Extended collaborative support services (ECSS)
Introduction

• Overview of the ECSS program
• ECSS case study – collaboration with Mao Ye
• Opportunities for collaboration beyond ECSS
And a little background

• I am not an economist or expert in finance. My work with Mao focused purely on improvements to the computing

• Therefore anything dumb I say in this talk is due to limitations of my knowledge of the field and does not reflect on Mao

• My broader role is co-lead of XSEDE’s ECSS program (together with PSC’s Phil Blood)
Extended Collaborative Support Services (ECSS)

- Collaboration between researchers and XSEDE
- Expertise is available in a wide range of areas:
  - Performance analysis and optimization
  - Software parallelization and scalability improvements
  - Gateway and web portal development
  - Specialized scientific software
  - Visualization
  - Workflows
- Can solicit Advanced Support at any time through the Allocations tab at the XSEDE User Portal
- Requires written request
- Learn more at [https://www.xsede.org/for-users/ecss](https://www.xsede.org/for-users/ecss)
ECSS areas (under the hood)

ECSS consists of five areas. As an end user, you won’t need to be aware of the areas and our staff will figure out where you fit best.

• Extended Support for Research Teams (ESRT): Traditional ECSS projects to improve software and make best use of XSEDE resources.
• Extended Support for Community Codes (ESCC): Similar to ESRT, but with emphasis on codes available for public use.
• Extended Support for Science Gateways (ESSGW): Development of web interfaces to XSEDE resources.
• Novel and Innovative Projects (NIP): Assistance to users from domains that are relatively new to XSEDE and high performance computing.
• Extended Support for Training, Education and Outreach (ESTEO): Technical support for use of advanced cyberinfrastructure.
ECSS justification

To request ECSS, need to answer five questions

1. What do you want to accomplish with the help of expert staff? Have you already done any work on this aspect of your software?
2. How would the success of this collaboration benefit your project?
3. Which member(s) of your team would collaborate with ECSS staff?
4. Have you had significant interaction on previous projects related to your current proposal or discussed your extended support needs with any XSEDE staff? If so, please indicate with whom.
5. Have you received XSEDE advanced support in the past? If so, please indicate the time period, and how the support you received then relates to the support you request now.
# ECSS projects

List of ECSS projects (current and completed) can be found at [https://www.xsede.org/web/site/for-users/ecss/ecss-projects](https://www.xsede.org/web/site/for-users/ecss/ecss-projects)

