SAND2009-2561P

Preparing for exascale computing: Software technologies for more efficient utilization of extreme-scale machines

Pacific Northwest National Laboratory

2009-04-28

Curtis Janssen

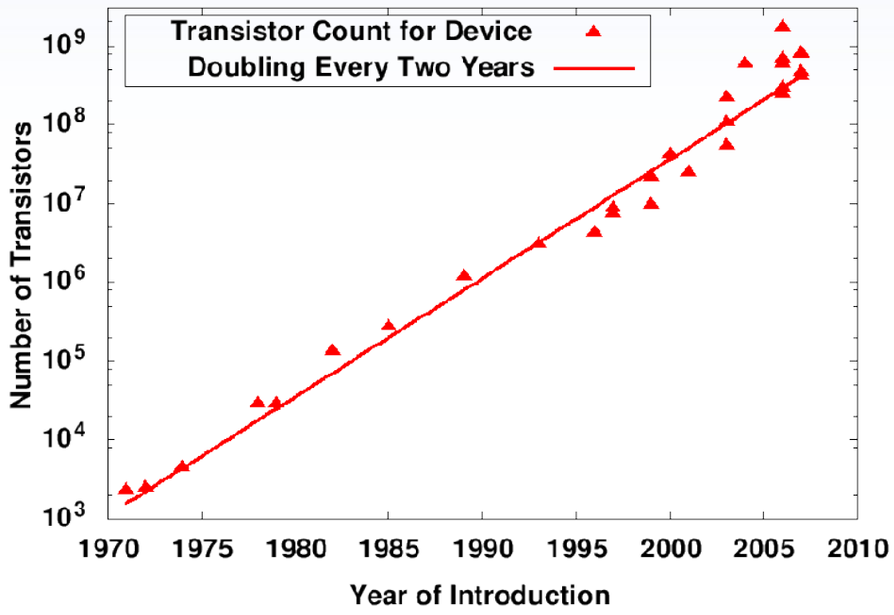
Sandia National Laboratories

Livermore, CA

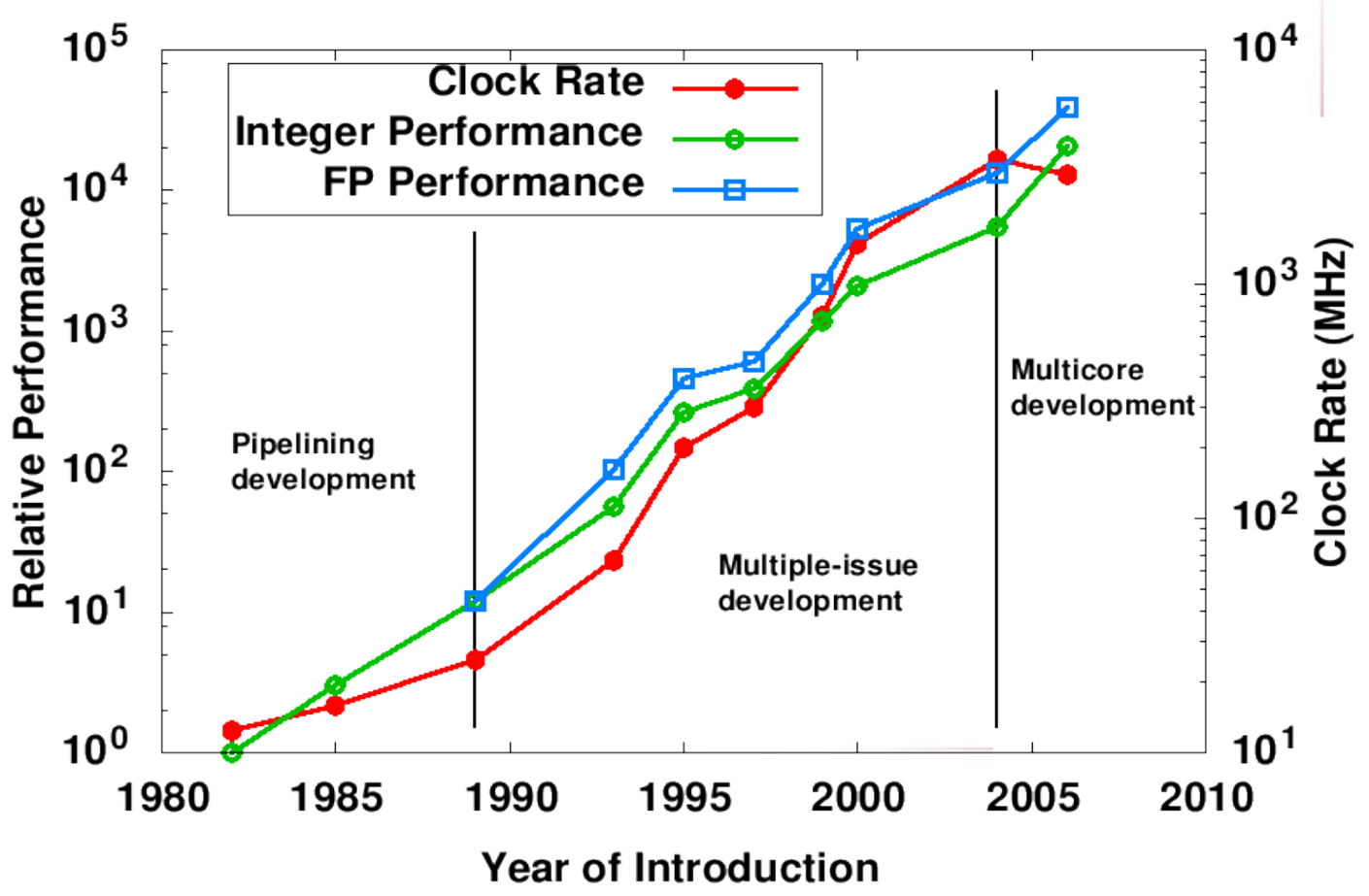
Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

The nature of computing hardware is changing

- Moore's Law continues to apply: the transistor count is doubling every two years

