Analysis of Climate Data over Fast Networks
&
Parallel Tetrahedral Mesh Refinement

Mehmet Balman
Scientific Data Management Research Group
Lawrence Berkeley National Laboratory

July 18th, 2011

CScADS 2011

Climate Analysis

Applications:

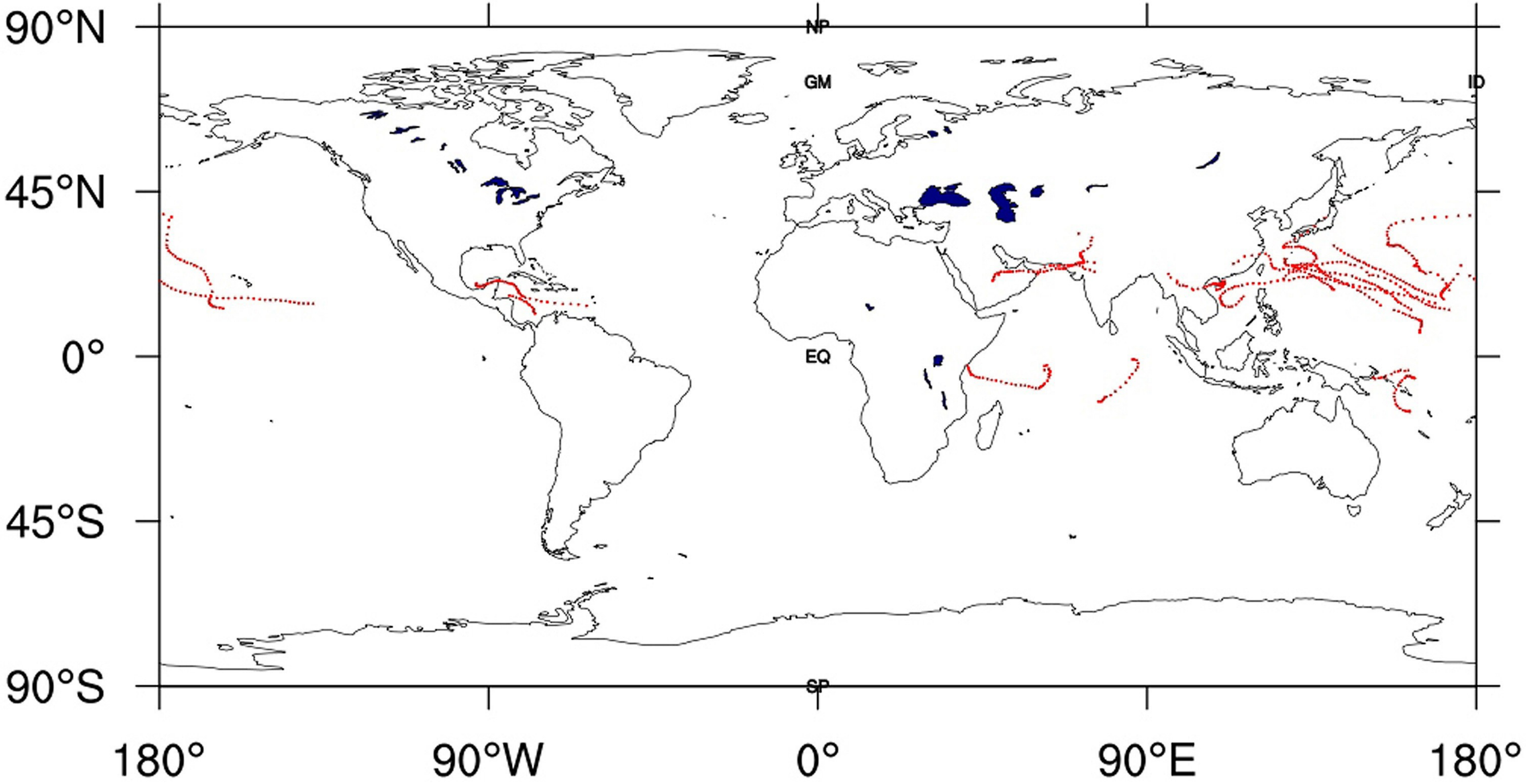
Embarrassingly parallel

- Detecting Tropical Storms (compute intensive)
- Atmospheric Rivers (data intensive)

- Earth System Grid Center For Enabling Technologies (ESG-CET)

  - NetCDF data files
  - Fortran / C
  - Python scripts (visualization, etc.)

