SAND2011-5207C

Pushing back the point of diminishing returns for parallel performance

Workshop on Exascale Programming Challenges

July 27 – 29, 2011

USC/ISI

Marina del Rey, CA

Curtis Janssen, Sandia National Laboratories

cljanss@sandia.gov

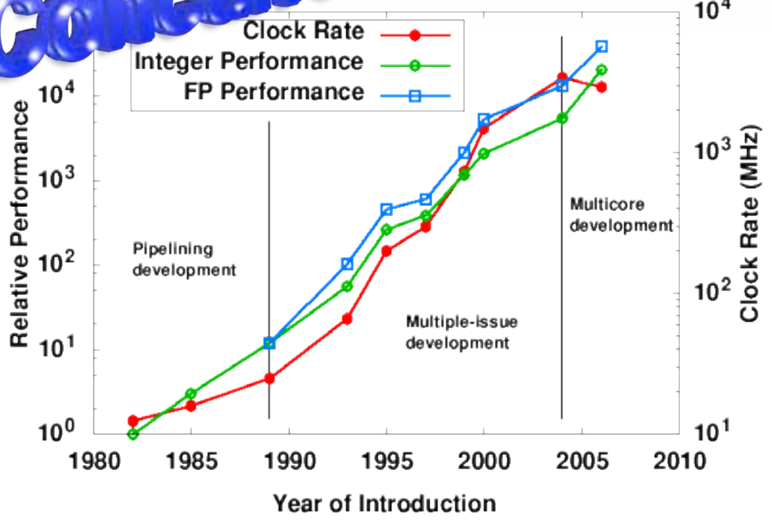
Unlimited Release

SAND 2011-5207C

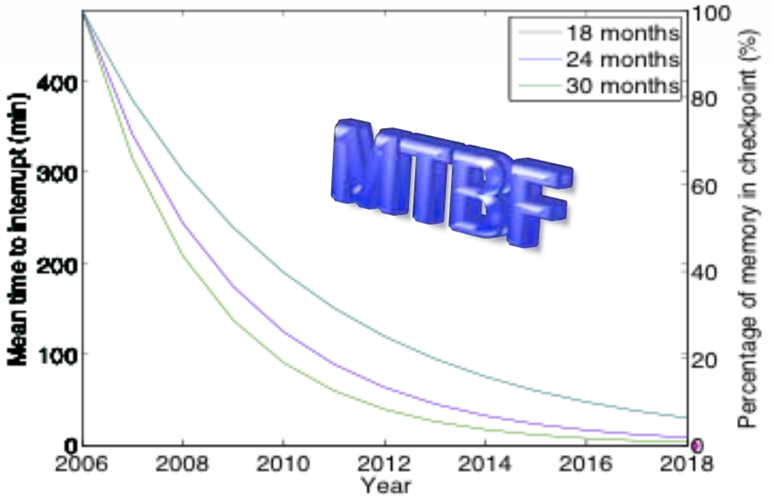
Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Challenges impacting exascale application performance

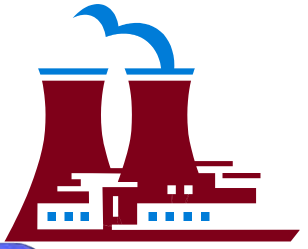
Concurrency



MTBF

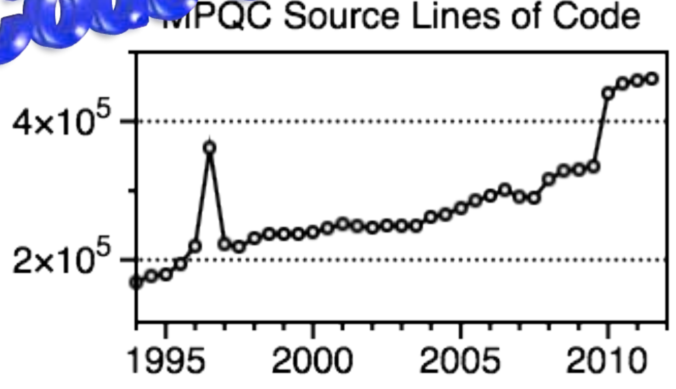


Schroeder and Gibson, Journal of Physics: Conference Series, **78** (2007) 012022, SciDAC 2007 Proceedings.



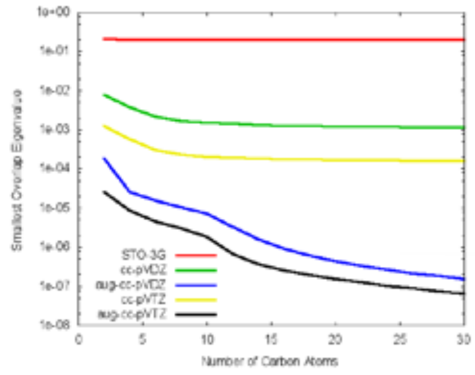
Power

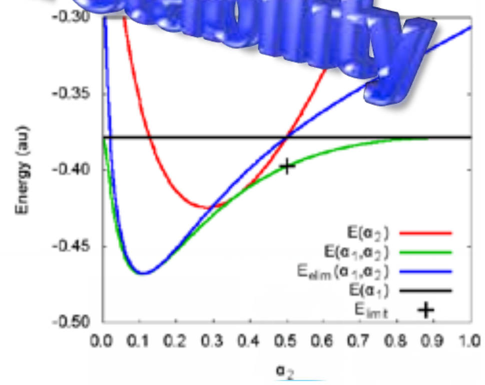
Code Complexity



Generated using David A. Wheeler's 'SLOCCount'