## Research Teams

<table>
<thead>
<tr>
<th>Project Name</th>
<th>PI</th>
<th>PI Institution</th>
<th>ECSS Consultant(s)</th>
<th>Allocation End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>An implicit, Chimera-based discontinuous Galerkin solver: development and application</td>
<td>Paul David Orkis</td>
<td>University of Cincinnati</td>
<td>Davide Del Vento, Shiquan Su</td>
<td>9/30/18</td>
</tr>
<tr>
<td>Statistical Analysis for Partially-Observed Markov Processes with Marked Point Process Obs, Y4</td>
<td>Yong Zeng</td>
<td>University of Missouri, Kansas City</td>
<td>Mitchell DeWayne Horton</td>
<td>6/30/18</td>
</tr>
<tr>
<td>Genome-Wide microRNAs and Single Gamete Based Genetic Profiling of Sweet Sorghum Varieties for Biofuel Production</td>
<td>Ahmad Naseer Aziz</td>
<td>Tennessee State University</td>
<td>Alex Ropelewski</td>
<td>5/16/18</td>
</tr>
<tr>
<td>DISCO: a Digital Instrument for Sound Synthesis and Composition, Y2</td>
<td>Seye Tipe</td>
<td>University of Illinois at Urbana-Champaign</td>
<td>Paul Rodriguez</td>
<td>6/13/18</td>
</tr>
<tr>
<td>Six Degrees of Francis Bacon, Y2</td>
<td>Christopher Norton Warren</td>
<td>Carnegie Mellon University</td>
<td>David Walling</td>
<td>6/30/18</td>
</tr>
<tr>
<td>Allocation Request on Bridges for Joint Analysis of Metagenomics and Metabolomics Data, Y4</td>
<td>Ping Ma</td>
<td>University of Georgia</td>
<td>Paul Rodriguez, Philip Blood</td>
<td>9/30/18</td>
</tr>
<tr>
<td>Analysis of human cortical electrophysiological data: database design for rapid analysis</td>
<td>Max Nuvoli</td>
<td>University of Pittsburgh</td>
<td>Ruben Lu Guerre</td>
<td>7/10/18</td>
</tr>
<tr>
<td>Assessment of Competition in the US Markets Based on Retail Scanner Data</td>
<td>Philip Garland Gayle</td>
<td>Kansas State University</td>
<td>Kwai Wong, Od Oobodraiah</td>
<td>9/4/18</td>
</tr>
<tr>
<td>Simulation for 2D Semiconductor with Parallel Uniform and Adaptive Multigrid Method for Multi-component Phase Field Crystal Models, Y2</td>
<td>Zhen Guan</td>
<td>University of California, Irvine</td>
<td>David Boak, Dmitry Pekurovsky, Sudhakar Pamidighantam</td>
<td>9/30/18</td>
</tr>
<tr>
<td>Turbulent Mixing in a Magnetic Field and Flow structure under Successive Axi-symmetric Straining, Y2</td>
<td>Pui-xuan Yeung</td>
<td>Georgia Institute of Technology</td>
<td>Lars Koesterke</td>
<td>9/30/18</td>
</tr>
<tr>
<td>Modeling Heliospheric Phenomena with M6-FLUKSS and Observational Boundary Conditions</td>
<td>Nikolai Pogorelov</td>
<td>University of Alabama, Huntsville</td>
<td>Laura Carrington</td>
<td>6/30/18</td>
</tr>
<tr>
<td>The &quot;Morelli Machine&quot;: A Proposal Testing a Critical, Algorithmic Approach to Art History</td>
<td>Christopher James Nygren</td>
<td>University of Pittsburgh</td>
<td>Alan Craig, Paul Rodriguez</td>
<td>10/9/18</td>
</tr>
<tr>
<td>New Frontiers of Direct Laser Acceleration in Megatesa Magnetic Fields</td>
<td>Alex Areffiev</td>
<td>University of California, San Diego</td>
<td>Amit Chourasia, Shiquan Su</td>
<td>2/6/19</td>
</tr>
</tbody>
</table>
ECSS project case study – Mao Ye

- Now that we’ve seen an overview of the ECSS program, we’ll make things more concrete and highlight our collaboration with Mao Ye.
- Takeaway – big success that led to 100x reduction in time to construct limit order books, followed by an additional (compounded) 50x speedup.
- Caveat – Past Performance Is No Guarantee of Future Results.
Limit order book

• Market data consists of 17 different types of messages, describing a variety of activities. For our purposes, we’re only concerned with a few of these
  – Add new order to buy/sell with (F) or without (A) market participant ID
  – Execute order with (C) or without (E) price message
  – Delete an outstanding order, either fully (D) or partially (X)
  – Update an order (U) – replace an old order with a new one

• A limit order is an order placed on an exchange to buy or sell a certain number of shares at a price equal or better than market price. The limit order book (LOB) is just a record of unexecuted orders

• Once we have the LOB, we can calculate bid-ask spread and market depths (number of outstanding orders at each price)
### Seven types of messages that we care about

<table>
<thead>
<tr>
<th>Type</th>
<th>Timestamp (nanoseconds)</th>
<th>Order Reference Number</th>
<th>Buy/Sell</th>
<th>Shares</th>
<th>Stock</th>
<th>Price</th>
<th>Original Order Reference Number</th>
<th>Market Participant ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>53435.759668667</td>
<td>335531633</td>
<td>S</td>
<td>300</td>
<td>EWA</td>
<td>19.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>40607.031257842</td>
<td>168914198</td>
<td>B</td>
<td>100</td>
<td>NOK</td>
<td>9.38</td>
<td></td>
<td>UBSS</td>
</tr>
<tr>
<td>U</td>
<td>53520.367102587</td>
<td>336529765</td>
<td>B</td>
<td>300</td>
<td></td>
<td>19.45</td>
<td>335531633</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>53676.740300677</td>
<td>336529765</td>
<td></td>
<td></td>
<td></td>
<td>19.45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>57603.003717685</td>
<td>625843333</td>
<td></td>
<td>100</td>
<td></td>
<td>32.25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>53676.638521222</td>
<td>336529765</td>
<td></td>
<td>100</td>
<td></td>
<td>32.25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>53676.740851701</td>
<td>336529765</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **A** Add order anonymously
- **F** Add order with Market participant ID
- **U** Update: replace old order with new order
- **E** Order Execution
- **C** Order Executed with Price Message
- **X** Partial cancellation
- **D** Order Deletion
Sample Limit Order Book

