Using the Pyramid Library for Finite Element Mesh Adaptation

Charles D. Norton, John Z. Lou, and Thomas A. Cwik
National Aeronautics and Space Administration
Jet Propulsion Laboratory
California Institute of Technology
http://hpc.jpl.nasa.gov/APPS/AMR
HPCC/Earth and Space Sciences Project
PYRAMID: Parallel Unstructured Adaptive Mesh Refinement

Modern... Simple... Efficient... Scalable...

Technology Description
An advanced software library supporting parallel adaptive mesh refinement in large-scale, adaptive scientific & engineering simulations.

State-of-the-Art Design!
- Efficient object-oriented design in Fortran 90 and MPI
- Automatic mesh quality control & dynamic load balancing
- Scalable to hundreds of processors & millions of elements

Application Arena
- Computer Modeling & Simulation Applications with complex geometry
- Electromagnetic and semiconductor device modeling
- Structural/Mechanical/Fluid dynamics applications

John Z. Lou, Charles D. Norton, & Thomas A. Cwik
High Performance Computing Systems and Applications Group
http://hpc.jpl.nasa.gov/APPS/AMR

Initial geometry courtesy of SCOREC (Rensselaer)
Our Parallel AMR Process

Organization
- Partitioning, Adaptive Refinement, Load Balancing, Mesh Migration, and Element Quality Control

- Initial Mesh Partitioning
- Application Computation
  - Estimated Error > Tolerance? Y
  - Adaptive Refinement (logical phase)
  - Load Balancing? Y

- Addressed by AMR Quality Control
  - Mesh Improvement
  - Mesh Smoothing?
    - Adaptive Refinement (physical phase)
  - Mesh Repartition and migration
For more info...

Fortran 90 Programming. Ellis, Philips, & Lahey; Addison Wesley, 1994
http://hpc.jpl.nasa.gov/PEP/nortonc/oof90.html
A Minimal PYRAMID Program

- Initialization Section
  - Optional arguments override defaults

```fortran
PROGRAM pyramid_example
USE pyramid_module
implicit none
! Statements omitted...
type (mesh), dimension(2) :: meshes
call PAMR_INIT()
call PAMR_LOAD_MESH_PARALLEL( meshes(1), in_file )
call PAMR_REPARTITION( meshes(1) )
! Adaptive refinement loop...
call PAMR_ELEMENT_COUNT( meshes(2) )
call PAMR_VISUALIZE( meshes(2), "visfile.plt" )
call PAMR_FINALIZE( mpi_active = .true. )
END PROGRAM pyramid_example
```
Technology

- A Minimal PYRAMID Program
  - Adaptive Refinement

```plaintext
PROGRAM pyramid_example
  ! Adaptive refinement loop...
  do i = 1, refinement_level
    call PAMR_ERROR_EST( meshes(1), &
                         meshes(2) )
    call PAMR_LOGICAL_AMR( meshes(1) )
    call PAMR_REPARTITION( meshes(1) )
    call PAMR_PHYSICAL_AMR( meshes(1), meshes(2) )
  end do
END PROGRAM pyramid_example
```

- Users must specify their error estimation method
- Mesh hierarchies can be defined
Object-Based Access to Data Structure

- Explicit reference to element coordinates is complicated

  type (mesh) :: this
  real, dimension(3) :: xyz_pos
  xyz_pos = this%nodes(this%elements(2)%node_index(1))%coord

- PYRAMID simplifies such references

  type (mesh) :: this
  real, dimension(3) :: xyz_pos
  real, dimension(3,4) :: all_pos
  real, dimension(3,3,4) :: n_normal
  xyz_pos = PAMR_ELEMENT_COORD(this, element_index=2, &
                             node_index=1)
  all_pos = PAMR_ELEMENT_COORD(this, element_index=2)

  ! Access signed local normal basis for all faces
  n_normal = PAMR_FACE_NORMALBASIS(this, element_index=3)
Numerous User-Driven Commands Are Included

- Initialization, Mesh I/O, Termination, Adaptive Refinement, Repartitioning, Data Migration, Visualization, Data Structure Access, Mathematical, and Auxiliary
- Almost every command contains optional arguments for use customization

- Most commands are generic based on the mesh component applied
Dynamic Load Balancing with ParMetis

- ParMetis gives partitioning, PYRAMID performs migration
- Migration handles irregular communication patterns with a scalable and efficient non-blocking algorithm

- We are investigating Zoltan (Sandia National Labs) as an additional option for partitioning
Technology

Automatic Mesh Quality Control
- Modify coarse element refinement if successive refinements cause poor aspect ratios
- Controls quality at the expense of additional elements
Technology

Automatic Mesh Quality Control
- Benefit of quality control applied to triangular elements
Technology