Numerical Stability





Example application: Hartree-Fock theory

- Approximate solution to Schrödinger's equation

$$H = \frac{1}{2}\sum_i^n \nabla_i^2 - \sum_i^n \sum_a^N \frac{q_a}{r_{ia}} + \sum_{i<j}^n \frac{1}{r_{ij}} + \sum_{a<b}^N \frac{q_a q_b}{r_{ab}}$$

- Electron interact with average field of other electrons, giving rise to a generalized eigenvalue problem
- Major steps (assuming spin restricted closed shell):
  - Integral computation:

$$S_{pq} = \int \chi_p(\mathbf{r})\chi_q(\mathbf{r})\,d\mathbf{r} \qquad H_{pq} = \int \chi_p(\mathbf{r})\left(\nabla^2 - \sum_a^N \frac{q_a}{r_A}\right)\chi_q(\mathbf{r})\,d\mathbf{r}$$

$$G_{pqrs} = \int \chi_p(\mathbf{r}_1)\chi_q(\mathbf{r}_1)\frac{1}{r_{12}}\chi_r(\mathbf{r}_2)\chi_s(\mathbf{r}_2)\,d\mathbf{r}_1\,d\mathbf{r}_2$$

  - Fock matrix formation:

$$F_{pq} = H_{pq} + P_{rs}\left(v_{pqrs} + \frac{1}{2}v_{prqs}\right)$$

  - Diagonalization:

$$\mathbf{FC} = \mathbf{SCe} \qquad \mathbf{CSC}^T = \mathbf{1}$$

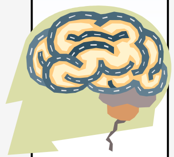
  - Density computation:

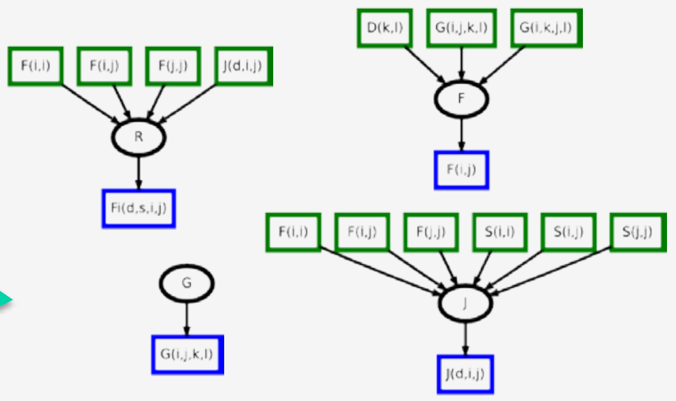
$$P_{pq} = 2\sum_a^{N/2} C_{pa}C_{qa}$$

Unteasing concurrency from applications

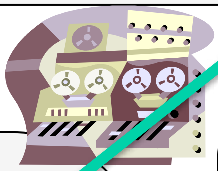
```
Form the atomic orbital Fock, F, and overlap, S
Synchronize so that F is complete on all nodes
Begin iterative eigensolver
  For each set of independent shell pairs
    Compute the rotation matrix
    Synchronize so rotation matrix is complete
    Rotate F and S
    Synchronize so that F and S are complete
  End loop over independent shell pairs
End eigensolver iterations
```