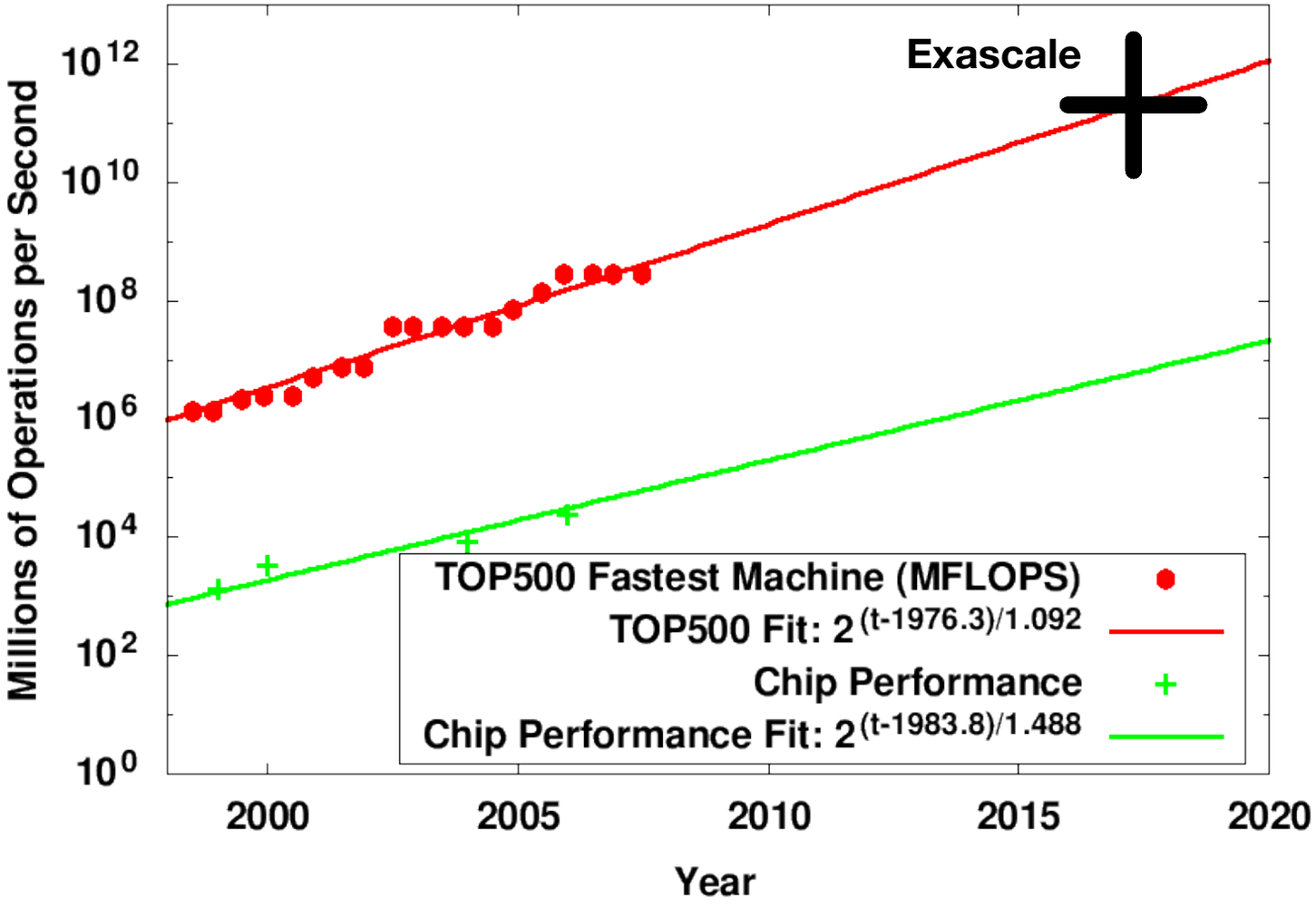


- Many have assumed a corollary: that single processor performance would improve due to improvements in clock rates and instruction level parallelism.
- *This is no longer true.*



The performance of the world's fastest machines doubles about every year

- Many problem domains will benefit from extreme levels of computing. Some examples are:
  - Materials: Understanding and rational design of nanomaterials with desired chemical optical, mechanical, and electronic properties
  - Climate: Multi-physics, high-fidelity (cloud resolving) earth model
  - Biology: Modeling coupled genomic, proteomic, metabolomic networks
- Target is to have exascale computing available before 2018



  - There are many obstacles to continuing a straight line trajectory, but work is underway to overcome these obstacles.
  - In any case, we can expect dramatic gains in the amount of available computing power

Sandia National Laboratories

Challenges for designers of high performance applications will be particularly acute

- Applications must expose much more parallelism
  - They must utilize a 3 order of magnitude increase in computing power
    - Little if any of this gain will be due to faster cores—it will be through having more cores
  - Computing power gains will continue to outstrip latency and bandwidth gains
    - Multiple hardware thread contexts are being implemented to hide memory latency.
  - Result is that 4 to 5 additional orders of magnitude of parallelism will have to be exposed.
- Can we assume the machine is homogeneous?
  - Some cores might be throttled if they are running too hot, etc.
- Can we assume the machine is reliable?
  - More components reduces MTBF
  - Smaller feature sizes make chips more susceptible to failure
  - Driving down power consumption increases failure rate
  - MTBF is approaching the time to checkpoint a machine
- There are many projects that address elements of these problems, DARPA HPCS, for example, but, ultimately, we will be forced to change the way we use large scale machines, and we do not yet know how

This talk will give an overview of work in three areas that could help manage complex software and hardware environments

- The Common Component Architecture (CCA) and its use in quantum chemistry applications

- Exposing more parallelism in quantum chemistry applications: Moving beyond the MPI and hybrid MPI/Multithreaded programming model

- SST/macro: a macroscale discrete event simulator for predicting application performance on large-scale parallel machines

Sandia National Laboratories

The Common Component Architecture (CCA) and its use in quantum chemistry applications

CCA/Chemistry team members and collaborators:

Pacific Northwest National Laboratory
Operated by Battelle for the U.S. Department of Energy

NWChem high performance computational chemistry software

Yuri Alexeev
Manojkumar Krishnan
Elizabeth Jurrus
Carl Fahlstrom
Jarek Nieplocha

Sandia National Laboratories

MPQC

Joe Kenny
Curtis Janssen
Ida Nielsen
Rob Armstrong
Ben Allan

Gary Kumfert

Argonne National Laboratory · University of Chicago

Steve Benson
Jason Sarich
Lois Curfman McInnes

ornl Oak Ridge National Laboratory

David Bernholdt
Ricky Kendall

AMES LABORATORY

GAMESS

Theresa Windus
Sasha Sosonkina
Fang Peng
Meng-Shiou Wu
Alexander Geanko
Mark Gordon

VirginiaTech Invent the Future

Edward Valeev

Sandia National Laboratories

Scientists as programmers have issues

- Not known for sound software engineering practices
  - Simplest approach often used, even if inefficient
    - or some use the most efficient approach possible, even if unmaintainable
  - Coding practices are often out-dated
    - Poor style, little documentation, incomplete testing
    - Difficult to convert to modern programming techniques
      - Learning curve, poor training, and legacy code are issues.
  - Common programming tools that helped make large-scale software efforts such as GNU/Linux successful are not uniformly utilized:
    - minimal use of software configuration management, build systems
- Diverse community of government/academic, noncommercial quantum chemistry (QC) packages
  - Many packages have fallen into the monolithic code trap
    - Limits ability to leverage existing software: limits both the quality and capabilities of what can be done.
  - Interaction between QC and other fields even more difficult
- All this is in addition to the complexity to utilize extreme scale machines

How can programmers become more productive?

- Object-oriented methodologies? There are issues:
  - Cannot leave out legacy codes
  - Even with modern codes, design patterns may be similar but implementation/language is not. Code bases incompatible at a low level.
  - Not a complete solution
- Characteristics of a solution for improving code sharing:
  - Must support multiple languages
  - Must allow for mostly independent programming in packages using it
  - Community must agree on a few well-defined or common elements in the design that place minimal constraints on each software package
  - The Common Component Architecture is designed to satisfy these requirements.

Illustration of complications in QC: Chromium hydroxides

- Accurate thermochemical knowledge needed to understand contamination in industrial settings and pollution
- Experimental data is missing or inconsistent
- Six reactions used to obtain heat of formation for $Cr(OH)_n$, n = 2–6 and $CrO(OH)_4$.

$$Cr(OH)_2 + Cr \rightarrow 2\ CrOH$$

$$Cr(OH)_3 + 2\ Cr \rightarrow 3\ CrOH$$

$$Cr(OH)_4 + 3\ Cr \rightarrow 4\ CrOH$$

$$Cr(OH)_5 + CrOH \rightarrow CrOH$$

$$Cr(OH)_6 \rightarrow CrO_3 + 3\ H_2O$$

$$CrO(OH)_4 \rightarrow CrO_3 + 2\ H_2O$$

I'll come back to these two examples later.