Tropical Storms (from fvCAM2.2 simulation encompassing 1979-1993)



Credit: Daren et al. CloudCom2010

Data Access & I/O Pattern

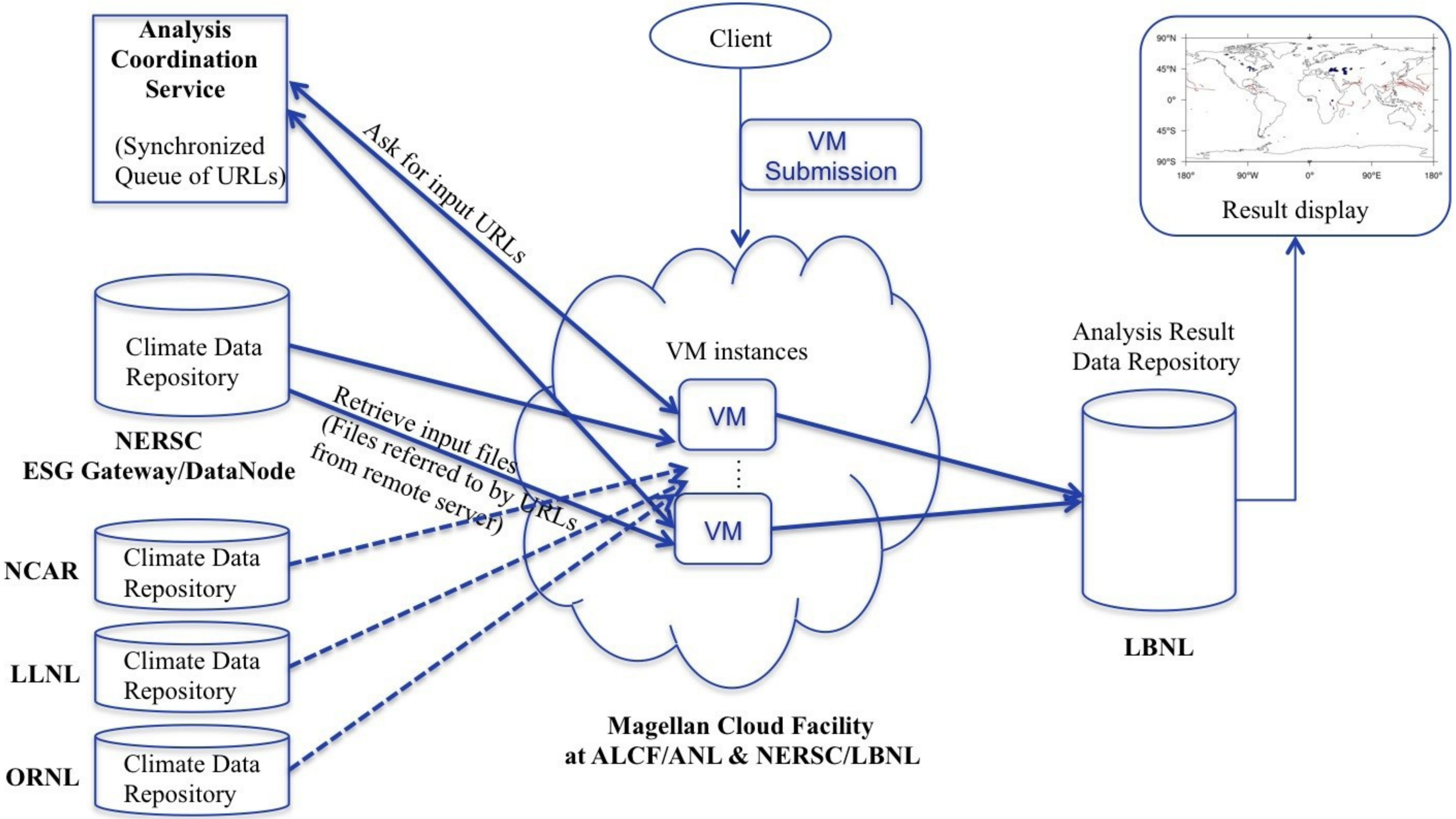
- Distribution of input data files (file dispatcher)
  - Input files: ~2G, 0.5G each (total several TBs, will be PBs)
  - One file per process
  - Join results for further analysis

