A Fault Tolerant Spacecraft Supercomputer to Enable a New Class of Scientific Discovery

Part 2: The Application Cluster

Supercomputing 2000
November, 2000
R. Ferraro, D. Katz, A. Silliman,
Jet Propulsion Laboratory
The REE Vision:

**Past**
- Raw data sent to Earth for processing

**Future**
- Moving high performance computing and applications to the spacecraft
- Health status and "interesting" result transmitted to Earth
Chip flight qualification cycle

- **Launch**
- Flight system
- Engineering model
- Gov procurement
- Delay in licensing
- New chip on market

Timeline:
- 0 years
- 1 year
- 2 years
- 3 years
- 4 years
- 5 years
Chip flight qualification cycle

REE Launch
Flight system
Engineering model
Gov procurement
Delay in licensing
New chip on market

0  1  2  3  4  5
years

13
Chip flight qualification cycle

REE Launch
Flight system
Engineering model
Gov procurement
Delay in licensing
New chip on market

0  1  2  3  4  5
years
Five Science Application Teams Chosen to Drive Requirements and Demonstrate Benefits of Onboard Computing

Next Generation Space Telescope - John Mather/GSFC
  • Onboard Cosmic Ray correction to the data
  • Autonomous control and optimization of the adaptive optics

Gamma ray Large Area Space Telescope
  • Peter Michelson/Stanford
    • Onboard cosmic ray rejection
    • Real time gamma ray burst identification

Orbiting Thermal Imaging Spectrometer - Alan Gillespie/U Washington
  • Onboard Atmospheric corrections, Radiance calculations

Mars Rover Science - R. Steve Saunders/JPL
  • Autonomous optimal terrain navigation
  • Autonomous Field Geology

Solar Terrestrial Probe Program - Steve Curtis/GSFC
  • Constellation/Formation Flying missions to probe the Sun-Earth Connection
  • Onboard Plasma moment calculations, multi-instrument cross correlations, autonomous operations
REE First Generation Testbed

Standard VME Chassis

20 REE Node Modules

Expansion Slots
(Spare Node Modules or Standard VME Modules)

Myrinet Network

Standard VME Backplane

REE Testbed

Remote Users
The SIFT Recovery Hierarchy

The Core Cluster

The Application Clusters

SIFT Layer
application manager
and application
services

Local Fault Detection using
Algorithm-based
fault tolerance
Microcomputer Processor History
The SIFT Recovery Hierarchy

The Application Cluster

- SIFT Layer
  - application manager
  - and application services

- Local Fault Detection using Algorithm-based fault tolerance
Image Texture Application

Frequency and orientation tuned filters convolved with image to produce Feature Vectors

Vectors for each pixel near each other in feature space are grouped together into the same cluster.
Register Faults: 0
Memory Faults: 8
Total Restarts: 0
ABFT Not in Use

Progress v Time

Image 2 Labels
Example of application processing with no fault tolerance

<table>
<thead>
<tr>
<th>Node #</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application manager started</td>
</tr>
<tr>
<td>2</td>
<td>Application process 1 started</td>
</tr>
<tr>
<td>3</td>
<td>Application process 2 started</td>
</tr>
</tbody>
</table>

Rock 1 clustered image complete
Register Faults
0
Memory Faults
5
Total Restarts
0
ABFT Not in Use

Progress v Time

Image 1 Labels
Example of application processing with no fault tolerance

<table>
<thead>
<tr>
<th>Node #</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application manager started</td>
</tr>
<tr>
<td>2</td>
<td>Application process 1 started</td>
</tr>
<tr>
<td>3</td>
<td>Application process 2 started</td>
</tr>
</tbody>
</table>

Rock 2 clustered image complete
Register Faults
0
Memory Faults
8
Total Restarts
0
ABFT Not in Use

Progress v Time

Image 2 Labels
Example of application processing with no fault tolerance

<table>
<thead>
<tr>
<th>Node #</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application manager started</td>
</tr>
<tr>
<td>2</td>
<td>Application process 1 started</td>
</tr>
<tr>
<td>3</td>
<td>Application process 2 started</td>
</tr>
</tbody>
</table>

Job complete
Register Faults
0
Memory Faults
11
Total Restarts
0
ABFT Not in Use

Progress v Time

Image 3 Labels
• Algorithm-Based Fault Tolerance (ABFT) can detect, locate, and correct faults by exploiting structure of linear numerical operations

• Wrap operation (from, e.g., ScaLAPACK, FFTW) in ABFT shell, as above
  – Protects against faults to data (cache, registers, FPU) during the main computation

• Checksum of inputs is used to enforce a necessary condition on result’s checksum
Register Faults
0
Memory Faults
13
Total Restarts
0
ABFT in Use

Progress v Time

Image 1 Labels
Register Faults
0
Memory Faults
27
Total Restarts
2
ABFT in Use

Progress v Time

Image 2 Labels
Register Faults: 0
Memory Faults: 35
Total Restarts: 2
ABFT in Use

Progress v Time

Image 3 Labels
Simulate node failure
Register Faults
0
Memory Faults
23
Total Restarts
0
ABFT Not in Use

Progress v Time

Image 1 Labels
Register Faults
0
Memory Faults
55
Total Restarts
0
ABFT Not in Use
Register Faults
0
Memory Faults
56
Total Restarts
3
ABFT Not in Use

Original Photo 3

Progress v Time

Image 3 Labels
Where are we?

- Functional testbed system has been established
- Demonstrated that several fault tolerant techniques work effectively
- Over the next 18 months many more techniques will be investigated:
  - Fault tolerant MPI
  - Application services such as checkpointing
  - Enhanced application manager
- After the research phase a flight prototype will be build
Where are we?

- Functional testbed system has been established
- Demonstrated that several fault tolerant techniques work effectively
- Over the next 18 months many more techniques will be investigated:
  - Fault tolerant MPI
  - Application services such as checkpointing
  - Enhanced application manager
- After the research phase a flight prototype will be build
Where do we go from here?
Credits

- University of Illinois - Ravi Iyer and Keith Whisnant
- General Dynamics - Chris Wink and John Pawasuskas
- W. W. Technology Group - Chris Walter and Brian LaValley
- Chalmers University - Neeraj Suri
- University of California Los Angeles – Dave Rennols
- JPL - Fannie Chen, Loring Craymer, Jeff Deifik, Al Fogel, Dan Katz, Al Silliman, Rafi Some, Sean Upchurch, Mike Turmon, Robert Granat, John Davidson, Robert Ferraro, John Beahan, John Thomas, Scott Packard, Yee Lee, Paul Springer, Roger Lee
Chameleon elements

Local Fault Handling uses SIFT layer based on Chameleon

R. Iyer, UIUC