Sandia National Laboratories

High accuracy is hard

- Thousands of hours of CPU time and four quantum chemistry code suites later ...

	$Cr(OH)_2$ Rxn 1	$Cr(OH)_3$ Rxn 2	$Cr(OH)_4$ Rxn 3	$Cr(OH)_5$ Rxn 4	$Cr(OH)_6$ Rxn 5	$CrO(OH)_4$ Rxn 6
ΔE_{rxn}[HF]	7.71	−4.60	−50.63	132.49	−29.21	24.02
δ[CCSD]	+7.17	+18.22	+40.47	−44.51	+15.57	+1.45
δ[CCSD(T)]	+0.36	+4.62	+13.06	−11.03	+9.62	+0.78
δ[basis]	−0.30	−1.15	−2.11	−3.49	−3.91	−2.63
δ[core]	−0.37	+0.76	+1.82	[b]	+2.08	+0.90
δ[rel]	−2.53	−2.87	−3.19	+3.89	+2.06	+3.42
δ[ZPVE]	−1.06	−2.64	−4.12	−5.53	−8.07	−4.99
$\Delta H^{\circ}_{\mathrm{rxn},0}$	10.98	12.34	−4.70	71.82	−11.86	22.95
$\Delta H^{\circ}_{f,0}$	−72.08	−151.21	−211.97	−241.74	−235.85	−213.55
$\Delta H^{\circ}_{f,298.15}$	−73.19	−153.36	−214.94	−247.07	−242.64	−218.09

- Limited by abilities of each code
  - Assumed additive contributions for different effects
  - Choice of methods not always optimal

Table from: Nielsen, Allendorf, J. Phys. Chem. A, **110**, p4093, 2006

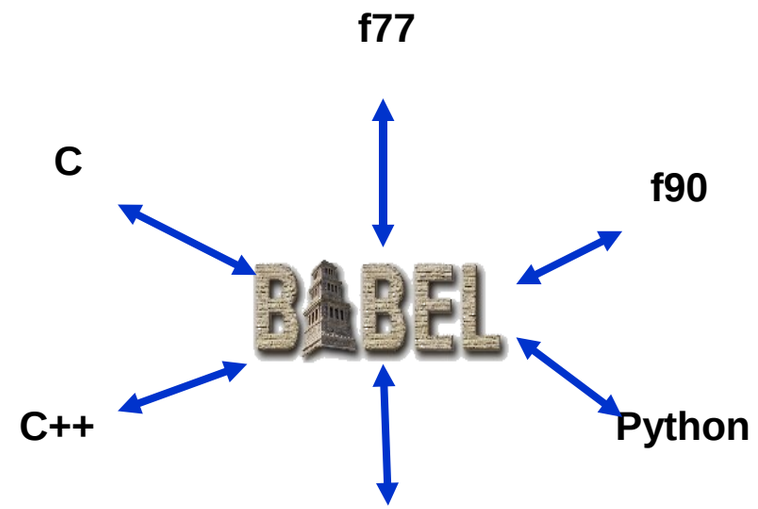
Sandia National Laboratories

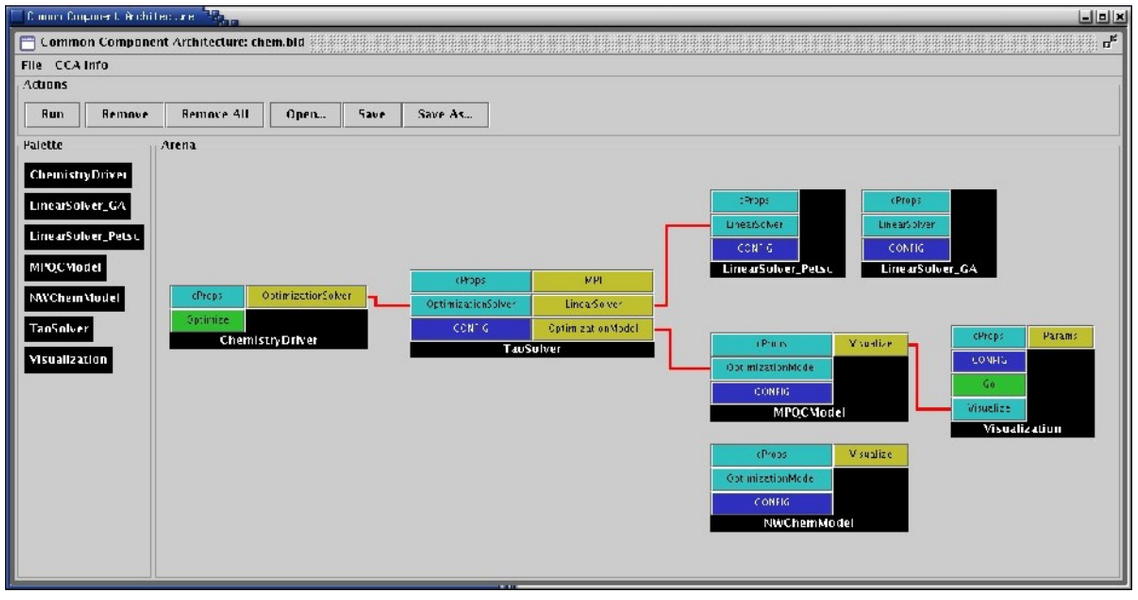
What made this problem so hard?

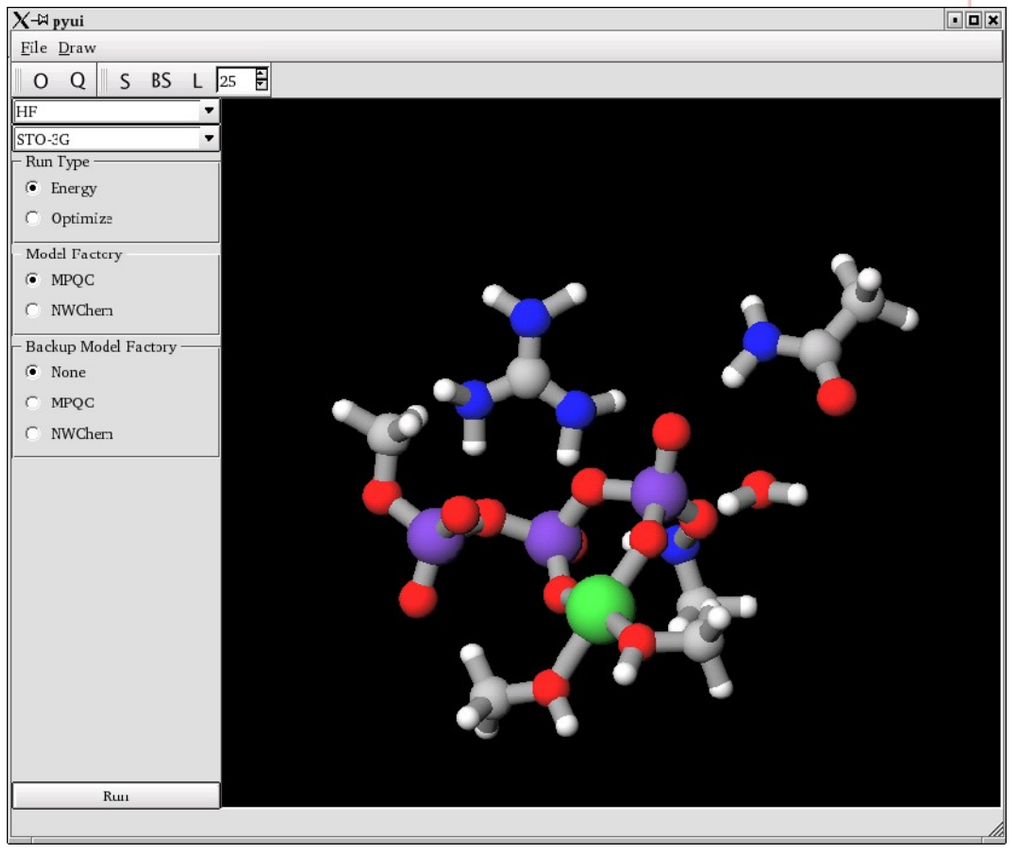
- Different program suites have different strengths
  - Some overlap, but important differences in supported methods
  - Different numerical properties
  - Different levels of support for various architectures
  - Need better ways of interchanging program suites and sharing capabilities between suites
- Gets even harder when quantum chemistry is a component of multi-scale, multi-physics computations
  - Building applications that rely on multiple application domains is even more complex
  - Need ability to export and import capabilities

