THE UNIVERSITY of NEW MEXICO

Towards an Autonomous MRNet

Dorian Arnold
Department of Computer Science
University of New Mexico

Paradyn/Dyninst Meeting
University of Maryland, College Park
April 27-28, 2009

My Research Goals

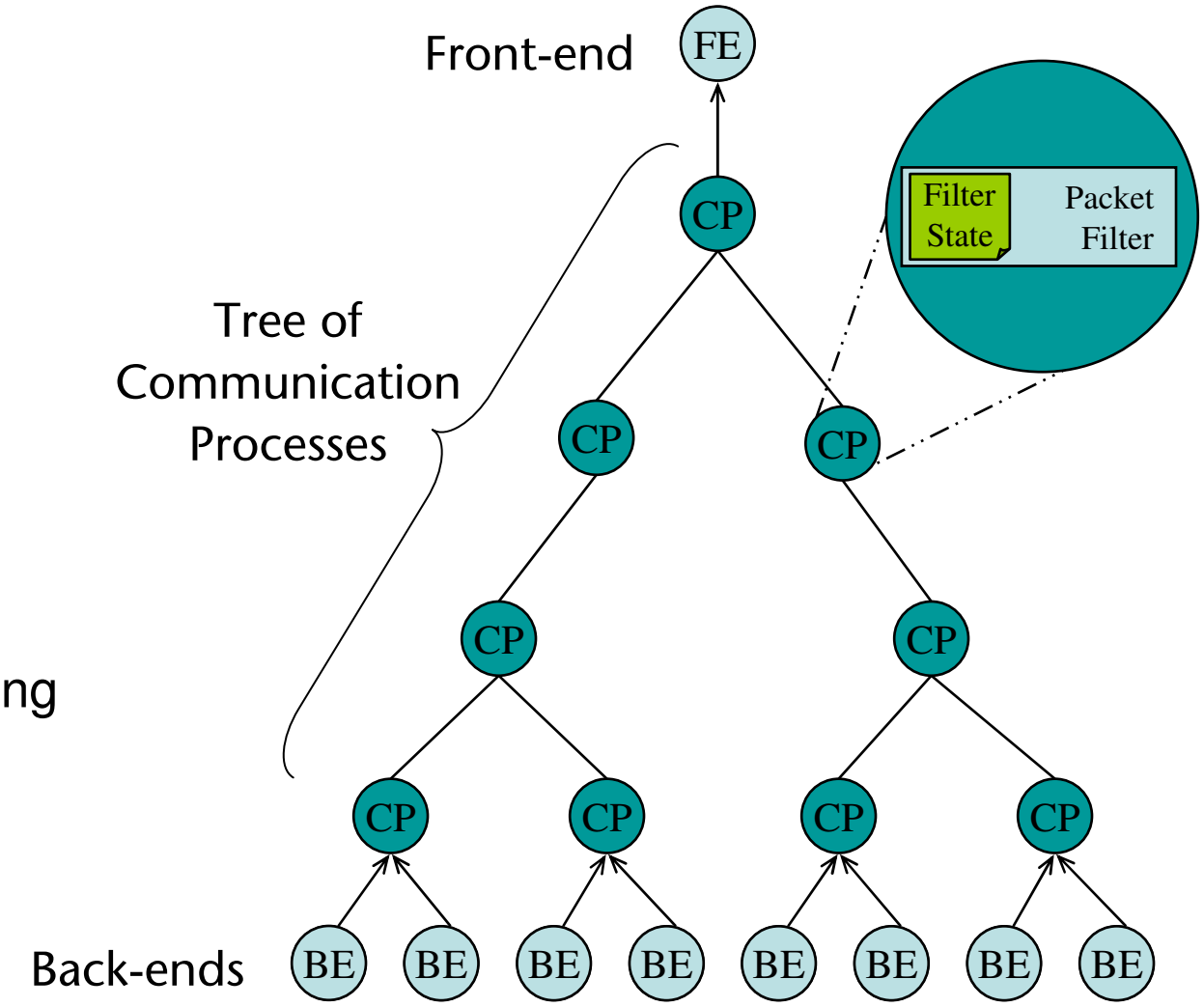
Make HPC systems easier to use without sacrificing efficiency or reliability

