“Raw” and Uncut…
An MxN Socket Multiplexer!

(a.k.a. What Dr. Frankenstein did whilst writing SciDAC proposals… :-)

March 15, 2001

Jeeeembo Kohl, David Bernholdt ~ ORNL

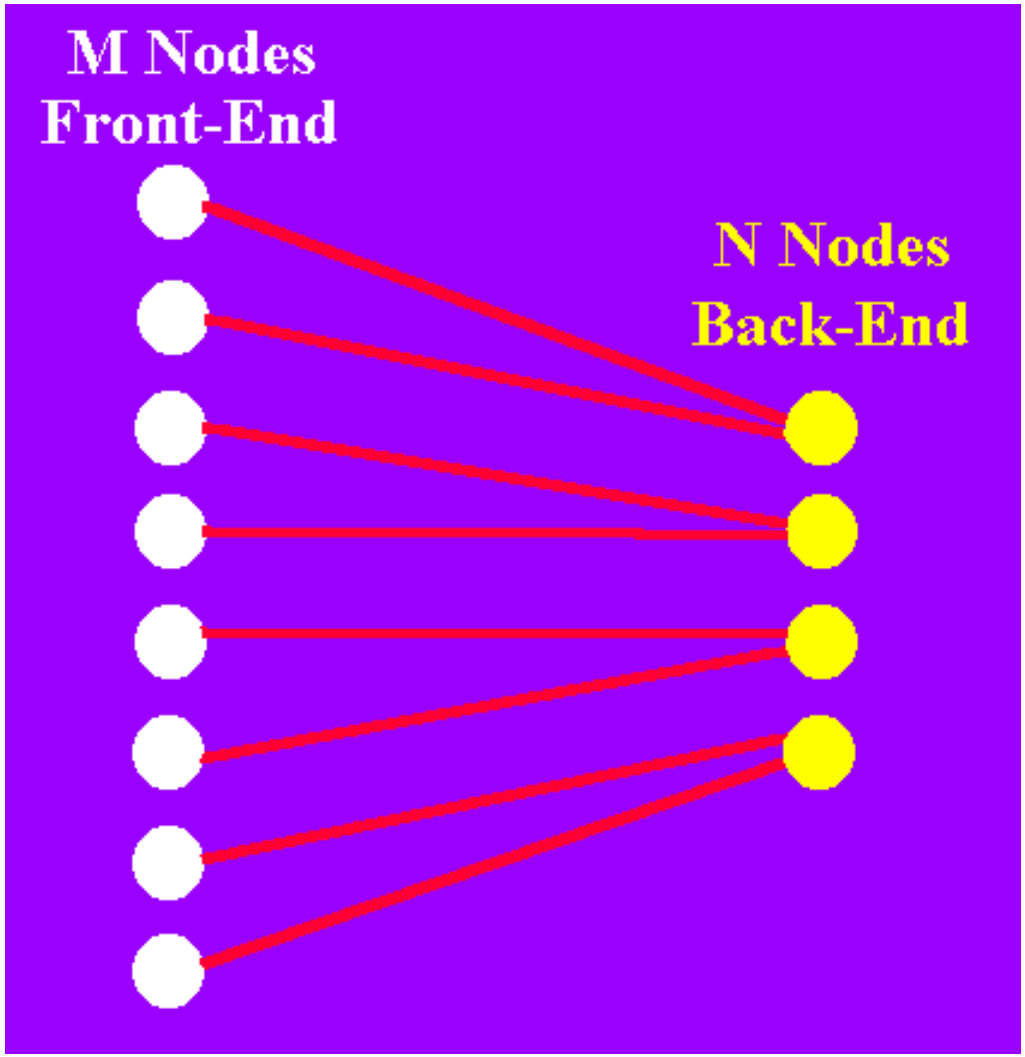
Ben Allen ~ SNL

Research supported by the Mathematics, Information and Computational Sciences Office, Office of Advanced Scientific Computing Research, U.S. Department of Energy, under contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.