Component architectures are designed to address these problems

- Language neutral interface specification
  - Different code teams focus only on the common interface
  - Use SIDL: Scientific Interface definition language
- Provides a runtime environment
  - Can dynamically compose an application





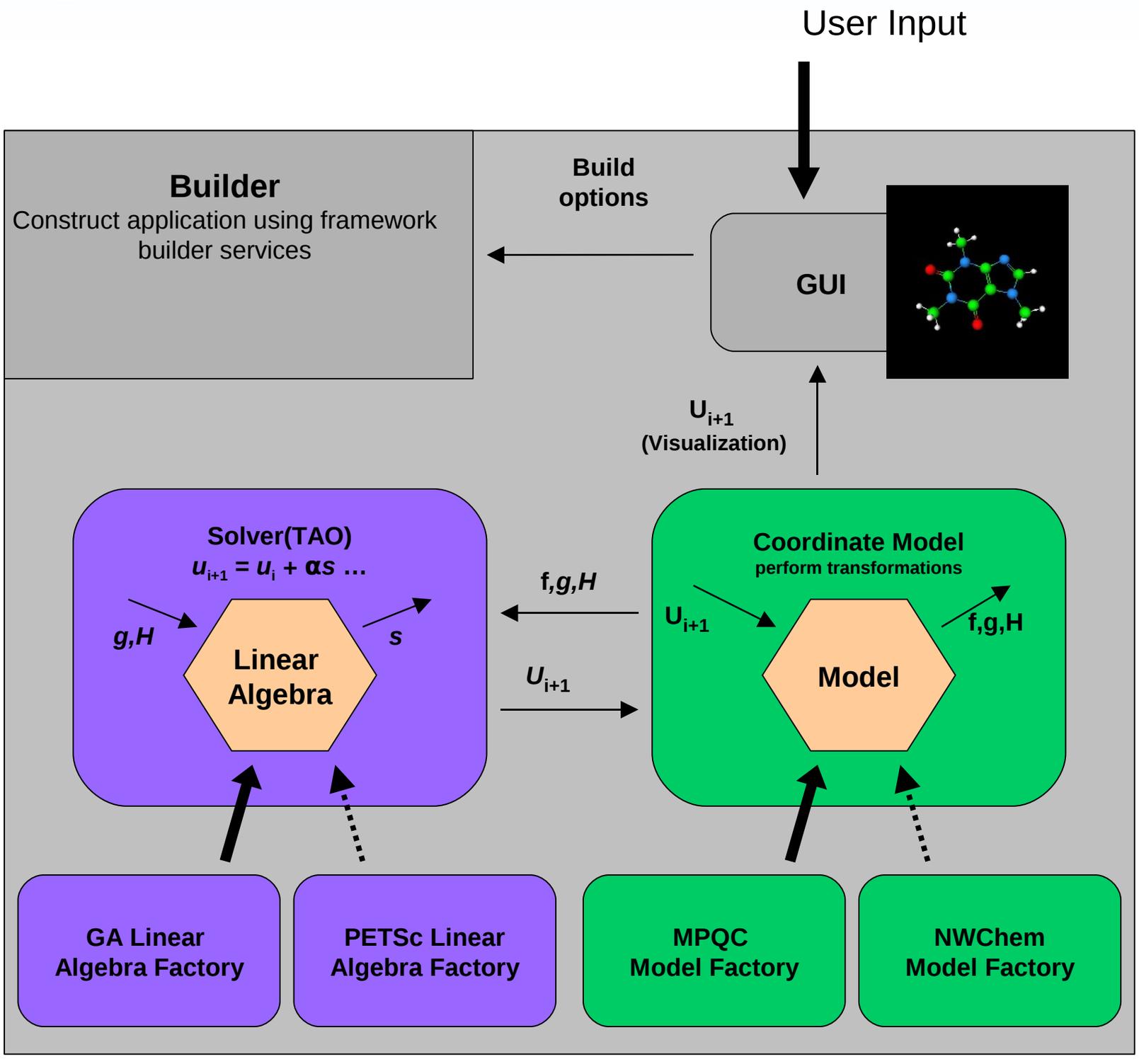


Two applications of the Common Component Architecture

- High-level components for geometry optimization
- Low-level components for integral evaluation

Sandia National Laboratories

High level components and their use in geometry optimization



f energy
u cartesian coordinates
u internal coordinates
g gradient in cartesians
g gradient in internals
H Hessian in cartesians
H Hessian in internals
s update in internals

Chemistry Components
Mathematics Components
Infrastructure
SIDL Classes

Sandia National Laboratories

Examination of the ModelInterface (give it a molecule and get energies, gradients, etc.)

```
package Chemistry { package QC { interface ModelInterface {
  void set_molecule(in Chemistry.MoleculeInterface molecule);
  Chemistry.MoleculeInterface get_molecule();
  double get_energy();
  void set_energy_accuracy(in double acc);
  double get_energy_accuracy();
  void set_do_energy(in bool doit);
  array<double,1> get_gradient();
  void set_gradient_accuracy(in double acc);
  double get_gradient_accuracy();
  array<double,2> get_hessian();
  void set_hessian_accuracy(in double acc);
  double get_hessian_accuracy();
  array<double,2> get_guess_hessian();
  void set_guess_hessian_accuracy(in double acc);
  double get_guess_hessian_accuracy();
  int finalize();
}; }; };
```

Sandia National Laboratories

Enabled direct comparison of various solvers for molecular structures

QC Package	MPQC	MPQC	NWChem	NWChem	NWChem
Algorithm	BFGS	TAO/LMVM	BFGS	BFGS	TAO/LMVM
Line Search	no	yes	no	yes	yes
Guess Hessian	unit	scaled unit	0.5*unit	0.5*unit	scaled unit
Glycine ($C_2H_5NO_2$)	26/26	19/19	33/33	65/33	19/19
		+27%			
Isoprene (C_5H_{10})	75/75	43/43	56/56	89/45	45/45
		+43%			
Phosphoserine ($C_3H_8NO_6P$)	85/85	62/62	79/79	121/61	67/67
		+27%			
Acetylsalicylic Acid ($C_9H_8O_4$)	54/54	48/48	43/43	83/42	51/51
		+21%			
Cholesterol ($C_{27}H_{46}O$)	27/27	30/30	33/33	—/—	30/30
		-11%			