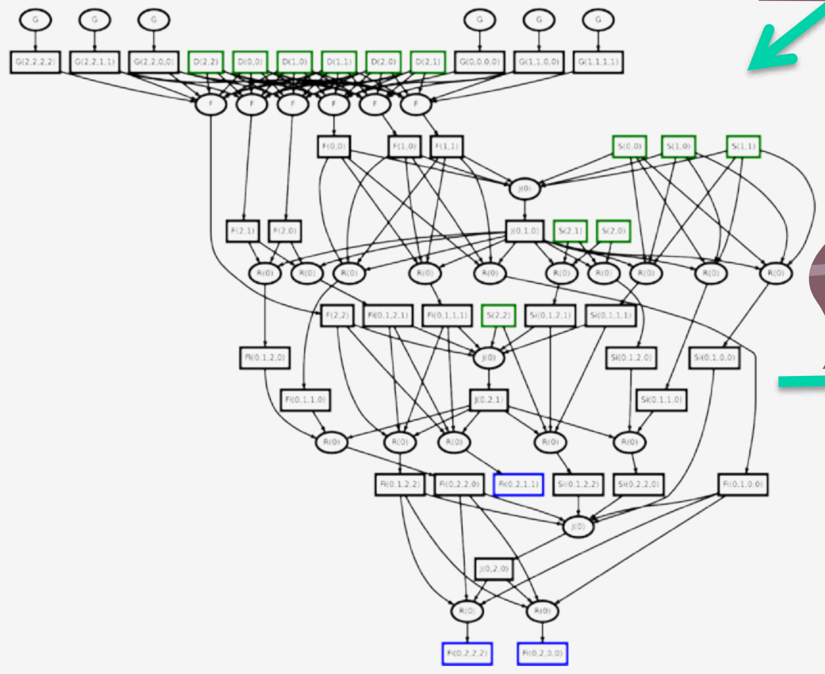
Traditional imperative formulation



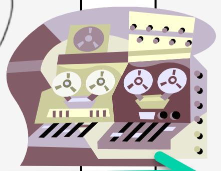


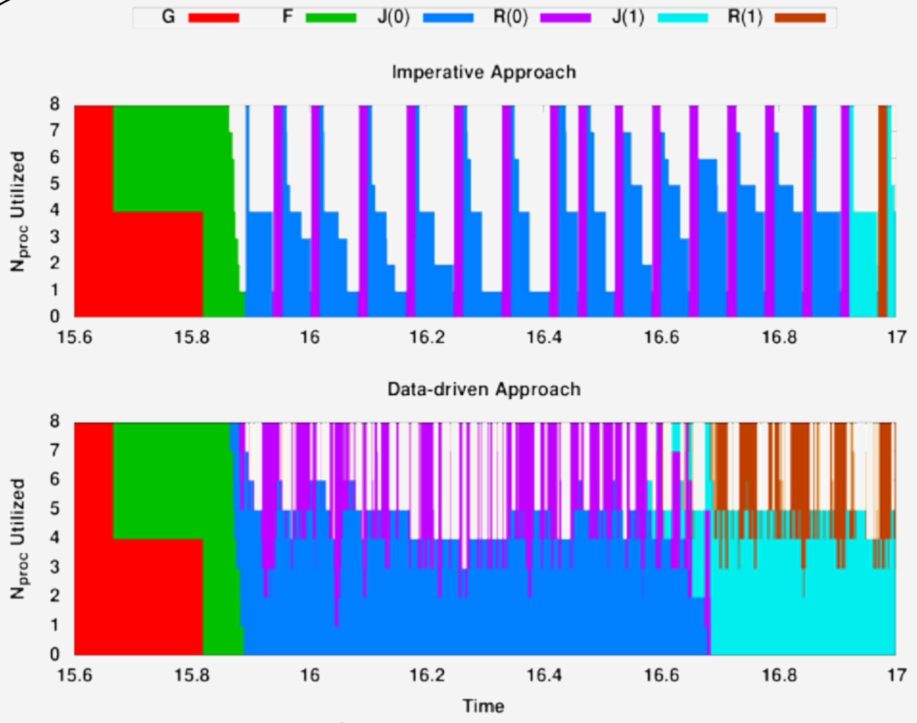
Elementary operations





Dataflow graph