- Scalable computational models for applications and tools
  - Communication
  - Data analysis
  - HPC middleware
- Autonomous systems (overlay networks)
  - Self-monitoring
  - Self-reconfiguring
  - Self-healing
  - Self-optimizing
- Fault-tolerance
  - New reliability models & mechanisms
- HPC Tools

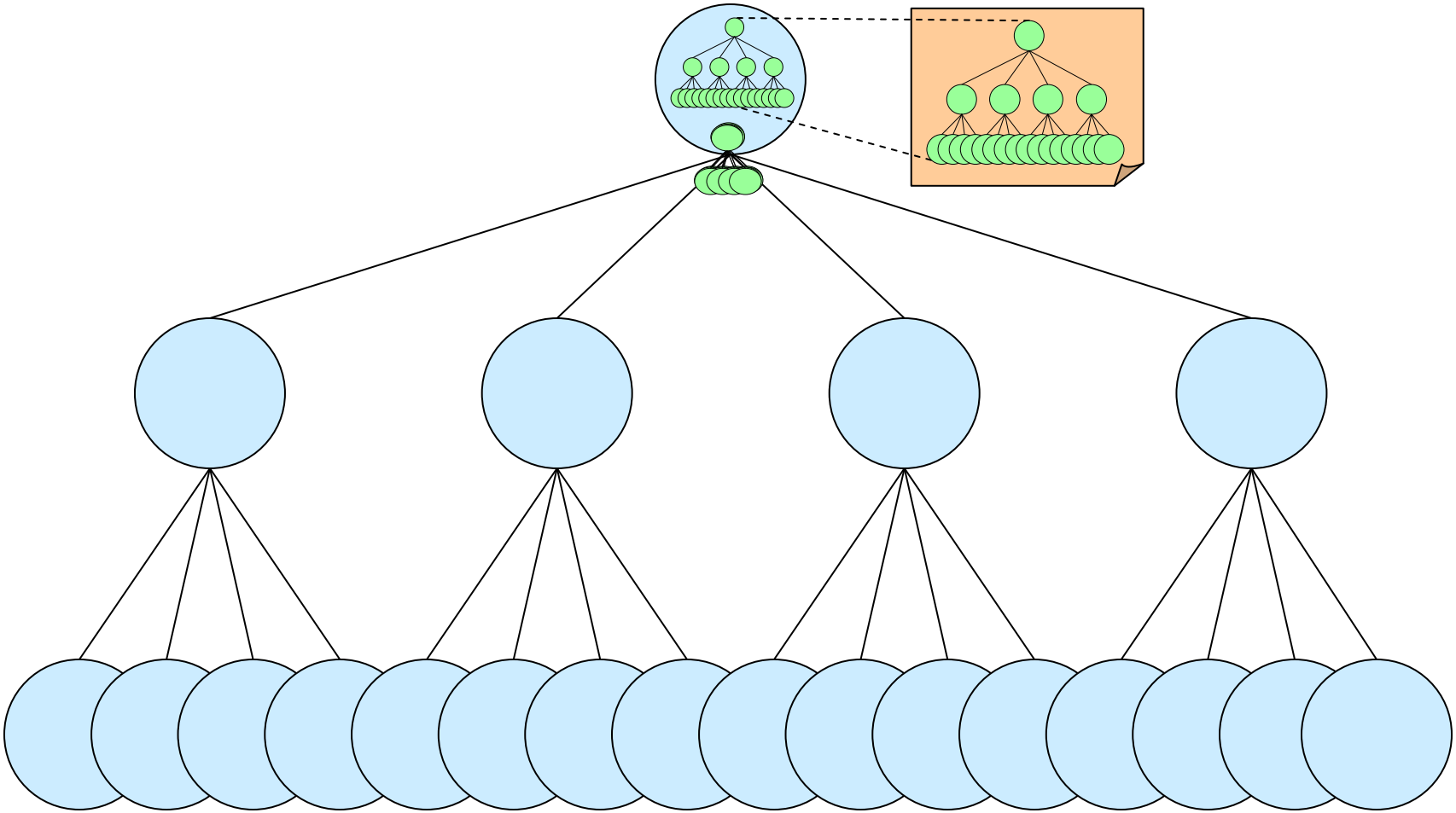
Let experts operate within their fields of expertise!