Stand-alone MPQC/NWChem

TAO Solver Component

Number of energy/gradient evaluations required to determine minimum energy structure

Integration gave us insights into problems with our solvers ... and a new solver

Sandia National Laboratories

Low-level components to extend capabilities of programs

- Integrals of many operators are at the core of quantum chemistry programs:

$$\int dr\,\phi_1(r)\nabla^2\phi_2(r)$$

$$\int dr_1 dr_2\,\phi_1(r_1)\frac{1}{r_{12}}\phi_3(r_2)\phi_4(r_2)$$

$$\int dr_1 dr_2\,\phi_1(r_1)\phi_2(r_1)r_{12}\phi_3(r_2)\phi_4(r_2)$$

$$\phi_i(r)=x_i^a\,y_i^b\,z_i^c\,e^{-\alpha_i(r-R_i)^2}$$

$$\int dr_1 dr_2\,\phi_1(r_1)\phi_2(r_1)[\nabla_1^2, r_{12}]\phi_3(r_2)\phi_4(r_2)$$

- Integrals programs do not implement all integral types
- Ability to share integrals and combine packages will
  - enable implementation of new methods
  - permit selection of most efficient package for each machine
- Worst case overhead for using the CCA interface for two electron integrals is about 5-8%

Using the integral components to develop a new method

- Douglas-Kroll allows simple relativistic effect inclusion:

$$h_1^{sf} = c\left(p^2 + c^2\right)^{1/2} - c^2 + A_p V A_p + B_p \overline{p \cdot V\ p} B_p + \ldots + F_p \overline{p \times V\ p} Y_p \overline{p \times V\ p} F_p + \ldots$$

special integral types

- r_{12} methods allow more rapid wfn convergence

$$\Psi^{(1)}_{\text{MP2-R12}} = d^{ij}_{ab} a^{ab}_{ij} \Phi + c^{ij}_{\boldsymbol{kl}} \bar{R}^{kl}_{\alpha\beta} a^{\alpha\beta}_{ij} \Phi$$

special integral type

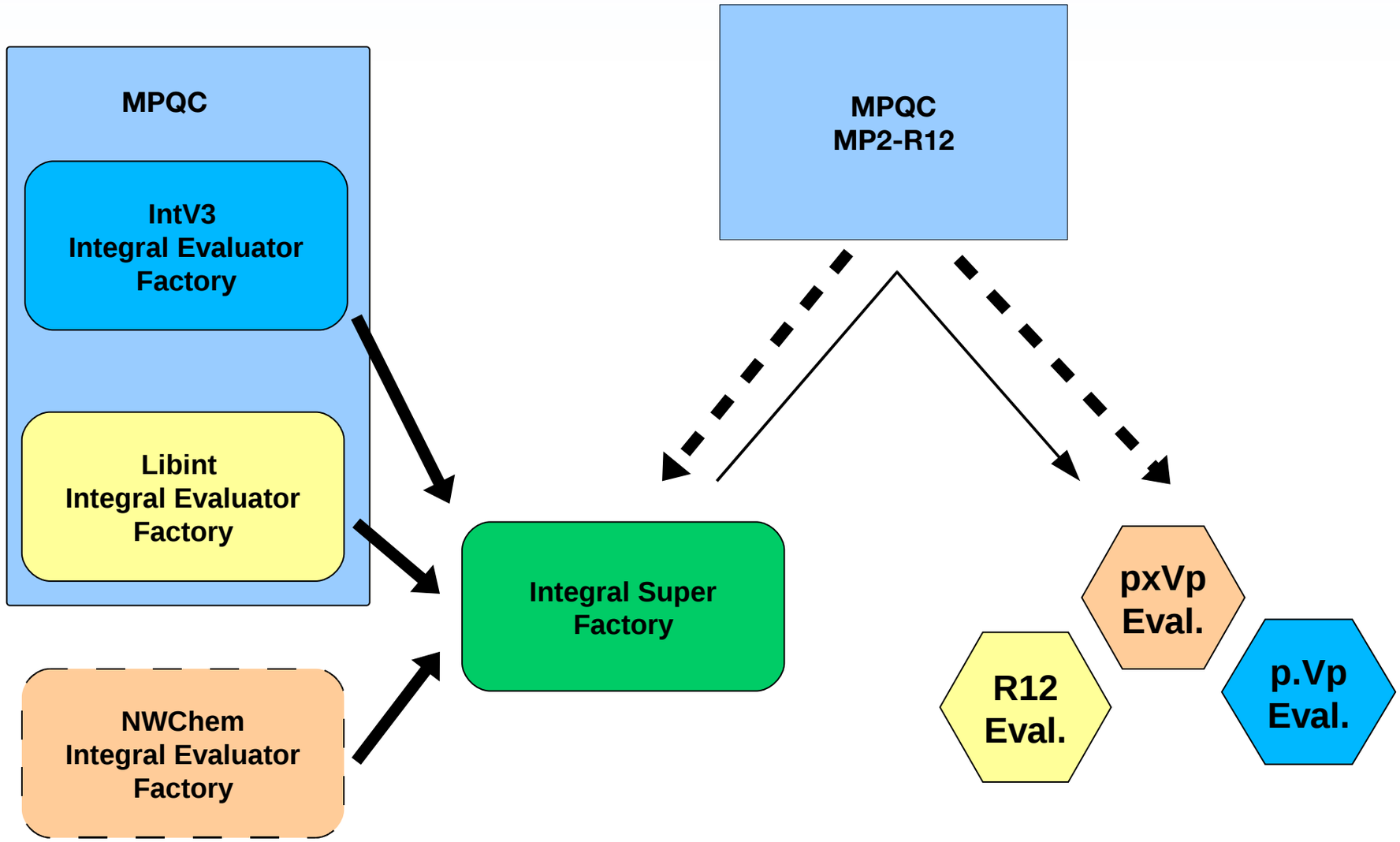
Combination of these methods had not been done

- Even though ideal for high Z core correlation

Sandia National Laboratories

Missing piece: component to combine multiple integral packages

Architecture:



- This is an example of where significant functionality gets implemented into component specific code

Sandia National Laboratories

Applying this to the chromium hydroxide example

- Opportunity to combine three corrections in to one:

  δ[core] + δ[rel.] + δ[basis] → δ[core+rel.+basis]

	$Cr(OH)_6$ Reaction 1		$CrO(OH)_4$ Reaction 2	
ΔE_{rxn}[HF]	−29.66		23.58	
δ[MP2]	+9.81		−7.47	
δ[CCSD]	+5.04		+8.45	
δ[CCSD(T)]	+9.94		+0.96	
	MP2	MP2-R12	MP2	MP2-R12
δ[basis]	−3.55		−2.39	
δ[core]	−0.32		−0.79	
δ[rel]	+2.08		+3.44	
δ[basis+core+rel]	**(Σ = -1.79)**	−1.85	**(Σ = 0.33)**	+0.32
δ[ZPVE]	−8.07	−8.07	−4.99	−4.99
$\Delta H^\circ_{rxn,0}$	−14.73	−14.79	20.79	20.85
$\Delta H^\circ_{f,0}$	−232.98	−232.92	−211.40	−211.45
$\Delta H^\circ_{f,298.15}$	−239.77	−239.71	−215.94	−216.00