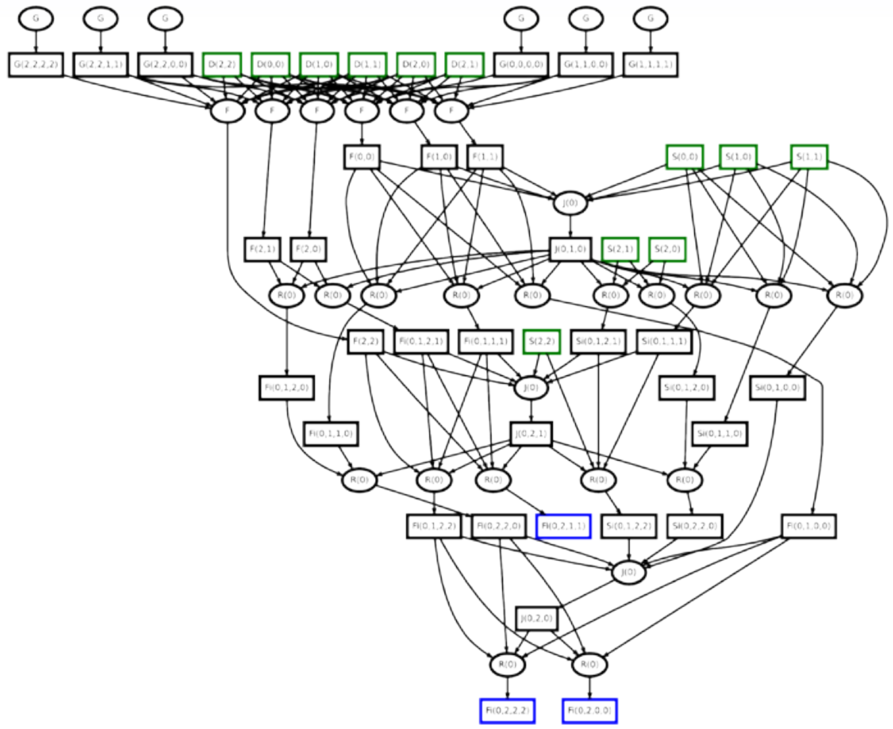




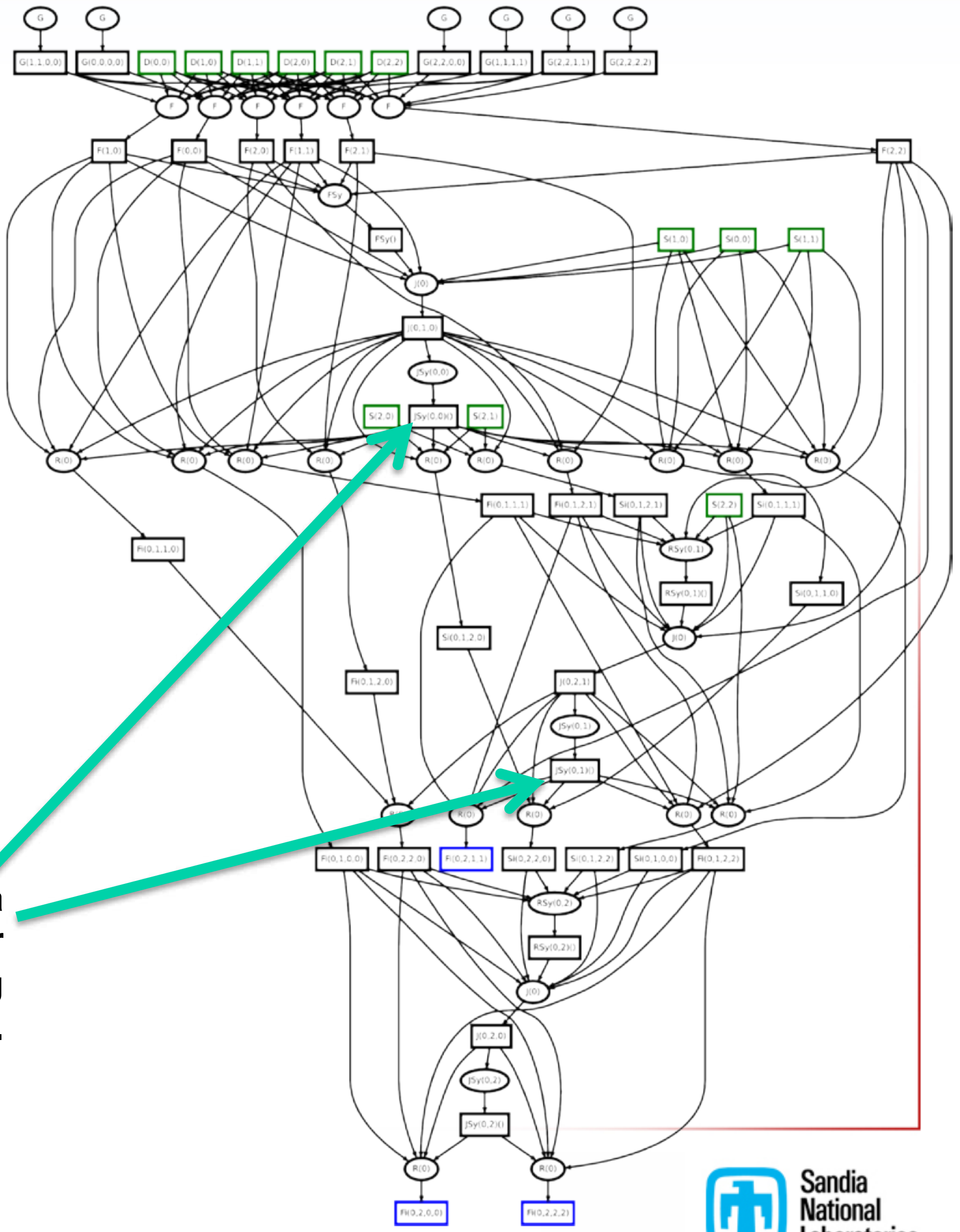
Simulated timings

Comparison of data dependencies with and without synchronization

Without synchronization:



With synchronization:

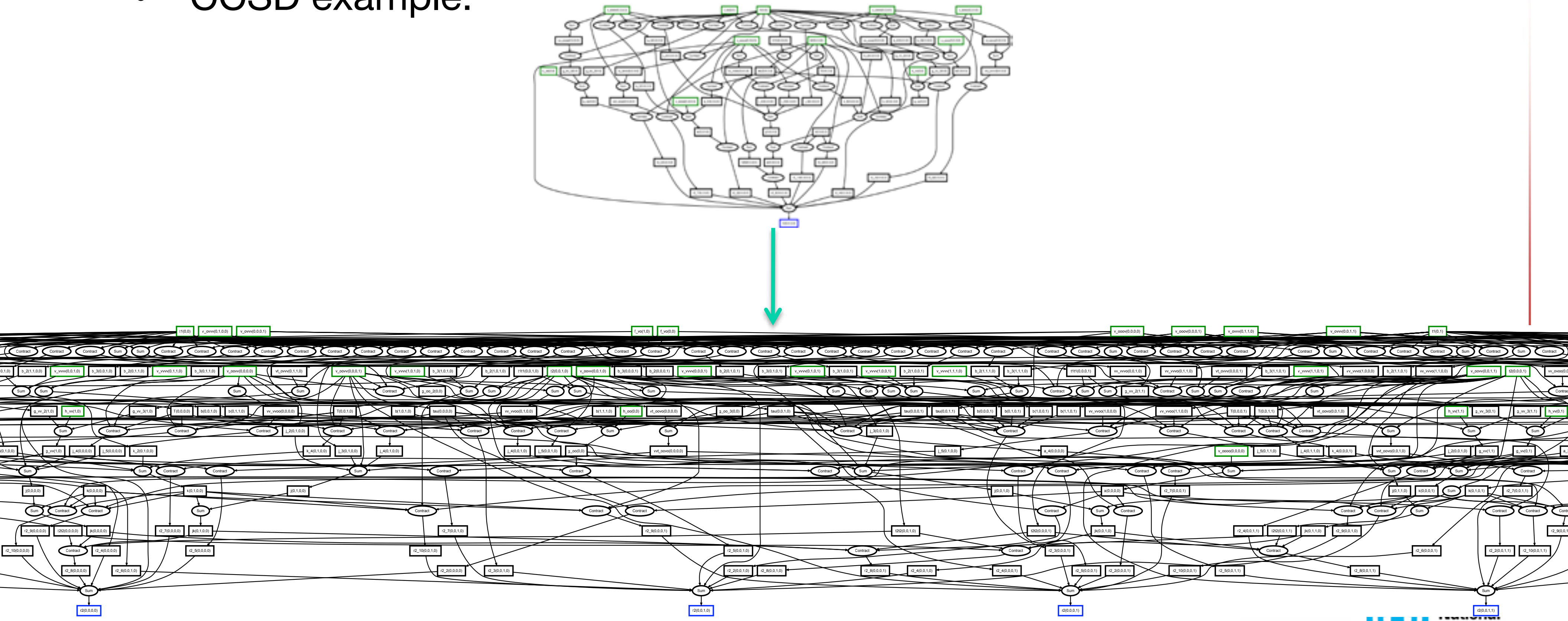


Synchronization increases the number of data dependencies. Thus, the overall potential for parallelization is reduced by synchronizing operations such as barriers and collectives.

Sandia National Laboratories

Hierarchical decomposition needed for locality and scalabilty

- Hierarchical in terms of operations
  - Eigenvectors constructed from Fock matrix constructed from integrals
- Hierarchical in terms of data
  - Large blocks containing small blocks, etc.
  - Map data hierarchy to memory hierarchy
  - CCSD example:



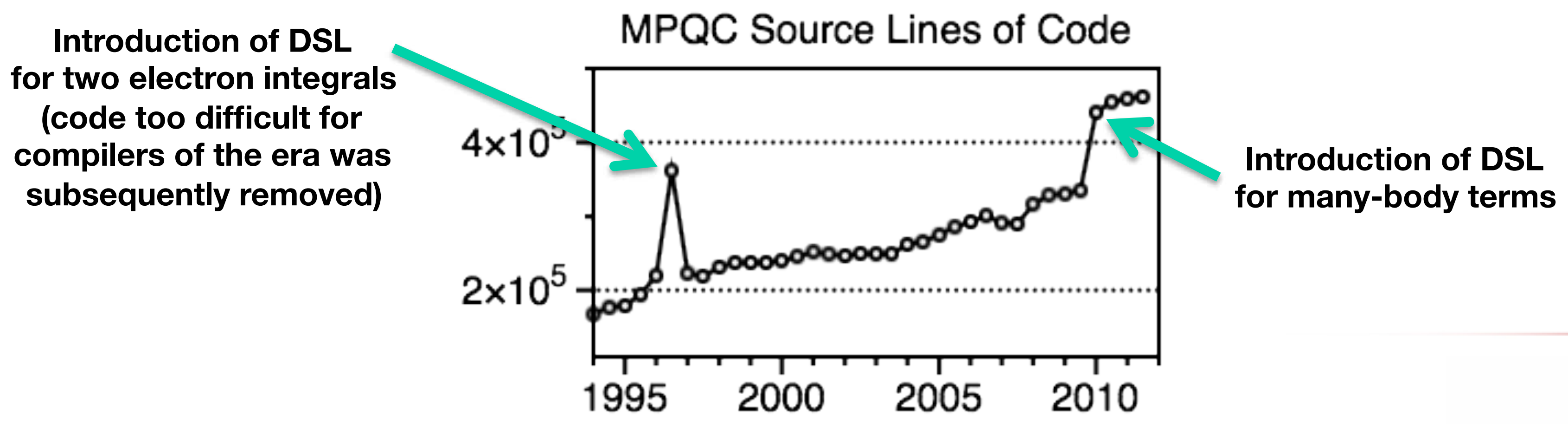
Be careful for what you ask …

Am I asking for a monolithic runtime system?

- No – this is the problem with MPI. Need a lightweight, portable, and low-level interface for fast messaging. Includes active messages and fault notification primitives.
- Varying levels of sophistication can be built upon this low-lying interface.

Am I asking for new languages?

- Yes and no – general purpose languages spoken and developed by a wide community will always play a role. Libraries, DSLs (to generate the underlying code), and embedded DSLs (to supplement the underlying language) will be essential to hide machine complexity.

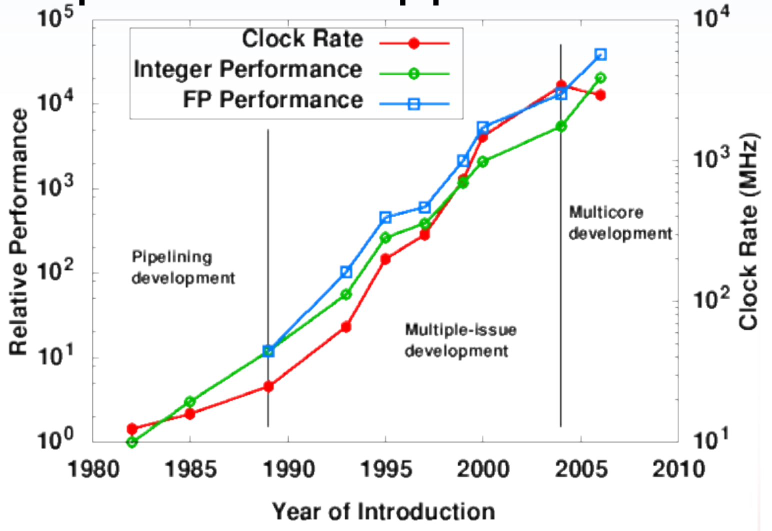


Supplemental Slides

Motivation: complexity of parallel machines is accelerating, but tools to manage this are not