- Batch Processing (Linux Cluster, Grid)
  - Retrieve data files over the network

- Cloud (NERSC Magellan, Eucalyptus)
  - Retrieve data files over the network

- MPI (for file coordination) (Franklin, Hopper)
  - Read files directly from the file system (GPFS)

Job Submission & Remote Data Access



Credit: Daren et al. CloudCom2010

Remote Data Repositories

- Challenge:
  - Petabytes of data
    - Data is usually in a remote location
    - How can we access efficiently?

- Data Gateways / Caching?
  - I/O performance?
  - Scalability (using many instances?)
    - Network Performance, Data Transfer protocols

Benefit from next generation high bandwidth networks

Adaptive Mesh Refinement

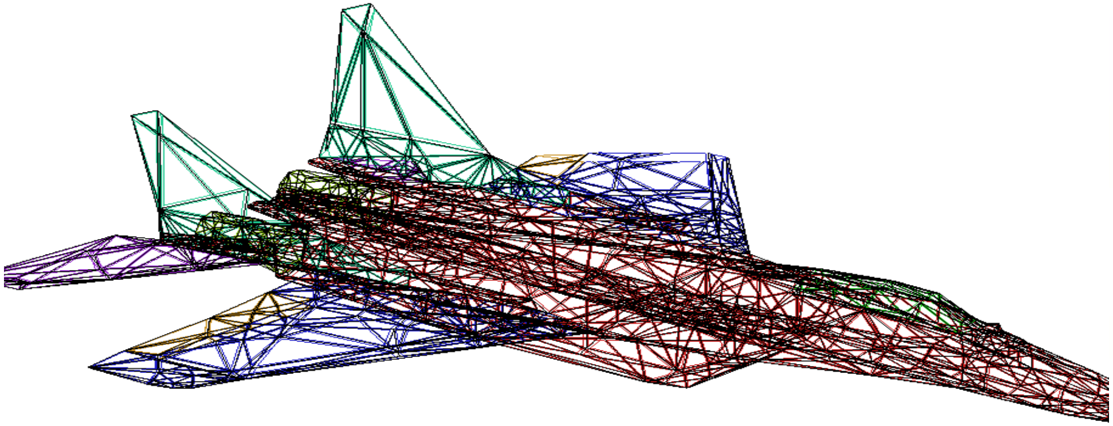
Parallel Tetrahedral Mesh Refinement (PTMR)

- Problem size and computational cost grow very rapidly in *3*-dimensional refinement algorithms

