site" : "${clusterId}"
}
```

```
{
  "type" : "executable",
  "path" : "${pmemdCudaMPI}",
  "profiles" : {
    "globus" : {
      "queue" : "gpgpu"
    }
  }
}
```



Grayson for Pegasus

- **Running Grayson**
  - Compile one or more GraphML files depicting workflows
  - Emit a Pegasus DAX modeled by the workflow
  - Emit site catalog information
  - Execute pegasus-plan to submit the generated DAX

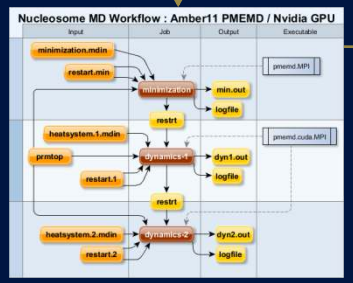
```
graysonc \
    --model nucleosome.graphml \
    --model blueridge-context.graphml \
    --namespace=gpmemd \
    --version=1.0 \
    --output=gridpmemd.dax \
    --site

pegasus-plan \
    -D pegasus.user.properties=pegasusrc \
    --sites TestCluster \
    --dir work \
    --output local \
    --dax gridpmemd.dax \
    --verbose \
    --submit
```

Grayson for Pegasus

Design

yEd



Compile / Plan / Submit

*.graphml

graysonc

pegasus-plan

```
gpmemd-0.dag succeeded
02/21 15:54:20  Done     Pre   Queued    Post   Ready   Un-Ready   Failed
02/21 15:54:20   ===     ===      ===     ===     ===        ===      ===
02/21 15:54:20    17       0        0       0       0          0        0

WORKFLOW STATUS : COMPLETED | 17/17 ( 100% ) | (all jobs finished successfully)
```

Monitor / Execute

condor

globus

pbs

cpu

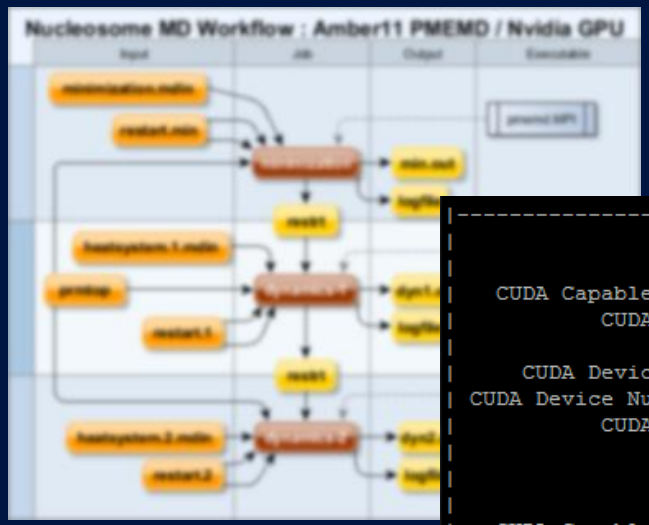
gpu

renci

Steven Cox: http://osglog.wordpress.com

Second Generation: Grayson / Pegasus / GPGPU

From concept…



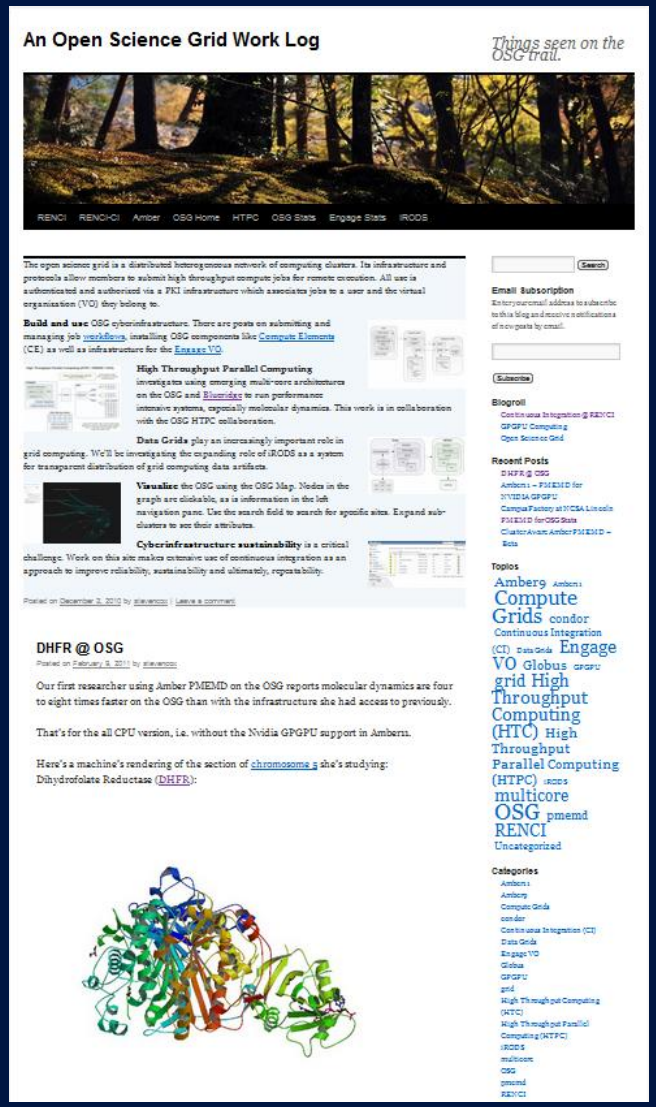

```
|------------------- GPU DEVICE INFO --------------------
|
|                          Task ID:      0
|   CUDA Capable Devices Detected:      4
|            CUDA Device ID in use:      0
|                 CUDA Device Name: Tesla T10 Processor
|     CUDA Device Global Mem Size:   4095 MB
| CUDA Device Num Multiprocessors:     30
|            CUDA Device Core Freq:   1.44 GHz
|
|
|                          Task ID:      1
|   CUDA Capable Devices Detected:      4
|            CUDA Device ID in use:      1
|                 CUDA Device Name: Tesla T10 Processor
|     CUDA Device Global Mem Size:   4095 MB
| CUDA Device Num Multiprocessors:     30
|            CUDA Device Core Freq:   1.44 GHz
|
|-----------------------------------------------------
```

to silicon

Conclusion

- HTPC MD on OSG ready for prime-time
- GPGPU via OSG/HTPC is demonstrated
  - Accounting work needed to reflect benefit
  - Design ongoing for GIP discoverability
- Grayson for Pegasus
  - Model Driven Architecture for Workflows
  - Semantically rich artifacts
  - Open standards and portability
  - Execution environment independent

References



- Steve's OSG Blog
- HTPC Wiki
- Pegasus WMS
- Amber PMEMD
- NVIDIA CUDA
- UNC CSB
- John Sondek's Lab