Sandia National Laboratories

Other work ongoing work with CCA/Chemistry

- Quantum mechanics/molecular mechanics interface & implementation
- Effective Fragment Potential
- General one body operator interfaces (solvation, for example)
- Python programming interface to MPQC/other CCA QC codes
  - Implementing an ASE Calculator (currently comparing the CCA approach to Boost.Python)

Sandia National Laboratories

- The Common Component Architecture (CCA) and its use in quantum chemistry applications

- Exposing more parallelism in quantum chemistry applications: Moving beyond the MPI and hybrid MPI/Multithreaded programming model

- ArchSim: a macro-scale discrete event simulator for predicating application performance on large-scale parallel machines

Sandia National Laboratories

Example application: Hartree-Fock theory

- Approximate solution to Schrödinger's equation

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

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

$$S_{pq} = \int \chi_p(\boldsymbol{r})\chi_q(\boldsymbol{r})\,d\,\boldsymbol{r} \qquad H_{pq} = \int \chi_p(\boldsymbol{r})\left(\nabla^2 - \sum_{a}^{N_{atom}}\frac{Z_A}{r_A}\right)\chi_q(\boldsymbol{r})\,d\,\boldsymbol{r}$$

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

  - Fock matrix formation:

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

  - Diagonalization:

$$\boldsymbol{F\,C = S\,C\,\epsilon} \qquad \boldsymbol{C\,S\,C^T = 1}$$

  - Density computation:

$$P_{pq} = 2\sum_{a}^{N_{elec.}/2} C_{pa}C_{qa}$$

Solved self consistently, since F depends on P

Sandia National Laboratories

Traditional (imperative) approach for Hartree-Fock

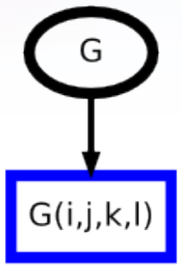
- Programmer specifies where all data resides
- Programmer specifies operations on data
- Typical parallel implementation (but using block cyclic generalized Jacobi eigensolver with tournament ordering to expose more parallelism):

```
Form the atomic orbital overlap matrix, S
Form the atomic orbital Fock matrix, F, computing integrals, G, as needed.
Synchronize (barrier or reduce-broadcast) 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 on all nodes
      Rotate F and S
      Synchronize so that F and S are complete on all nodes.
   End loop over independent shell pairs
End eigensolver iterations
```

Data-driven programming approach

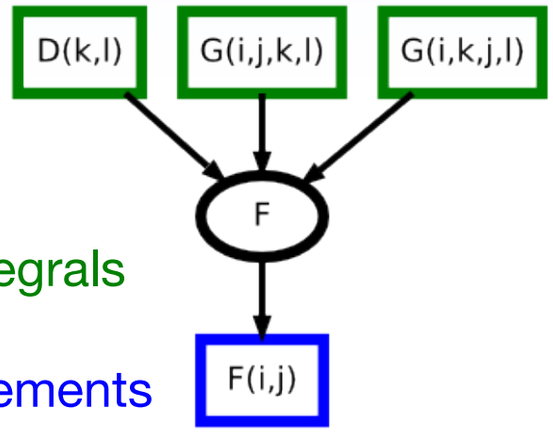
- Identify the elementary operations that transform one set of data into another
- All data that is used by an operation is specified as an input to that operation —global data is not allowed
- Let a runtime system schedule work onto the machine
  - The programmer does not explicitly dictate parallelism—it is only necessary to pick elementary operations that are not too tightly coupled
  - Significant bonus: The runtime would have sufficient information to provide fault tolerance to the running application
  - There will be extra communication and scheduling overhead, but potential for better scalability, better portability, and easier programming could outweigh this
- In the following, small test cases are examined for their parallelization potential by simulating performance for data-driven execution

Elementary operations for Hartree-Fock in terms of data dependencies



Two electron integrals formation, G:

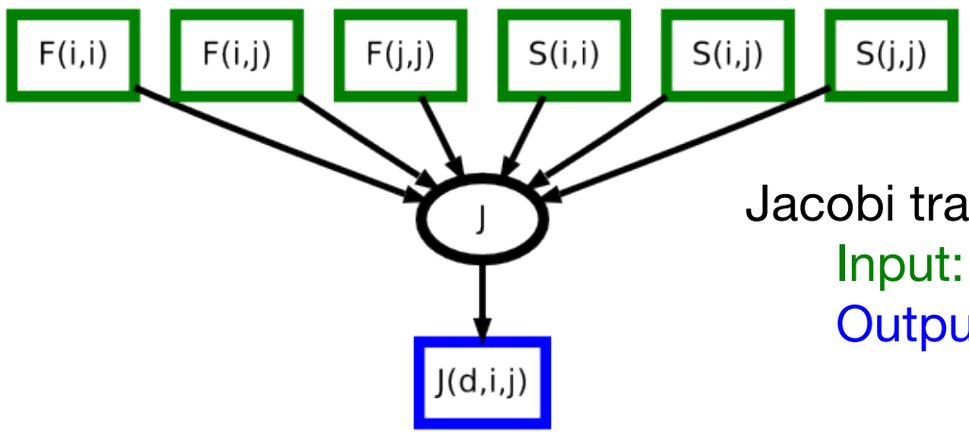
Output: G(i,j,k,l) for a tshell quartet



Fock matrix formation, F:

Input: Two electron integrals and density matrix

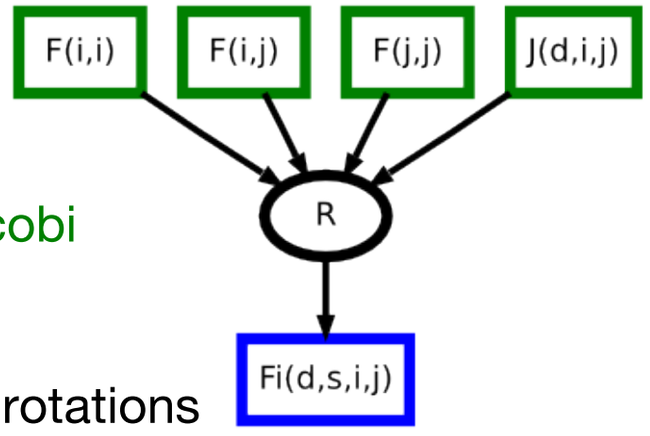
Output: Fock matrix elements for a shell pair