- Several complexity issues affect apps:
  - Extreme parallelism
  - More computation power enables more complex/higher fidelity simulations
    - More complex software
    - Numerical issues
  - Dropping mean time between failure
  - Energy enters optimization objective function

Improvements to chip performance over the

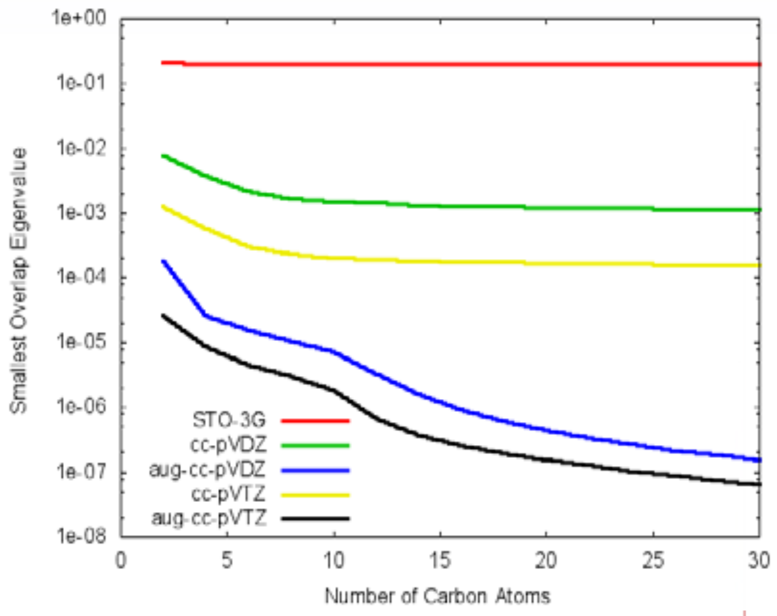


- Human effort does not scale easily to such a complex environment
  - Can another approach to programming solve some of these problems?
- Outline of current work:
  - Hartree-Fock theory selected due to its expense and scaling issues
    - Basis for many other electronic structure methods
  - Examine traditional implementation of Hartree-Fock theory
  - Show preliminary results of applying an alternative programming approach to Hartree-Fock and compare this to traditional implementations.

Sandia National Laboratories

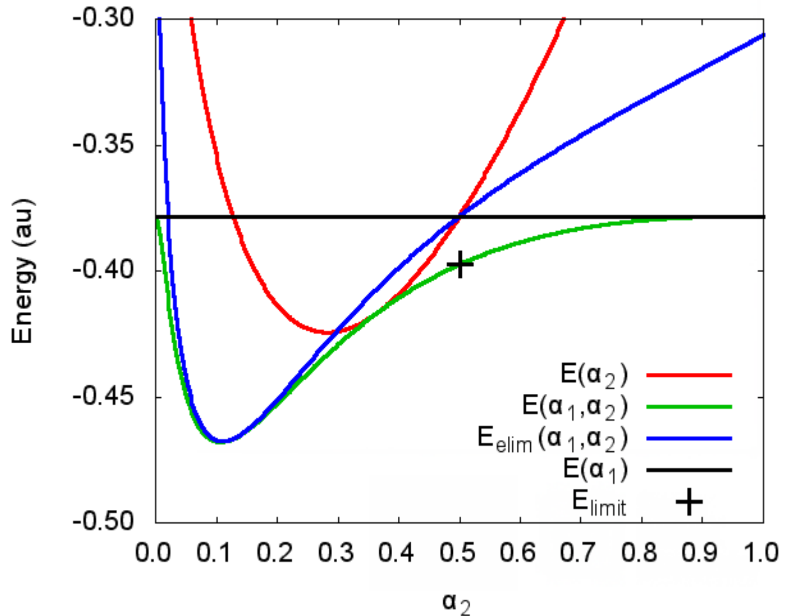
Illustration of numerical issues using Hartree-Fock theory as an example

Large systems are ill-conditioned: smallest overlap eigenvalue for linear alkane rapidly decrease as system grows for diffuse basis sets

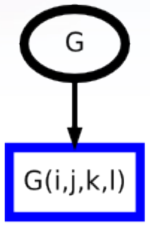


Eliminating near linear dependencies can change energies—even in the limit of an exact linear dep.