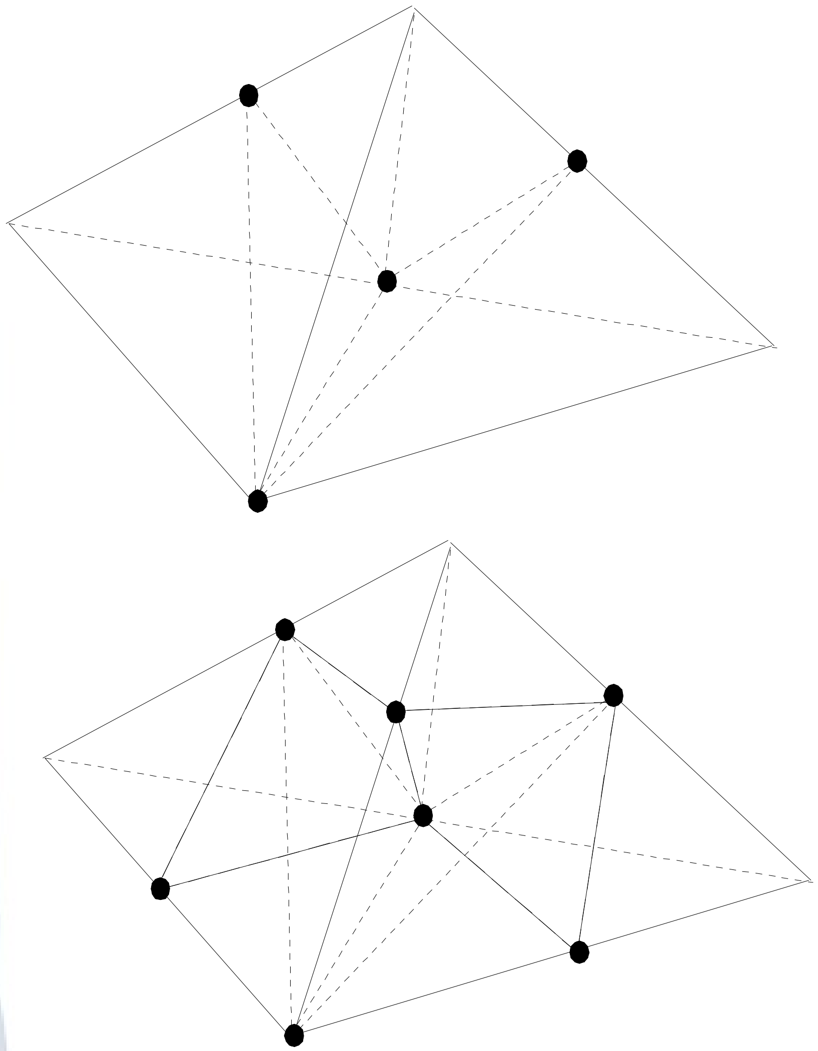
- Process very large data
- Distribute load over multiple processors

- Longest-edge bisection

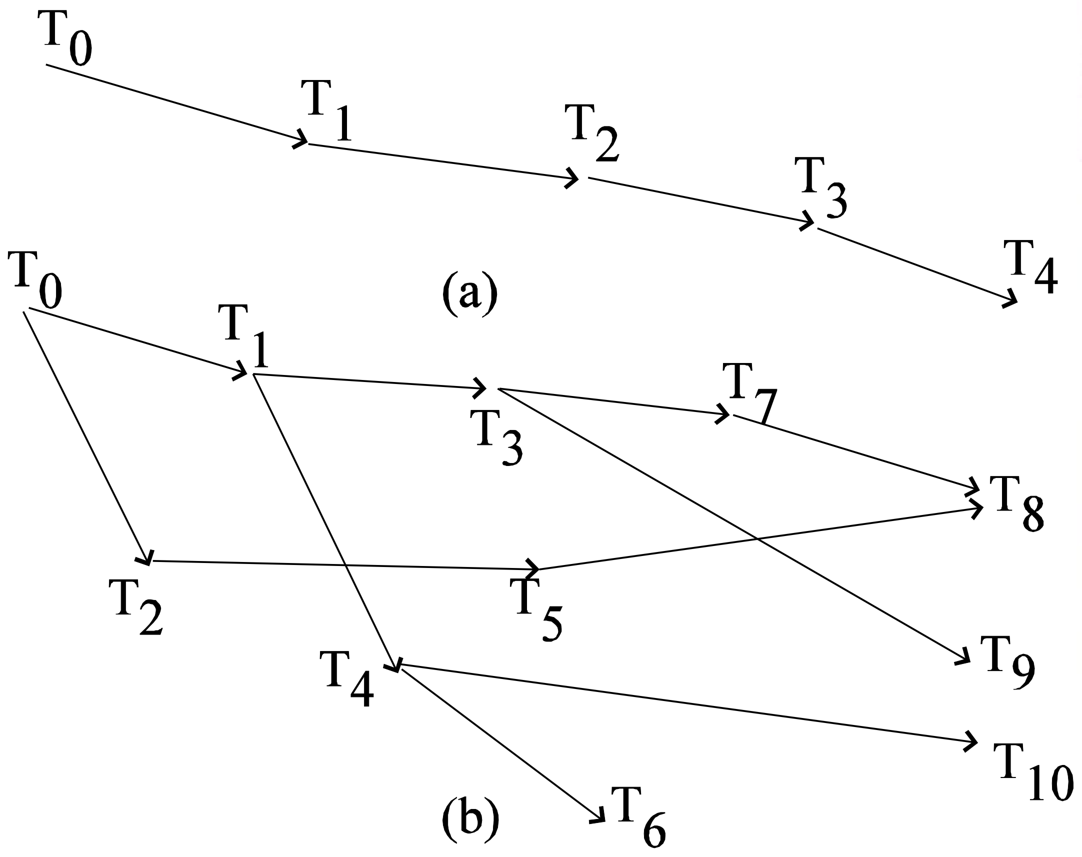
- Skeleton algorithms (8-Tetrahedra LE)