<table>
<thead>
<tr>
<th>Market</th>
<th>Symbol</th>
<th>Bid</th>
<th>Ask</th>
<th>Depth</th>
<th>Qtty</th>
<th>Spread</th>
</tr>
</thead>
<tbody>
<tr>
<td>025 - NASDAQ</td>
<td>MSFT</td>
<td>24.760</td>
<td>24.770</td>
<td>7/3</td>
<td>183/33</td>
<td>-0.010</td>
</tr>
<tr>
<td></td>
<td></td>
<td>24.750</td>
<td>24.780</td>
<td>5/4</td>
<td>117/85</td>
<td>-0.030</td>
</tr>
<tr>
<td></td>
<td></td>
<td>24.740</td>
<td>24.790</td>
<td>5/2</td>
<td>22/265</td>
<td>-0.050</td>
</tr>
<tr>
<td></td>
<td></td>
<td>24.730</td>
<td>24.800</td>
<td>4/2</td>
<td>22/7</td>
<td>0.070</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Id</th>
<th>Hour</th>
<th>Qtty</th>
<th>Bid</th>
<th>Ask</th>
<th>Qtty</th>
<th>Hour</th>
<th>Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>BTRD</td>
<td>14:00:15</td>
<td>5</td>
<td>24.760</td>
<td>24.770</td>
<td>15</td>
<td>14:00:22</td>
<td>BTRD</td>
</tr>
<tr>
<td>ARCA</td>
<td>14:00:17</td>
<td>69</td>
<td>24.760</td>
<td>24.770</td>
<td>10</td>
<td>14:00:24</td>
<td>AMEX</td>
</tr>
<tr>
<td>INCA</td>
<td>14:00:19</td>
<td>14</td>
<td>24.760</td>
<td>24.780</td>
<td>1</td>
<td>14:00:19</td>
<td>PERT</td>
</tr>
<tr>
<td>CINN</td>
<td>14:00:21</td>
<td>78</td>
<td>24.760</td>
<td>24.780</td>
<td>1</td>
<td>14:00:23</td>
<td>COWN</td>
</tr>
<tr>
<td>SBSH</td>
<td>14:00:34</td>
<td>10</td>
<td>24.760</td>
<td>24.780</td>
<td>78</td>
<td>14:00:35</td>
<td>BRUT</td>
</tr>
<tr>
<td>MWRI</td>
<td>14:00:41</td>
<td>5</td>
<td>24.760</td>
<td>24.780</td>
<td>5</td>
<td>14:00:37</td>
<td>CINN</td>
</tr>
<tr>
<td>SIZE</td>
<td>13:59:31</td>
<td>10</td>
<td>24.750</td>
<td>24.790</td>
<td>15</td>
<td>14:00:30</td>
<td>TDCM</td>
</tr>
<tr>
<td>MADF</td>
<td>14:00:02</td>
<td>2</td>
<td>24.750</td>
<td>24.800</td>
<td>1</td>
<td>14:00:17</td>
<td>MLCO</td>
</tr>
<tr>
<td>MWSE</td>
<td>14:00:21</td>
<td>10</td>
<td>24.750</td>
<td>24.800</td>
<td>6</td>
<td>14:00:53</td>
<td>SBSH</td>
</tr>
<tr>
<td>SCHB</td>
<td>14:00:28</td>
<td>94</td>
<td>24.750</td>
<td>24.810</td>
<td>10</td>
<td>13:59:34</td>
<td>TMBR</td>
</tr>
</tbody>
</table>
High frequency trading
Data processing pipeline

Three step data processing pipeline, with run time dominated by the limit order book construction. First two steps only done once for each day of market activity and results can be used for every stock traded that day.
LOB construction can be time consuming

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Wall time (s) original code</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWN</td>
<td>8,400</td>
</tr>
<tr>
<td>AMZN</td>
<td>55,200</td>
</tr>
<tr>
<td>AAPL</td>
<td>129,914</td>
</tr>
</tbody>
</table>

Timings obtained using all 16 cores on a single Gordon compute node (dual socket 2.6 GHz Intel Sandy Bridge). NASDAQ trading data from June 4, 2010.