In the beginning was MRNet

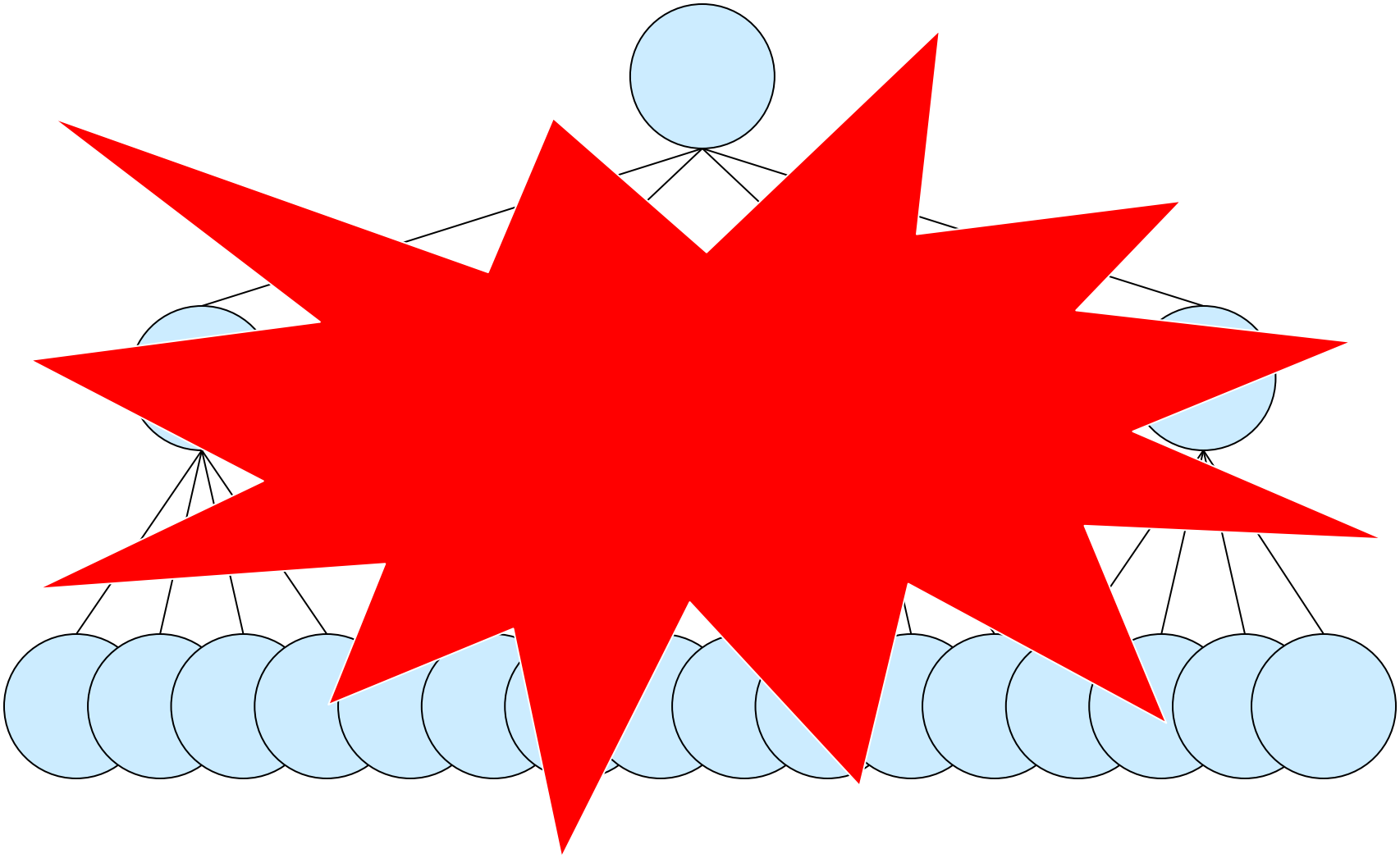
- Scalable data multicast and aggregation
- Flexible topologies
- User-defined filters
- Trade-off: extra processing nodes for performance



and MRNet only supported static topologies



and MRNet did not tolerate failures



And then there was MRNet 2.0

- Event detection service
  - Failure detection
  - Dynamic topology configuration
  - New MRNet instantiation protocol
- State composition for failure recovery