What Evil Lurks…

- Hook ASCI White to Sandia C-Plant Cluster
  - ⇒ Download Parallel Data to "Back-End"
  - ⇒ For Data Reduction, Analysis and Viz…
- Use Standard TCP/IP (for now… ☺)
  - ⇒ Hook Together 2 MPI Comm_Worlds
  - ⇒ Could Ultimately Port to MPI-2 / PVM
- *Unidirectional* Parallel Data Collection
- Why? Because They're Paying Us To Do It. ☺

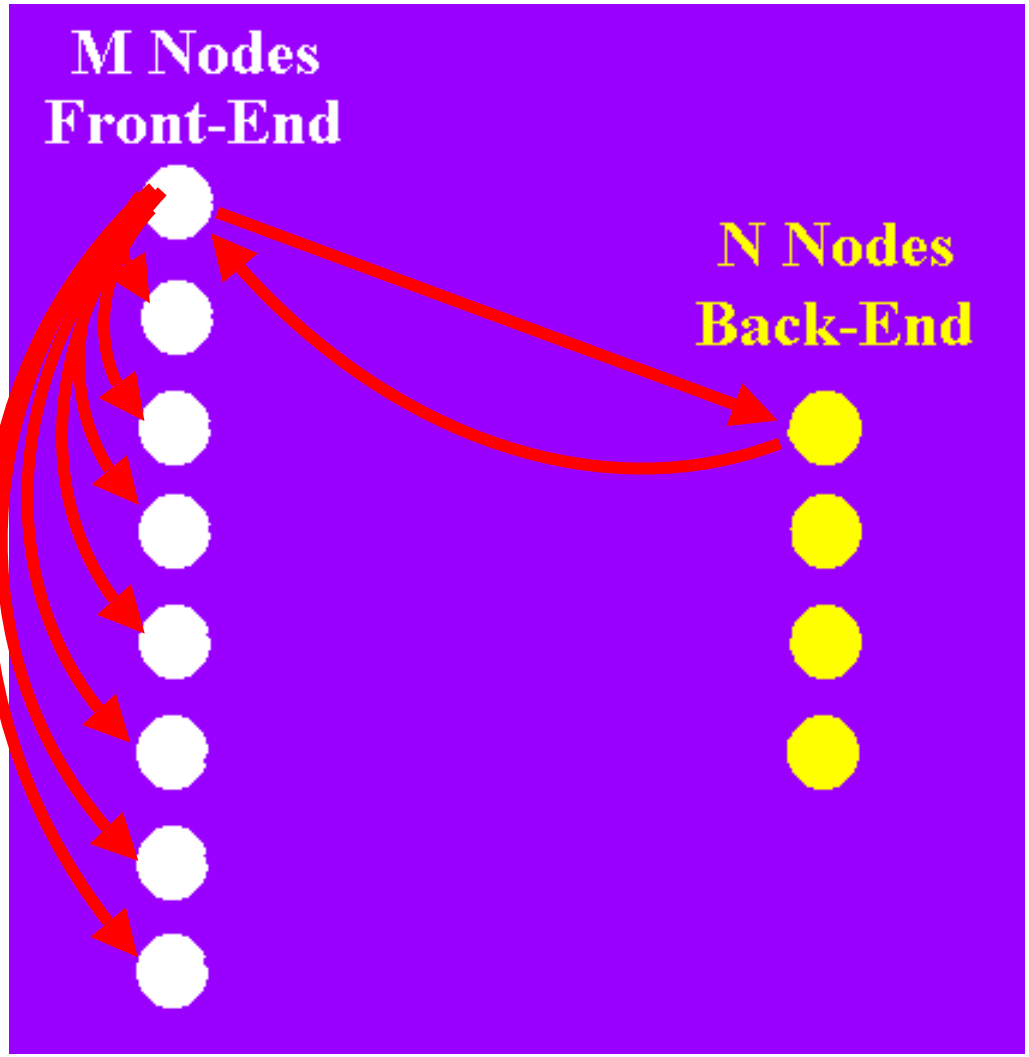
Approach



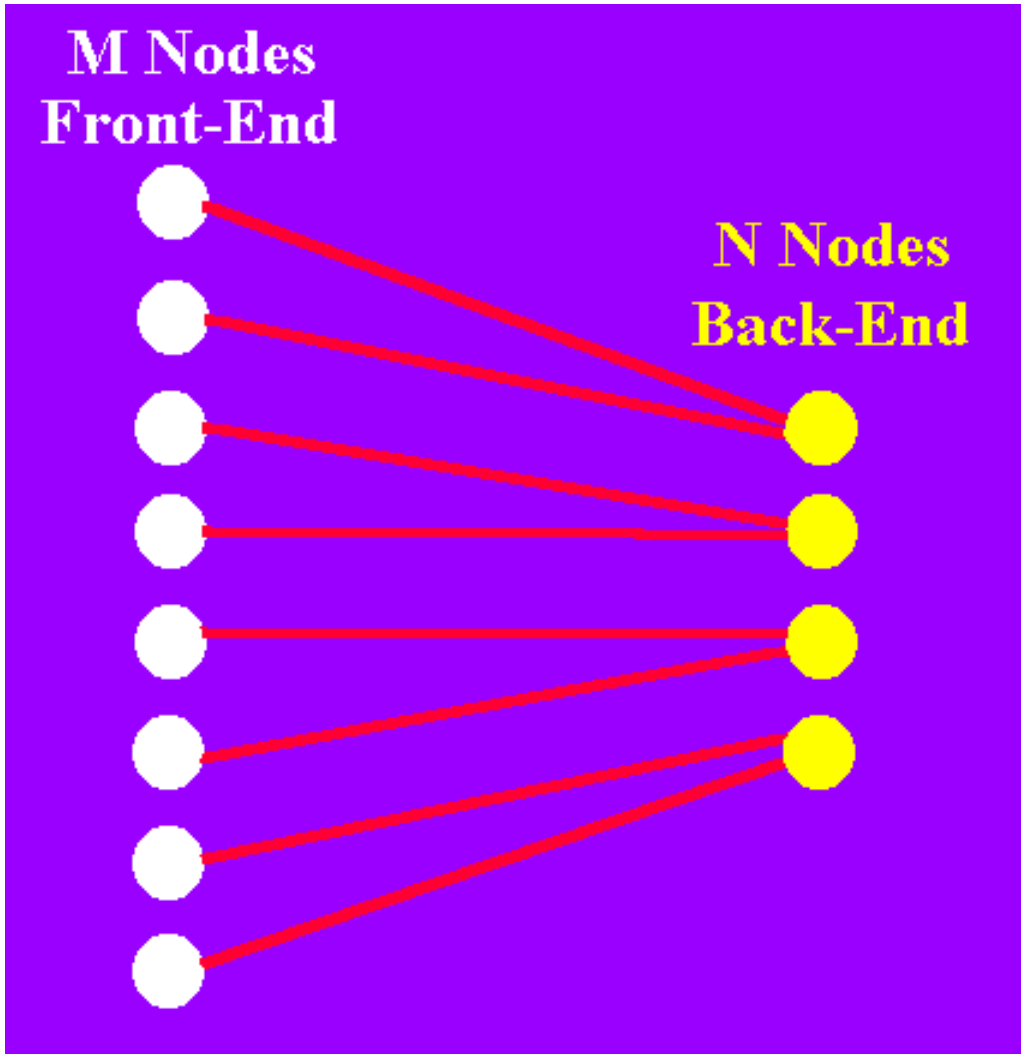
“I Have A Plan!”

- Front-End “M” Machines
  - ⇒ Read a Configuration File / Env Var at Startup
  - ⇒ Appl “Master” Attaches to Multiplexer (Socket)
- Back-End “N” Machines
  - ⇒ Multiplexer “Master” Accepts Connections
  - ⇒ Returns M-to-N Mapping Info to Appl Master…
- M Appl Tasks Connect to N Multiplexer Tasks

Attachment Protocol



Attachment Protocol



Back-End Data Handling

- Like Read-Only, Read-Once "File System"
  - $\Rightarrow$ No Data Interpretation
  - $\Rightarrow$ Just "Raw" Data Coordination
- Therefore, $M \geq N$! Else…
  - $\Rightarrow$ Map Only to First M of N Back-End Nodes
  - $\Rightarrow$ Replicate Data Streams to Multiple Back-End?
  - $\Rightarrow$ No Semantic Subdivisions Possible in Mux…

Front-End Interface (SPMDOutput)

```
interface SPMDOutput {
  /** Map the client to one of the server-side sockets. **/
  int setupConnections();

  /** Refresh the connections, if need be.  (ReleaseSocket() First If Connected) **/
  int updateConnections();

  /** Fetch the connection.
      @param socket ~ Output slot for the socket descriptor. **/
  int getSocket( int &socket );

  /** Release socket when done
     @param socket ~ Value returned from getSocket. **/
  void releaseSocket( int socket );
};
```

Back-End Interface ("Seminude")

```
interface SPMDInput {
  /** Set up socket so that client can connect.  Client tells us
    how many (M), we parcel out sockets on the N of us. **/
  void serveConnections();

  /** @param sockArray ~ Output array of  Read sockets.
      @param arraySize Output the number of sockets.
  int getSockets( const int *& sockArray, int & arraySize );
};
```

Gory Details

- Actually Use Real Sockets for 1st Prototype!
- Later Maybe PVM, MPI-2…?

  ⇒ Need True "InterCommunicator" Between MPIs
- Next Step ~ Wrap Up as CCA Components?
- Base Functionality Should be done soon…

  ⇒ Ben Says "2 Day Job"… heh… ☺