Propagation Path

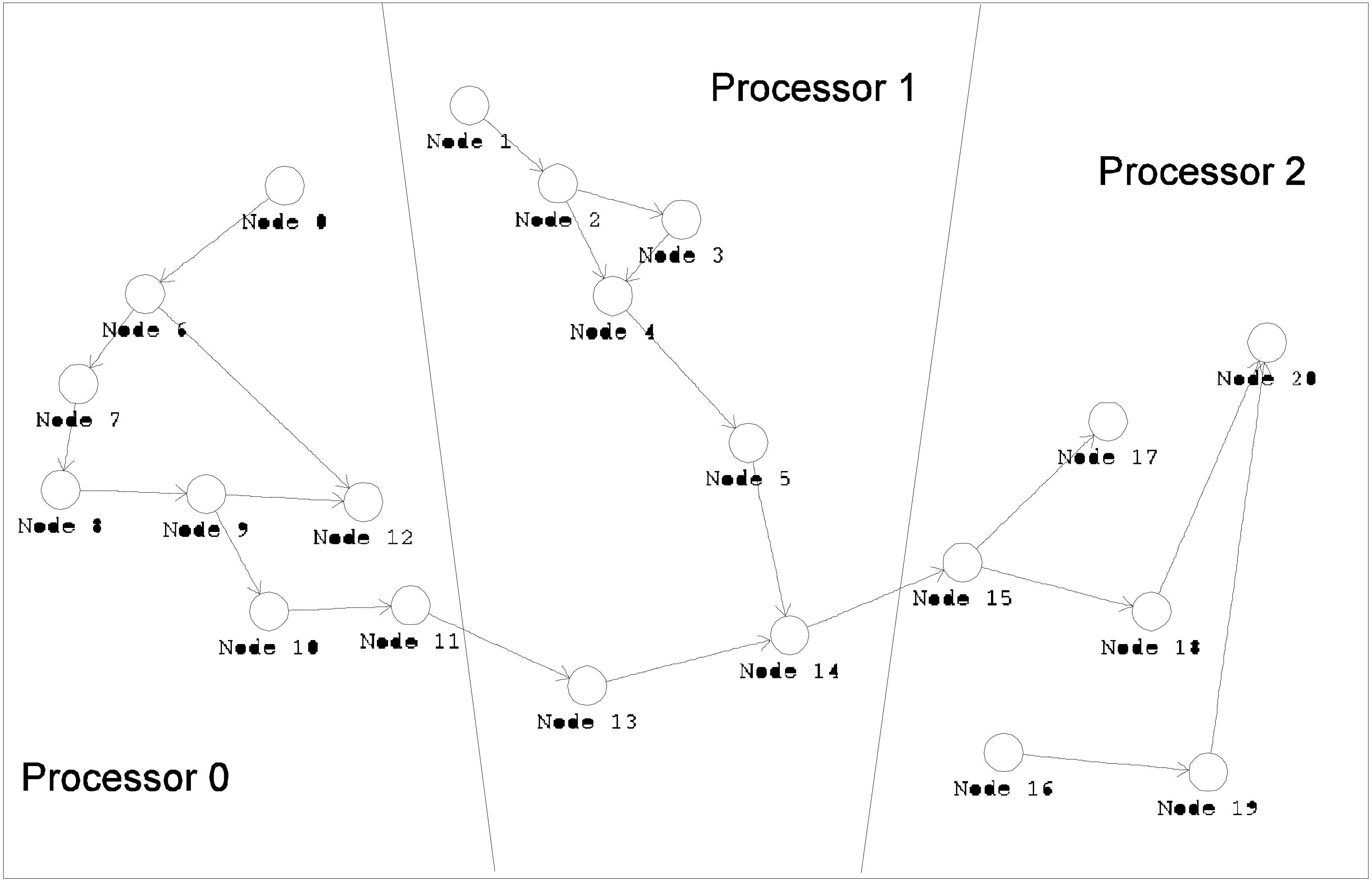


8-T LE refinement

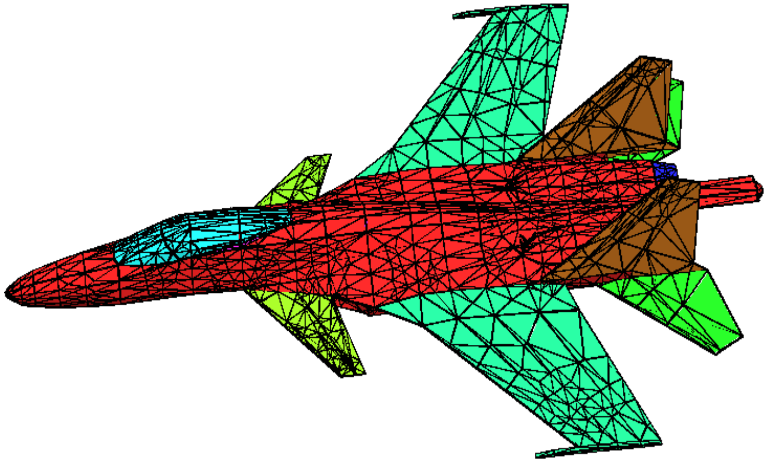


(a) 2-D
(b) 3-D

LEPP-Graph Partitioning & Synchronization



PTMR

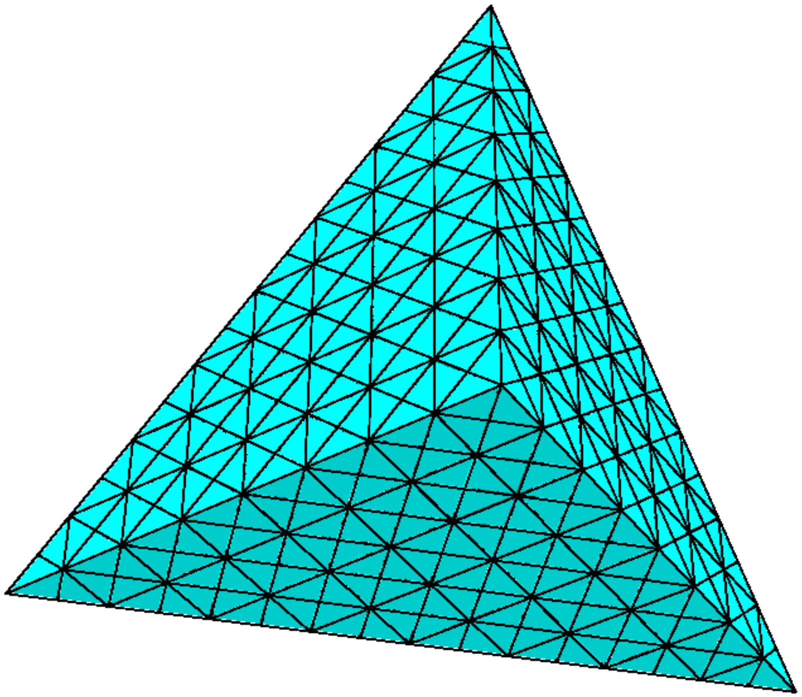


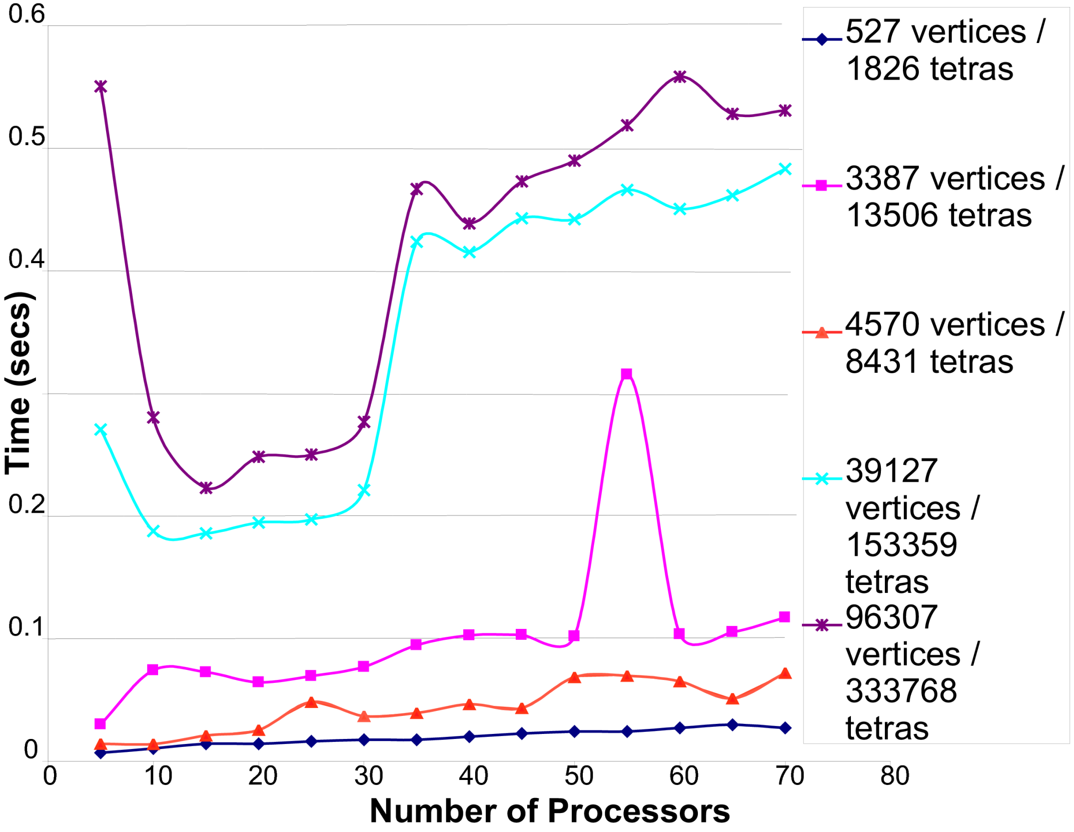
- Implemented using MPI
- I/O: only processor 0 (head node)
  reads/writes (input file, output file)
- Mesh structure is distributed among processing nodes (load balancing)
- Each process handles its local mesh data
- Synchronize local propagation paths (data is distributed)
  - Each process informs other processing nodes whether a border element in the local partition is selected

PTMR performance

- Processor 0 (head node) acts as a gateway
  - Each message sent through the gateway
  - Gateway aggregates messages

Reduce the number of MPI messages (improves overall performance)




0.6
0.5
0.4
0.2
0.1
0
Time (secs)
0
10
20
30
40
50
60
70
80
Number of Processors
527 vertices / 1826 tetras
3387 vertices / 13506 tetras
4570 vertices / 8431 tetras
39127 vertices / 153359 tetras
96307 vertices / 333768 tetras