Today's Talk

- Towards an autonomous MRNet
  - Self-monitoring
  - Self-healing
  - Self-configuring
  - Self-optimizing

MRNet Event Detection Service

Event Detection Service (EDS) thread

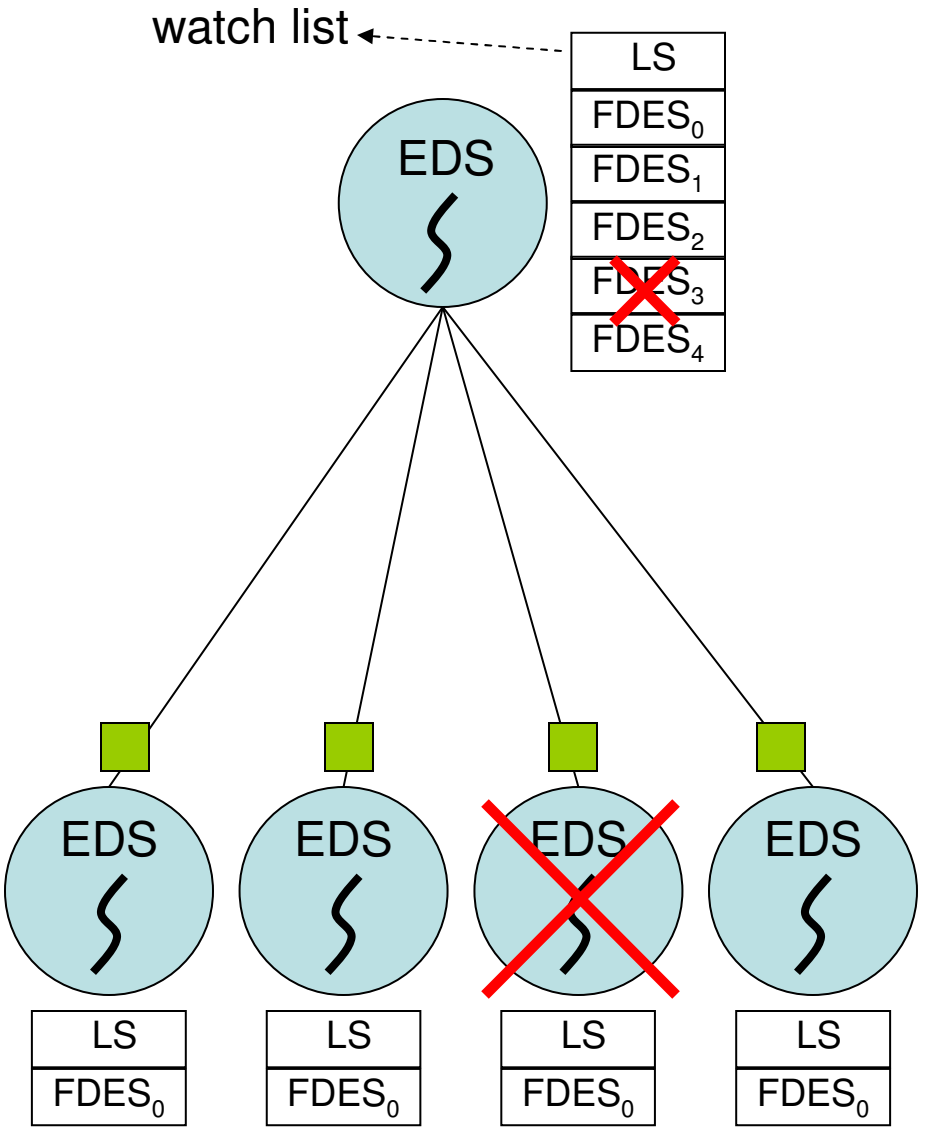
- In each MRNet process

- Passive detection of asynchronous events
  - Failure events for failure detection
  - Connection events for dynamic reconfiguration