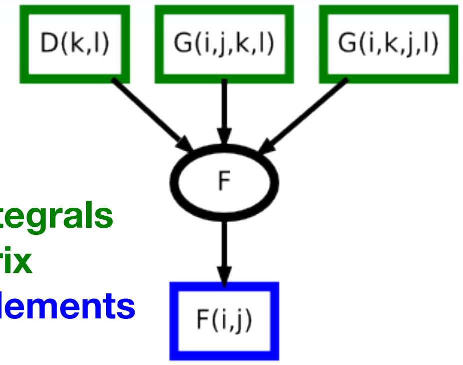
Errors due to keeping the nearly linear dep. functions grow like s_1^{-3}, and we need the difference between large numbers:



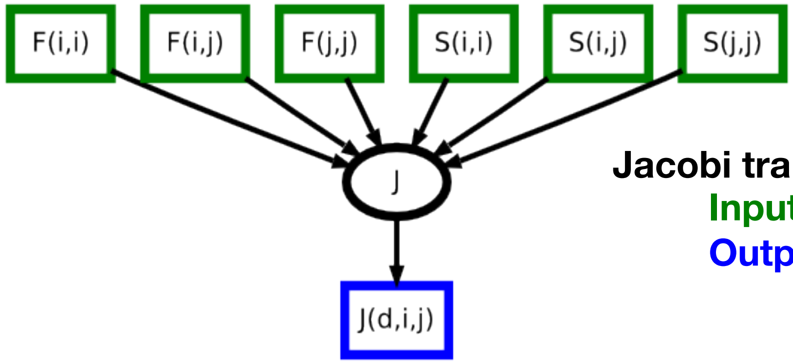
Elementary operations for Hartree-Fock in terms of data dependencies



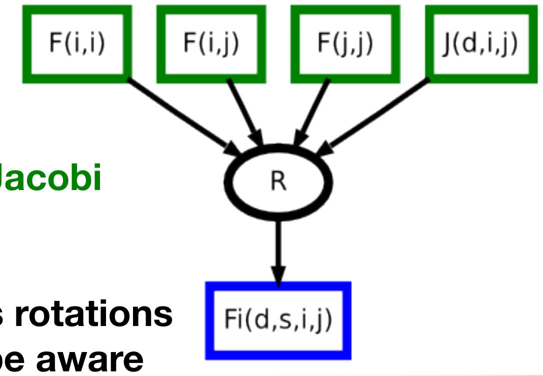
Two electron integrals formation, G:
Output: (ij|kl) for a shell quartet



Fock matrix formation, F:
Input: Two electron integrals and density matrix
Output: Fock matrix elements for a shell pair



Jacobi transform formation, J:
Input: Fock and overlap matrix elements
Output: Rotation matrix diagonalizing the sub-block