Original shared memory version of code parallelized using Pthreads

Like any other task that we expect to do repeatedly, reducing time to solution will make the researchers more productive.
Optimizing the software

- I’ll go through the next few slides very quickly. They’re intended to give a little taste of the kind of work that we do.
- The main take away is that most cutting edge research requires a multi-disciplinary team. In this case, a collaboration between domain experts (Mao and his team) and high performance computing experts (XSEDE ECSS staff).
Code optimization (part I) – do things once

Profiling code indicated that a large fraction of the run time was spent converting string to floats or integers. This was not initial I/O, but rather the repeated conversion inside inner loops. Also expending considerable time in string comparisons

// Operation performed inside loops
seqcurrent = atof(settled[y][8].c_str());
seqoriginal = atof(settled[y][9].c_str());
if (settled[y][3].compare("B") == 0)

// Do once at start of program
for (int y=0; y < numRows; y++) {
    rss[y].fset5 = atof(settled[y][5].c_str());
    rss[y].fset8 = atof(settled[y][8].c_str());
    rss[y].fset9 = atof(settled[y][9].c_str());
    rss[y].iset4 = atoi(settled[y][4].c_str());
    rss[y].isB   = settled[y][3].compare("B");
    rss[y].isS   = settled[y][3].compare("S");
}

// Then use results repeatedly
seqcurrent = rss.fset8[y];
seqoriginal = rss.fset9[y];
if (rss.isB == 0)
In main parallel loop, all threads write output to file. To avoid conflicts, locks set so that only one thread writes at a time. Unfortunately, this forces serialization. Instead, store results to array and output after exiting loop.

```c
for (...) {
    rc = pthread_mutex_lock(&mutex);
    checkResults("pthread_mutex_lock()\n",rc);
    lob_msft.open(writeFile,iios::app);
    lob_msft << ...
    lob_msft.close();
    rc = pthread_mutex_unlock(&mutex);
    checkResults("pthread_mutex_unlock()\n",rc);
}
```

**Before**

**Critical region**

```c
for(...) {
    accum[i] << ...
}
```

**Parallel**

```c
for(...) {
    lob_msft << accum[i].str();
}
```

**After**
for (int t = 1; t<numThreads-1; t++) {
    comp[t].start = comp[t-1].end;
    comp[t].end = comp[t-1].end+tInc;
} for (int t = 0; t<numThreads; t++) {
    pthread_create(&threads[t],NULL,tFunc1,(void *)&comp[t]);
} for (int t = 0; t<numThreads; t++) {
    pthread_join(threads[t],NULL);
}

Main function still showing imperfect load balancing. Iterations of key loop are independent, but take different amounts of time to execute. Strip out pthreads code (dense, hard to read & maintain) and replace with OpenMP directive with dynamic scheduling

#pragma omp parallel for private(i) schedule(dynamic,10)
for (...) {
    // Expensive, but independent operations
}
The iterations within the key function are not only independent, but can often be terminated early by taking advantage of data ordering.

Before

```java
for (...) { // main loop
    for(int y = numRows-1; y >= 0; y--) {
        seqcurrent = rss[y].fset8;
        seqoriginal = rss[y].fset9;
        if ( (seqcurrent  >  macro_seqcurrent) &&
            (seqoriginal < macro_seqcurrent) ) {
            // additional code not shown
        }
    }
}
```

After

```java
for (...) { // main loop
    for(int y = numRows-1; y >= 0; y--) {
        seqcurrent = rss[y].fset8;
        seqoriginal = rss[y].fset9;
        if (seqcurrent < macro_seqcurrent) break; // No need to keep going!
        if ( (seqcurrent  >  macro_seqcurrent) &&
            (seqoriginal < macro_seqcurrent) ) {
            // additional code not shown
        }
    }
}
```
## Impact of optimized code

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Wall time (s) original code</th>
<th>Wall time (s) modified code</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWN</td>
<td>8,400</td>
<td>128</td>
<td>66x</td>
</tr>
<tr>
<td>AMZN</td>
<td>55,200</td>
<td>437</td>
<td>126x</td>
</tr>
<tr>
<td>AAPL</td>
<td>129,914</td>
<td>1,145</td>
<td>113x</td>
</tr>
</tbody>
</table>
# LOB construction for the full NASDAQ

## 5/6/10 (2960 symbols) “Flash crash”

<table>
<thead>
<tr>
<th>Step</th>
<th>Gordon (s)</th>
<th>Stampede (s)</th>
<th>Blacklight (s)</th>
<th>Memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parse</td>
<td>1,315</td>
<td>1,158</td>
<td>1,672</td>
<td>33</td>
</tr>
<tr>
<td>LOB input</td>
<td>9,705</td>
<td>8,149</td>
<td>15,051</td>
<td>62</td>
</tr>
<tr>
<td>LOB construct</td>
<td>31,938</td>
<td>40,495</td>
<td>66,855</td>
<td>59</td>
</tr>
</tbody>
</table>