Automatic Mesh Quality Control
- Benefit of quality control applied to tetrahedral elements

Poor Mesh Elements Without Quality Control

Good Mesh Elements With Quality Control

Note: Tecplot shows some edges in the backplane that do not exist in the mesh...
Technology

Large Scale Parallel Mesh Generation
- Specify uniform error for generation from coarse meshes

Parallel Uniform Refinement
Performance issues in Algorithms and Networking for Beowulf-Class Clusters
Performance

Pentium III Beowulf Cluster vs. SGI O2K Parallel AMR
- O2K scales well although the processor is slower than the 800 Mhz Beowulf PIII
- Beowulf competes well, but performance is limited by 100 BaseT network

![Graph showing performance comparison between Pentium III Beowulf and SGI O2K](image_url)

![Graph showing network performance on SGI O2K](image_url)
Performance

---

Irregular Data Communication
- Migration requires irregular, but predictable, data movement that varies in size and destination

Circular-Shift "MPI_SENDRECV(...)"
- All processors inspect all of the data
- "Guarantees" handling of cyclic deadlock dependencies
- Irregular data sizes affect pipelined flow performance
- MPI implicit buffering, due to poor pipeline structure, leads to poor performance

Direct Data Transfers
- Processors send/receive specific messages
- Send continuously while checking for receives
- "Arbitrary" message ordering can flood the network switch, leading to poor performance
Performance

Irregular Data Communication
- Reduction schemes can improve performance, if implemented with care...

Reduce to 0 with Broadcast Scheme
- Not scalable and very inefficient for large data sets

Reduce (or Exchange) to Leader with Subset Broadcast Scheme
- More scalable and efficient, but still requires multiple broadcasts at each tree level

Note: Broadcasts simplify handling cases where the number of processors is not a power of two
Performance

Irregular Data Communication
- Reduction schemes can improve performance, if implemented with care...
- Our algorithm is non-blocking, scalable, and uses full-duplex communication

Our Algorithm Improvements
- Maximize exchanges at each level without repeated calculations
- Reduce data volume at each level with full-duplex communication
- Minimum number of broadcasts are required to support an arbitrary number of processors
- Processors which do not contribute to the calculation at a given level are idle (not shown in this example)
Performance

Pentium III Beowulf Cluster vs. SGI O2K Migration
- Beowulf performs well, but network still dominates with increasing processors
- O2K is also affected for large (>30MB) messages

Note: New migration algorithms are applied
Myrinet 2000

2 Gbit/s Clos Topology

Our Configuration
- 32 port M3-E32 Switch
- PCI 64B, Lanai 9, SuperMicro 370 DLE
  Motherboard Serverworks LE rev 5 chipset
- RedHat linux 6.2 Kernel 2.2.19 SMP
- MPICH-GM 1.2.4 (latest 1.2.8)
- Dual-PE 800 Mhz Pentium-III 52 Node System

MPICH-GM Performance
- 225 MBytes/s (1.8 Gbit/s) 9.3 μsec latency
- 130 MBytes/s (1.0 Gbit/s) 1.5 μsec latency shared-memory communication
**Myrinet Performance**

**Artery Mesh Repartitioning**

- Improvement for larger numbers of processors compared to 100-BaseT Ethernet
- The is essentially a communication benchmark

![Performance Comparison Across Architectures for Artery Mesh Repartitioning](image)

![Network Performance for Ping-Pong Tests](image)
Myrinet Performance

Effects of Shared-Memory for Dual-CPU Pentium III

- Shared-Memory processors access data from cache quickly for small messages, but performance is cache limited.
- New PCI chipsets reduce the need for shared-memory communication (Our PCI Bandwidth is 455 Mbytes/s)
Myrinet Performance

Muzzle Refinement and Earthquake Generation Meshes
- Myrinet improved communication, but computation still holds back overall improvement
- Perhaps Pentium III bandwidth to memory?

Performance Comparison for 3 Adaptive Refinements of Muzzle-Brake Mesh

Comparison of Old and New Migration Algorithms Across Systems

Performance for 3 Adaptive Refinement Levels of an Earthquake Mesh
Myrinet Performance

Artery Refinement, Bisection, and the Cray T3E Network
- AMR performance varies based on partitioning, but Myrinet helps
- Bisection bandwidth drops with increasing numbers of processors (no shared memory)
- Cray T3E network remains the benchmark
Next Generation Features

- Development is User-Driven
  - Used for adaptive refinement of multi-scale meshes for active device modeling

Additional Work Directions
- User-controllable boundary zone definition
- Interpolation methods among mesh levels
- Straightforward approaches for incorporating error estimation
- Coarsening

Demonstration Release
- hpc.jpl.nasa.gov/APPS/AMR

Note: Functionality is limited in demo release