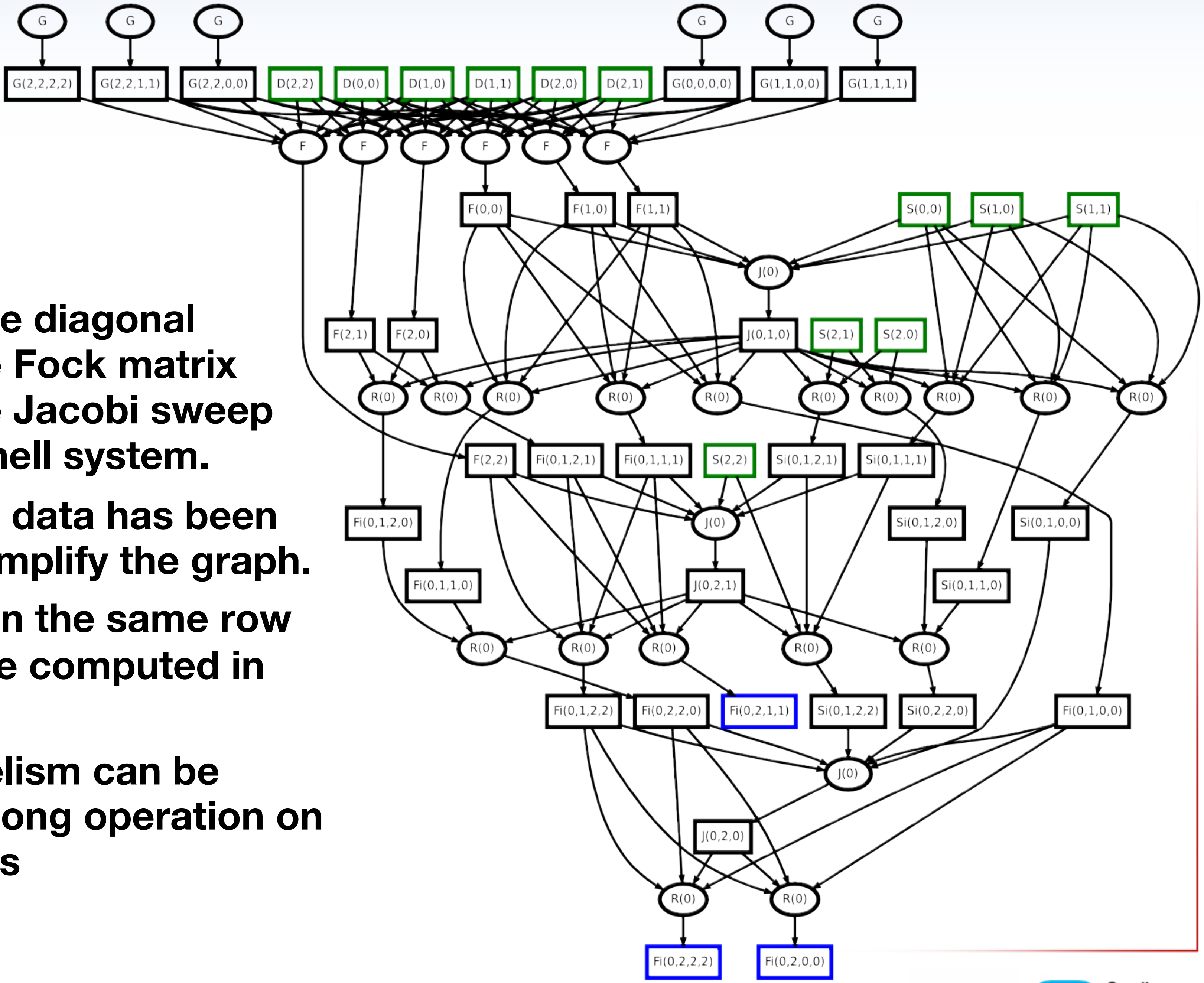


Matrix transformation, R:
Input: Fock or overlap matrix elements and Jacobi transform
Output: Transformed matrix elements
Note: output has a sequence number that ensures rotations are done in the correct order. Both J and R must be aware of sequence number

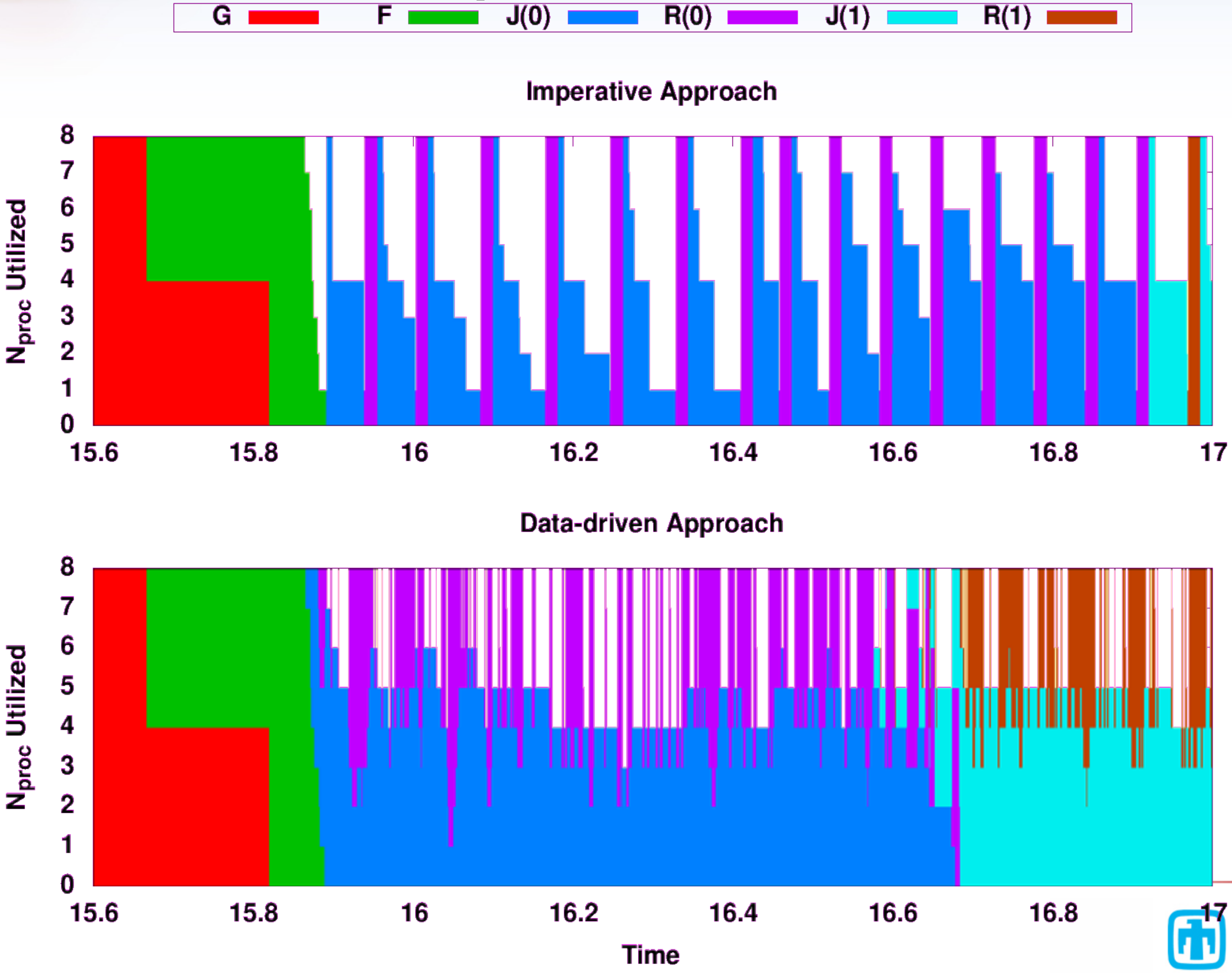
Sandia National Laboratories

Hartree-Fock data dependencies

- Computes the diagonal blocks of the Fock matrix after a single Jacobi sweep for a three shell system.
- Certain input data has been omitted to simplify the graph.
- Operations on the same row (ovals) can be computed in parallel
- Some parallelism can be exploited among operation on different rows



Simulated timings for 16 shells on 8 processors



Sandia National Laboratories