## 8/1/12 (2754 symbols) Knight Capital computer glitch

<table>
<thead>
<tr>
<th>Step</th>
<th>Gordon (s)</th>
<th>Stampede (s)</th>
<th>Blacklight (s)</th>
<th>Memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parse</td>
<td>511</td>
<td>451</td>
<td>660</td>
<td>29</td>
</tr>
<tr>
<td>LOB input</td>
<td>2,865</td>
<td>2,536</td>
<td>4,880</td>
<td>35</td>
</tr>
<tr>
<td>LOB construct</td>
<td>7,885</td>
<td>8,045</td>
<td>18,921</td>
<td>44</td>
</tr>
</tbody>
</table>

## 8/7/12 (2750 symbols) typical trading day

<table>
<thead>
<tr>
<th>Step</th>
<th>Gordon (s)</th>
<th>Stampede (s)</th>
<th>Blacklight (s)</th>
<th>Memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parse</td>
<td>361</td>
<td>313</td>
<td>461</td>
<td>27</td>
</tr>
<tr>
<td>LOB input</td>
<td>2,017</td>
<td>1,781</td>
<td>3,289</td>
<td>44</td>
</tr>
<tr>
<td>LOB construct</td>
<td>6,005</td>
<td>5,774</td>
<td>14,965</td>
<td>29</td>
</tr>
</tbody>
</table>
Going beyond ECSS

• ECSS projects are of limited duration ...
  – Initially one year
  – Can be ended early if goals are met
  – Often renewed

• ... and level of effort by the ECSS consultant
  – Typically 25% of a consultant’s time
  – Sometimes multiple consultants

• ECSS can seed the initial work that can lead to externally funded collaborations
Going beyond ECSS

• Our initial ECSS project (Sinkovits, O’Neal, Ye) was quite successful, resulting in 100x reduction in time to solution
• We recognized there was more to be done, but goals were too ambitious for an ECSS project
  – Required rethinking algorithms rather than just tuning existing algorithms
  – Making optimal use of local (flash) file systems
• Work supported by NSF EAGER grant to Ye and Sinkovits
Goal of NSF funded work (my component)

• Extend the LOB construction to much more challenging problems (e.g. combined analysis of all stocks, across multiple markets, on very busy days of trading)
  – Requires further reduction in time to solution
  – New algorithms needed to reduce memory footprint so that standard, as opposed to specialized large memory, hardware could be used
  – Data management strategies for working with large input data and large file counts
1. Split message data

Message Data

x01

x02

x15

x16

LOB Construction workflow

Blue = parallel file system  Orange = Gordon SSD
For clarity only 16 processes are shown. Can be extended to all cores on a shared memory compute node.
1. Split message data

- **Message Data**
  - x01
    - AAPL_01
    - GOOG_01
    - NVDA_01
  - x02
    - AAPL_02
    - GOOG_02
    - NVDA_02
  - ...
  - x15
    - AAPL_15
    - GOOG_15
    - NVDA_15
  - x16
    - AAPL_16
    - GOOG_16
    - NVDA_16

2. Split into stock-specific results (parallel)

LOB Construction workflow:

- Blue = parallel file system
- Orange = Gordon SSD
For clarity only three stocks are shown. Can be extended to arbitrary number of stocks.
1. Split message data

2. Split into stock-specific results (parallel)

3. Combine partial files (parallel)

LOB Construction workflow

Blue = parallel file system
Orange = Gordon SSD
1. Split message data

2. Split into stock-specific results (parallel)

3. Combine partial files (parallel)

4. Construct LOBs (parallel)

Blue = parallel file system
Orange = Gordon SSD
1. Split message data

2. Split into stock-specific results (parallel)

3. Combine partial files (parallel)

4. Construct LOBs (parallel)

Blue = parallel file system  Orange = Gordon SSD

Message Data

AAPL_01  GOOG_01  NVDA_01

AAPL_02  GOOG_02  NVDA_02

AAPL_15  GOOG_15  NVDA_15

AAPL_16  GOOG_16  NVDA_16

AAPL

AAPL_LOB

GOOG

GOOG_LOB

NVDA

NVDA_LOB
Impact of second round of optimizations

New algorithms and approach to parallelization resulted in 50x speedup over previously optimized versions of the codes