- Connection-based (TCP) mechanisms
  - Monitor *watch list* of *event sockets*
    - Listening socket
    - *New Failure Detection Connection* protocol message
    - *New Data Connection* protocol message

MRNet Self-monitoring: Detecting Functional Failures (cont'd)

- Each process monitors its peers (parent and children)
- Connect to peer EDS
- Send `New Failure Detection Connection` message
- Add failure detection event sockets to watch list
- Socket error → peer failure



Upon Failure Detection …

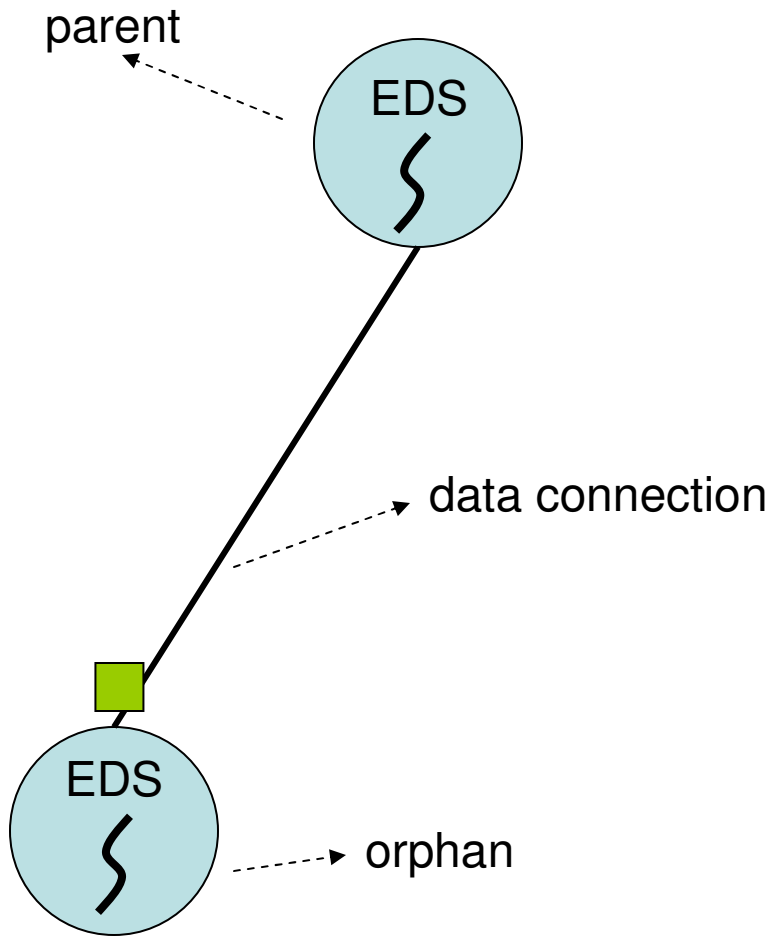
1. The MRNet tree must be reconfigured

2. MRNet must recover any lost state
   (that it can)

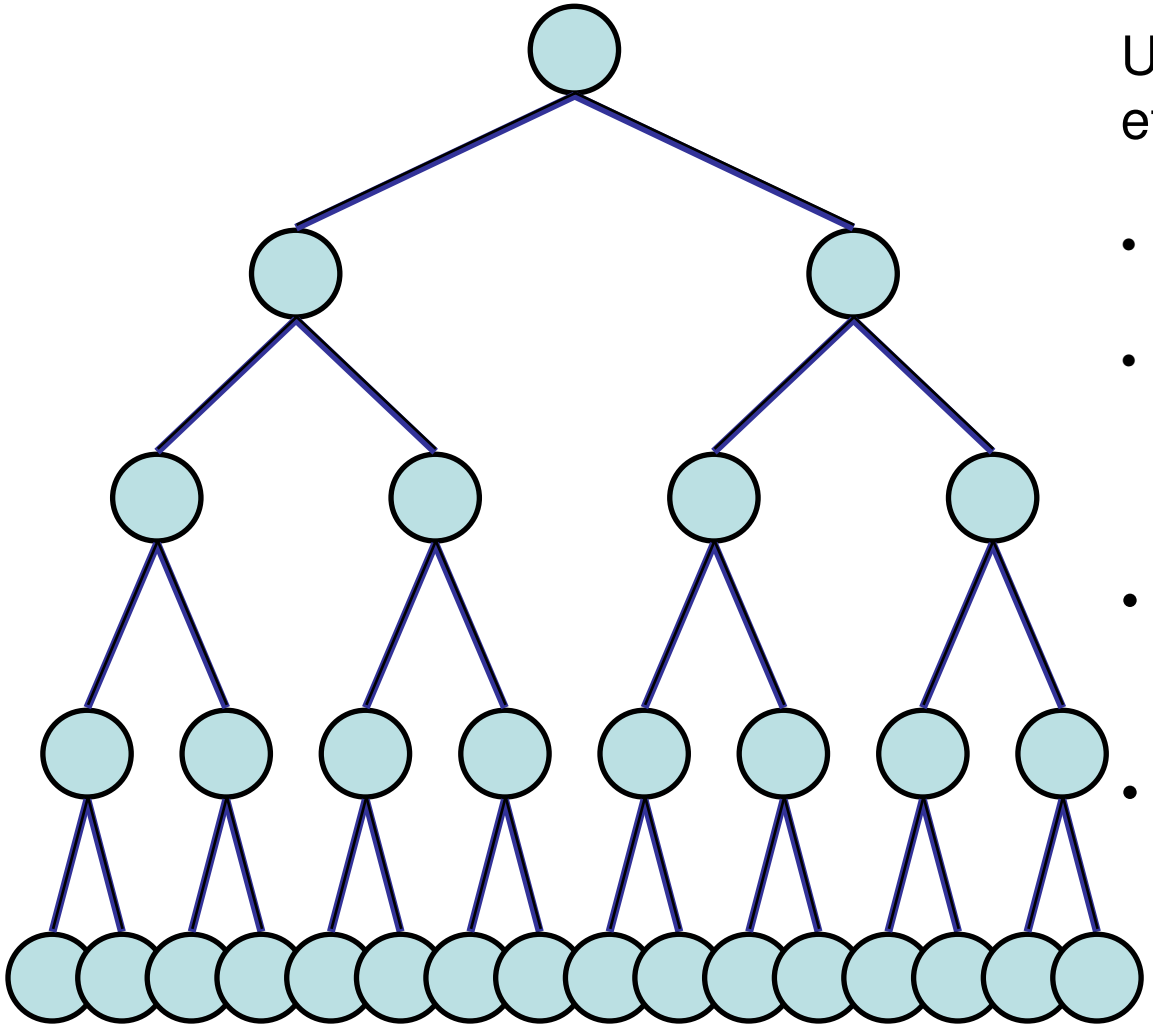
MRNet Self-healing: Dynamic (Re)configuration

At initialization or after failures, orphan connect to new parent's EDS

- Send `New Data Connection` protocol message
- Child/parent establish data socket



Information Dissemination



Use tree structure for efficient global dissemination

- Failure report:
  - 32 bits: {failed rank}
- Reconfiguration report:
  - 64 bits: {child rank, parent rank}
  - Disconnected subtrees intact
- Disseminating process sends to parent and children
- Receiving processes send to peers other than source

MRNet Self-healing: State Recovery

- State Compensation
  - compensate for lost state using inherently redundant information from surviving processes
  - Avoid overhead of explicit data replication
- State Composition
  - Lightweight mechanism for idempotent aggregations
  - Reintegrated orphans send filter state to new parent

State Composition Interface

```
outPacket get_FilterState( void ** inFilterState );
```

- Inputs pointer reference to stream's filter state
- Outputs "packetized" version of filter state

```
int load_FilterState( const char * inSharedObject
                      const char * inFilterFunction );
```

- Used to dynamically load new filter functions
- Also queries for `get_FilterState` routine
  - If found, filter is *recoverable*

What's Next?

Can overlay networks for high performance data analysis dynamically and autonomously adapt to application-specific, time-varying workloads?