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

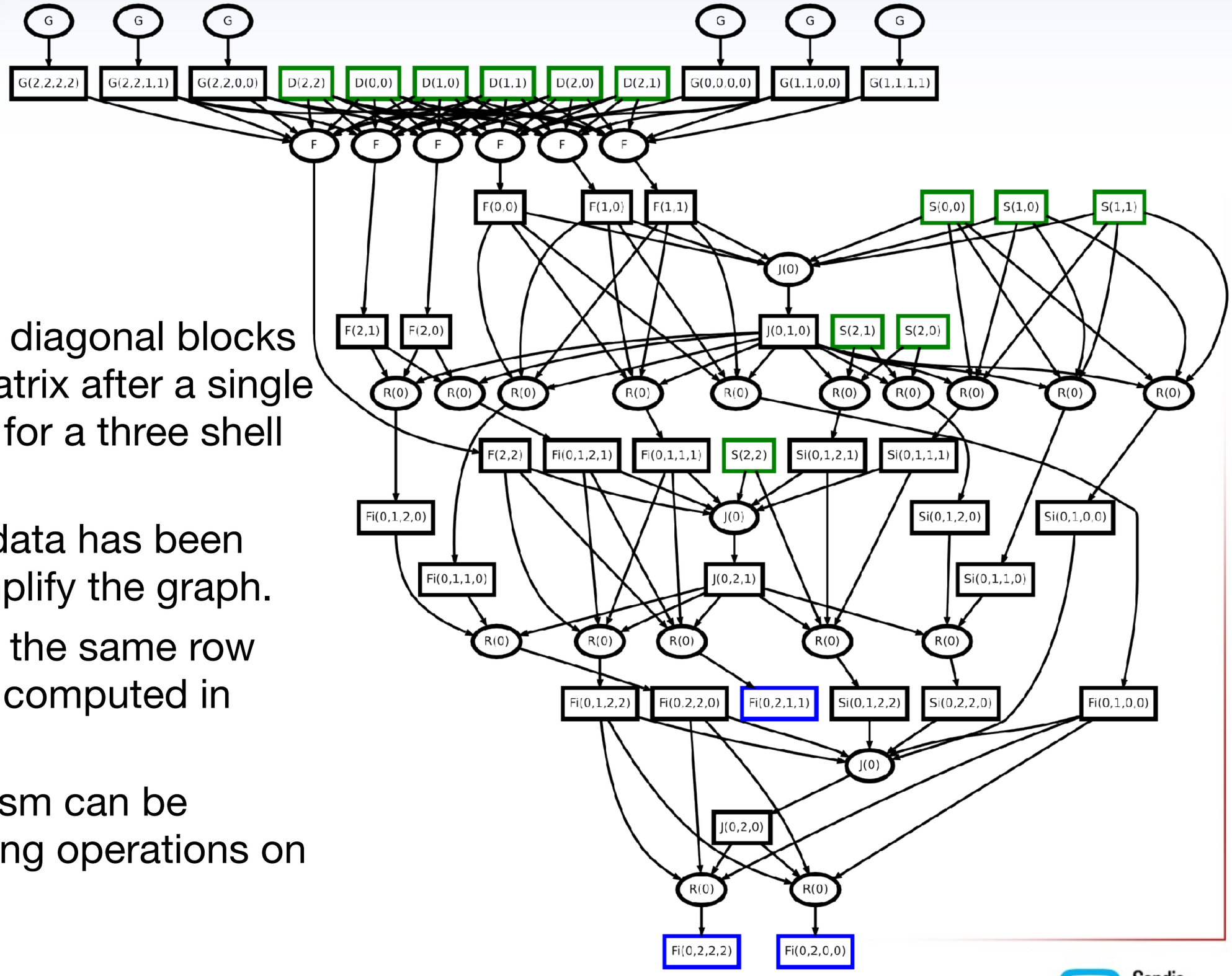
Output: Transformed matrix elements

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

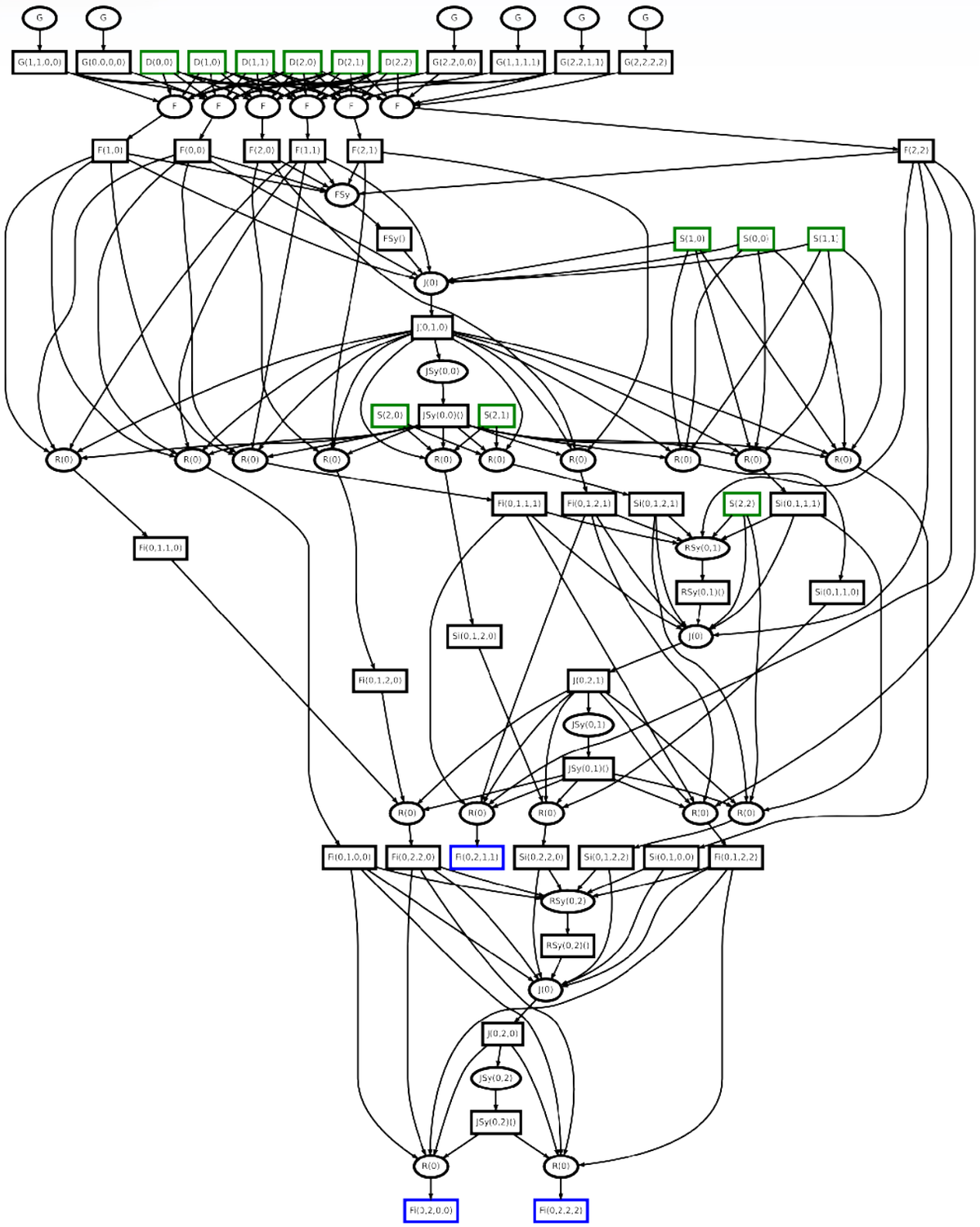
- Computes the diagonal blocks of the Fock matrix after a single Jacobi sweep for a three shell system.
- Certain input data has been omitted to simplify the graph.
- Operations on the same row (ovals) can be computed in parallel
- Some parallelism can be exploited among operations on different rows