<table>
<thead>
<tr>
<th>trading date</th>
<th>tickers</th>
<th>time original (s)</th>
<th>time new (s)</th>
<th>memory original (GB)</th>
<th>memory new (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/07/12</td>
<td>2750</td>
<td>6005</td>
<td>320</td>
<td>29</td>
<td>0.90</td>
</tr>
<tr>
<td>08/01/12</td>
<td>2885</td>
<td>7885</td>
<td>349</td>
<td>44</td>
<td>1.16</td>
</tr>
<tr>
<td>05/06/10</td>
<td>2960</td>
<td>31983</td>
<td>663</td>
<td>59</td>
<td>1.71</td>
</tr>
</tbody>
</table>
Tackling even more challenging problems

Just when we thought that we’re done, a more challenging problem comes along – let’s analyze the combined NYSE, BATS and NASDAQ on heaviest days of trading (to date)

<table>
<thead>
<tr>
<th>Trading date</th>
<th>tickers</th>
<th>data (GB)</th>
<th>results (GB)</th>
<th>split (mm:ss)</th>
<th>LOB (mm:ss)</th>
<th>tar (mm:ss)</th>
<th>total (mm:ss)</th>
<th>memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/07/12</td>
<td>2750</td>
<td>4.9</td>
<td>14</td>
<td>2:15</td>
<td>2:20</td>
<td>0:45</td>
<td>5:20</td>
<td>0.90</td>
</tr>
<tr>
<td>08/01/12</td>
<td>2885</td>
<td>6.1</td>
<td>17</td>
<td>2:45</td>
<td>2:10</td>
<td>0:54</td>
<td>5:49</td>
<td>1.16</td>
</tr>
<tr>
<td>05/06/10</td>
<td>2960</td>
<td>14</td>
<td>37</td>
<td>7:19</td>
<td>7:25</td>
<td>2:55</td>
<td>17:39</td>
<td>1.71</td>
</tr>
<tr>
<td>05/07/10</td>
<td>7892</td>
<td>81</td>
<td>219</td>
<td>60:58</td>
<td>38:32</td>
<td>21:36</td>
<td>121:02</td>
<td>2.87</td>
</tr>
<tr>
<td>08/10/11</td>
<td>7891</td>
<td>100</td>
<td>270</td>
<td>58:58</td>
<td>36:26</td>
<td>32:41</td>
<td>119:05</td>
<td>4.02</td>
</tr>
</tbody>
</table>
Tackling even more challenging problems

Just when we thought that we’re done, a more challenging problem comes along – let’s analyze the combined NYSE, BATS and NASDAQ on heaviest days of trading (to date)

<table>
<thead>
<tr>
<th>Trading date</th>
<th>tickers</th>
<th>data (GB)</th>
<th>results (GB)</th>
<th>split (mm:ss)</th>
<th>LOB (mm:ss)</th>
<th>tar (mm:ss)</th>
<th>total (mm:ss)</th>
<th>memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/07/12</td>
<td>2750</td>
<td>4.9</td>
<td>14</td>
<td>2:15</td>
<td>2:20</td>
<td>0:45</td>
<td>5:20</td>
<td>0.90</td>
</tr>
<tr>
<td>08/01/12</td>
<td>2885</td>
<td>6.1</td>
<td>17</td>
<td>2:45</td>
<td>2:10</td>
<td>0:54</td>
<td>5:49</td>
<td>1.16</td>
</tr>
<tr>
<td>05/06/10</td>
<td>2960</td>
<td>14</td>
<td>37</td>
<td>7:19</td>
<td>7:25</td>
<td>2:55</td>
<td>17:39</td>
<td>1.71</td>
</tr>
<tr>
<td>05/07/10</td>
<td>7892</td>
<td>81</td>
<td>219</td>
<td>60:58</td>
<td>38:32</td>
<td>21:36</td>
<td>121:02</td>
<td>2.87</td>
</tr>
<tr>
<td>08/10/11</td>
<td>7891</td>
<td>100</td>
<td>270</td>
<td>58:58</td>
<td>36:26</td>
<td>32:41</td>
<td>119:05</td>
<td>4.02</td>
</tr>
</tbody>
</table>

This would have taken 2400-7200 hours using the original version of the code on a large memory (> 64 GB node)
Wrap-up

- Hopefully you’ve gained an appreciation for how you can benefit from XSEDE’s ECSS program
- We look forward to working with you
- Questions?