… without user/application input

- H/w characteristics
- Performance monitoring
- Performance modeling
- Decision processes

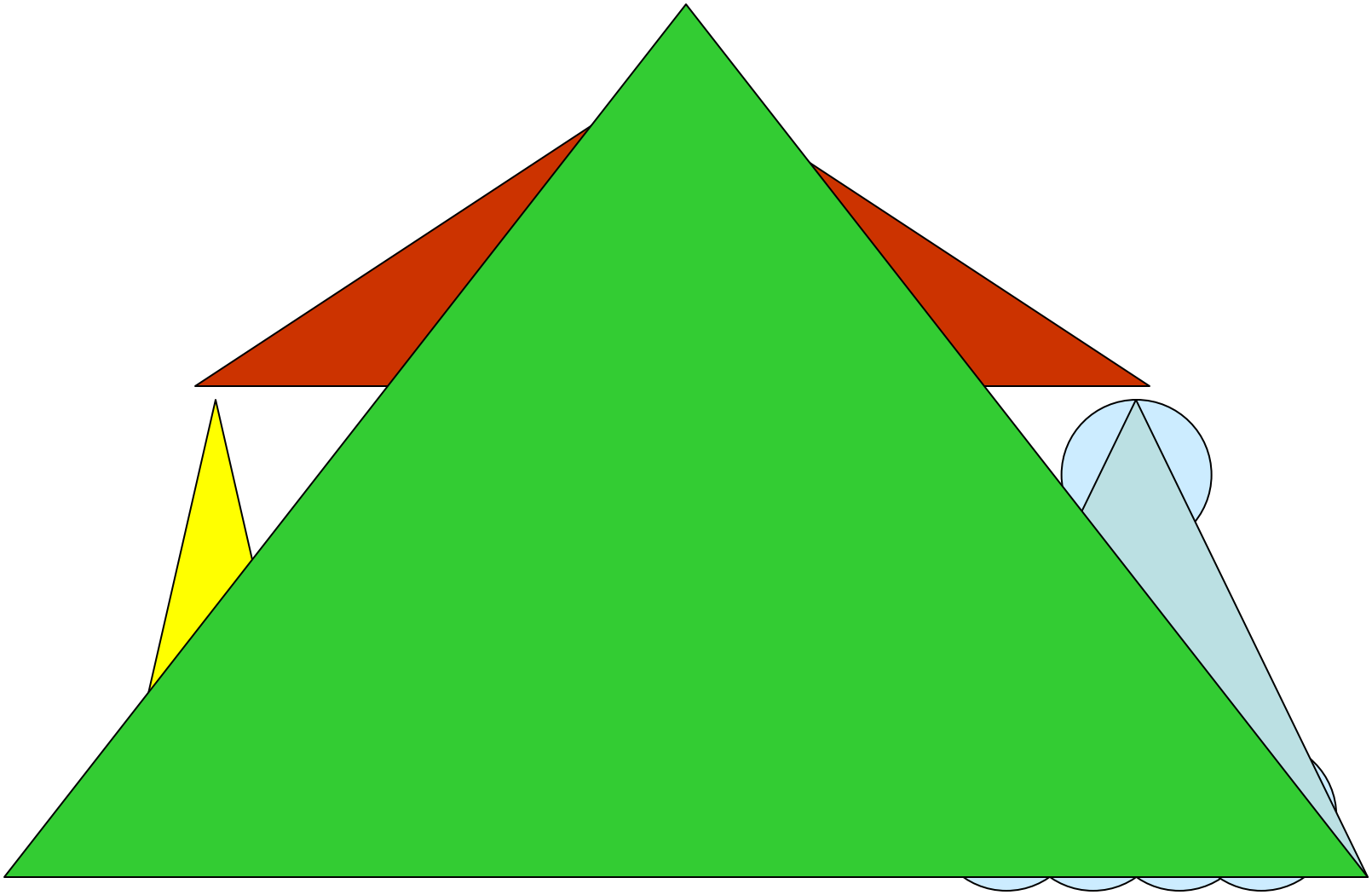
What's Next?

Can overlay networks for high performance data analysis dynamically and autonomously adapt to application-specific, time-varying workloads?

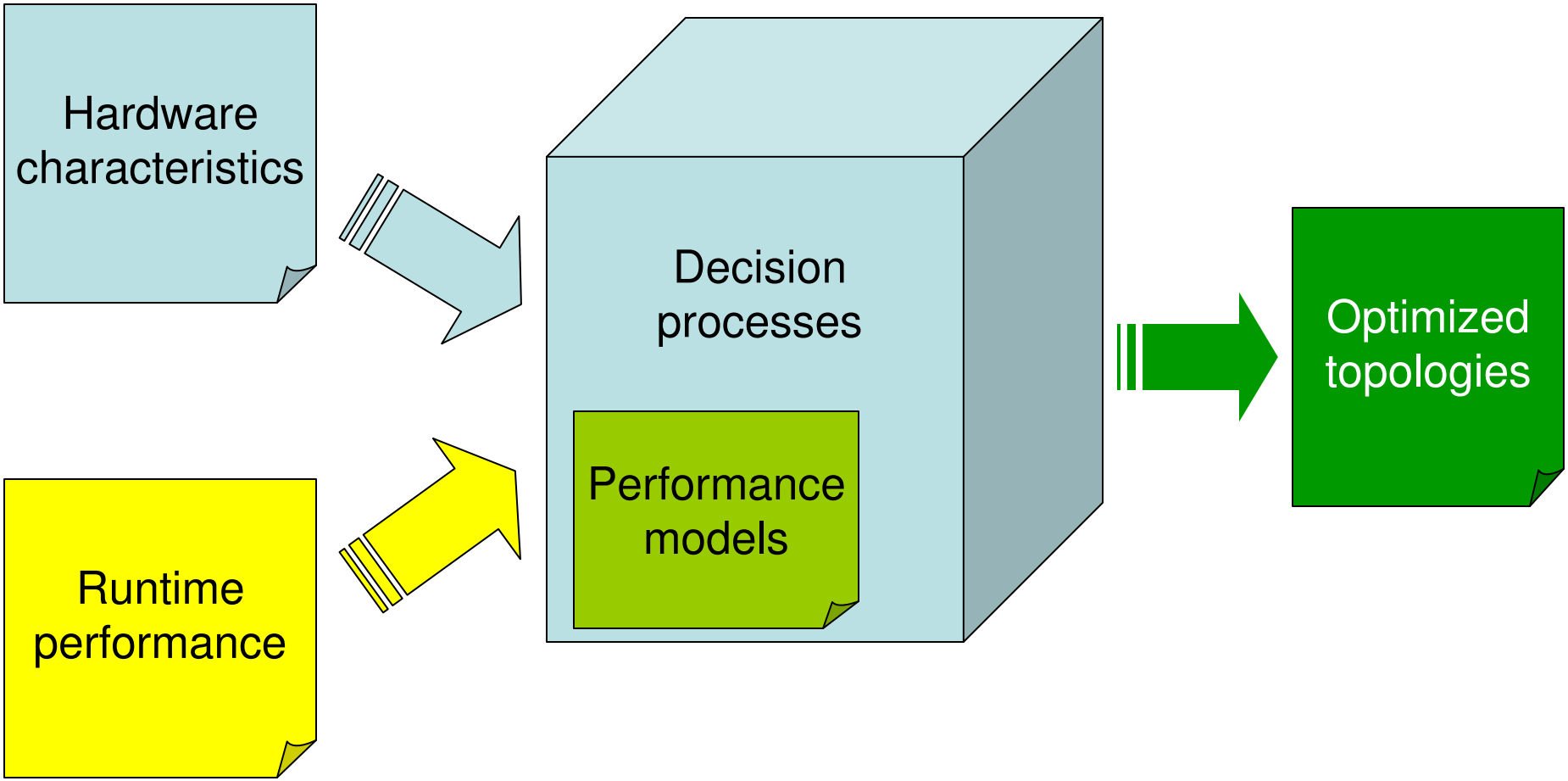
… with possible quality-of-service constraints

- Priority
- Performance
- Reliability
- …

Hierarchical Modeling



MRNet Self-optimization



Other Issues

- Monitoring infrastructure
- QoS
- Many MRNets
  - Simplicity
  - Protection
  - No interference
  - Ease of deployment

or 1

  - Faster startup
  - Better utilization
  - Help address collocation problems

What this means to you

- Simpler, yet better, MRNet
  - Doing (much) more with less!
- An excellent vehicle for data-intensive information analysis
  - Tools and applications!

Questions?