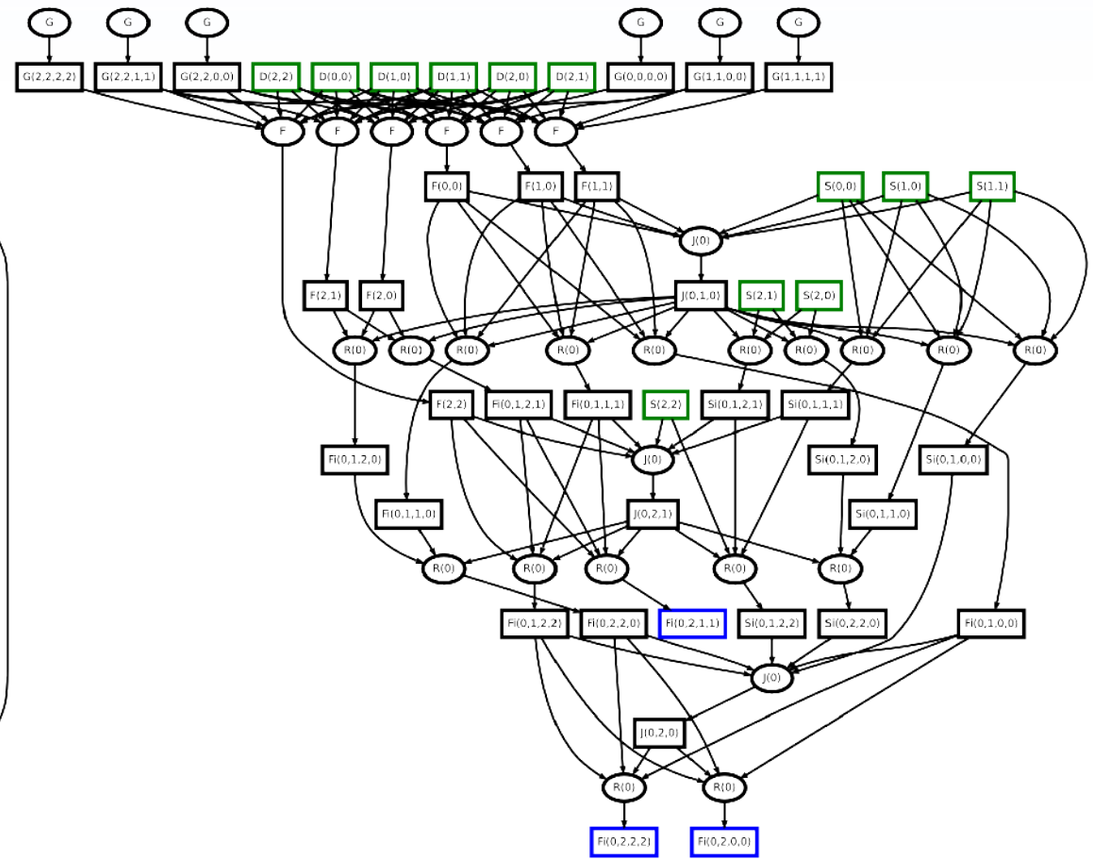
Sandia National Laboratories

Comparison of data dependencies with and without synchronization

With synchronization:



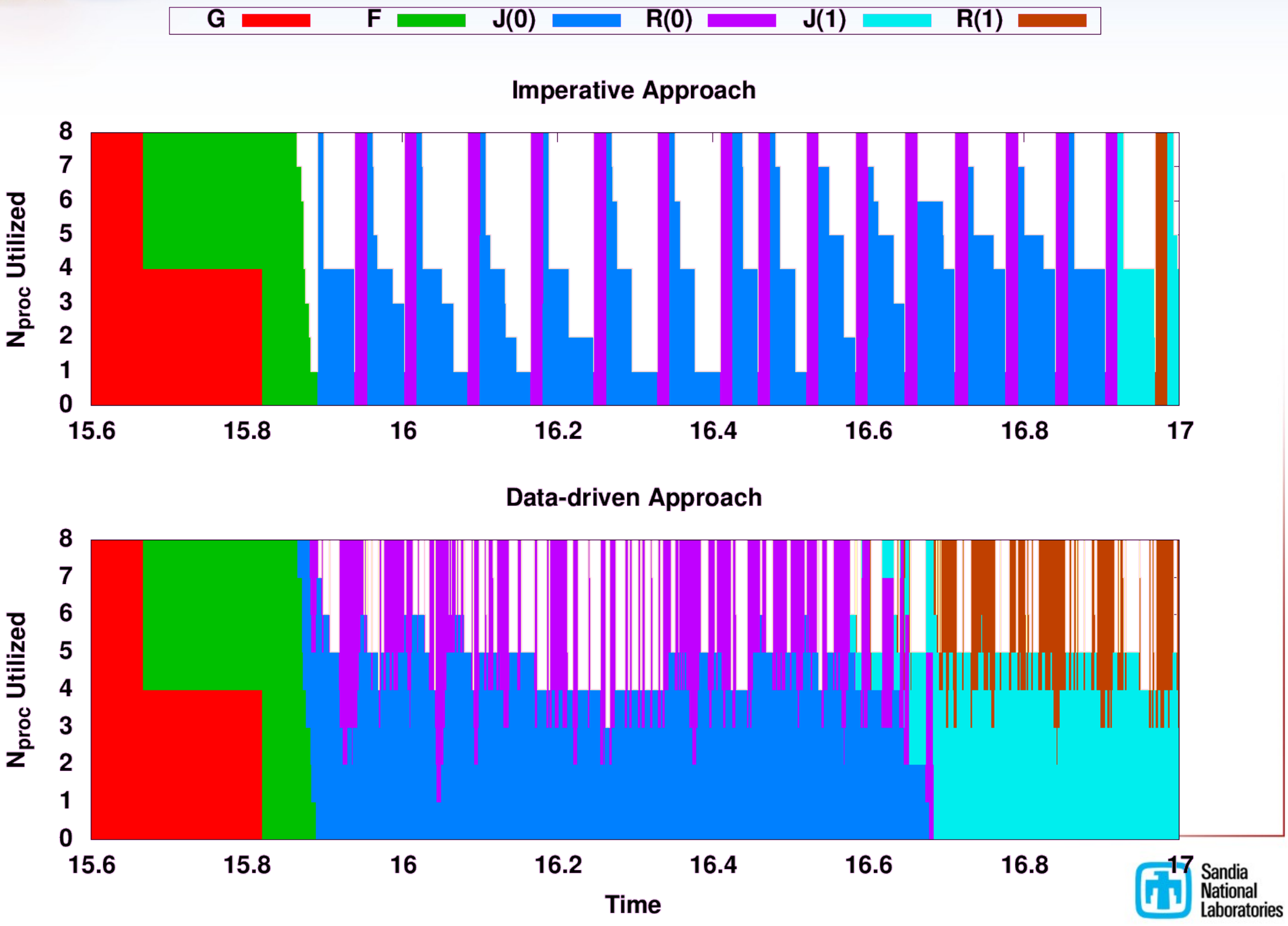
Without synchronization:



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

Simulated timings for 16 shells on 8 processors



Sandia National Laboratories

Summary of data-driven Hartree-Fock work

- Data-driven programming is a natural way to expose parallelism
  - Programmer is not concerned with explicit details of parallelization
  - Can explore more parallelism more easily than traditional approaches
- Simulated results show that a data-driven approach can improve performance
- Much more work is needed
  - Details of runtime system need to be worked out
  - Communication overhead needs to be studied
  - Overhead of scheduling algorithms must be considered
  - May require extended memory semantics or other architectural extensions to be efficient

- The Common Component Architecture (CCA) and its use in quantum chemistry applications

- Exposing more parallelism in quantum chemistry applications:
  Moving beyond the MPI and hybrid MPI/Multithreaded programming model

- SST/macro: a macro-scale discrete event simulator for predicating application performance on large-scale parallel machines

Sandia National Laboratories

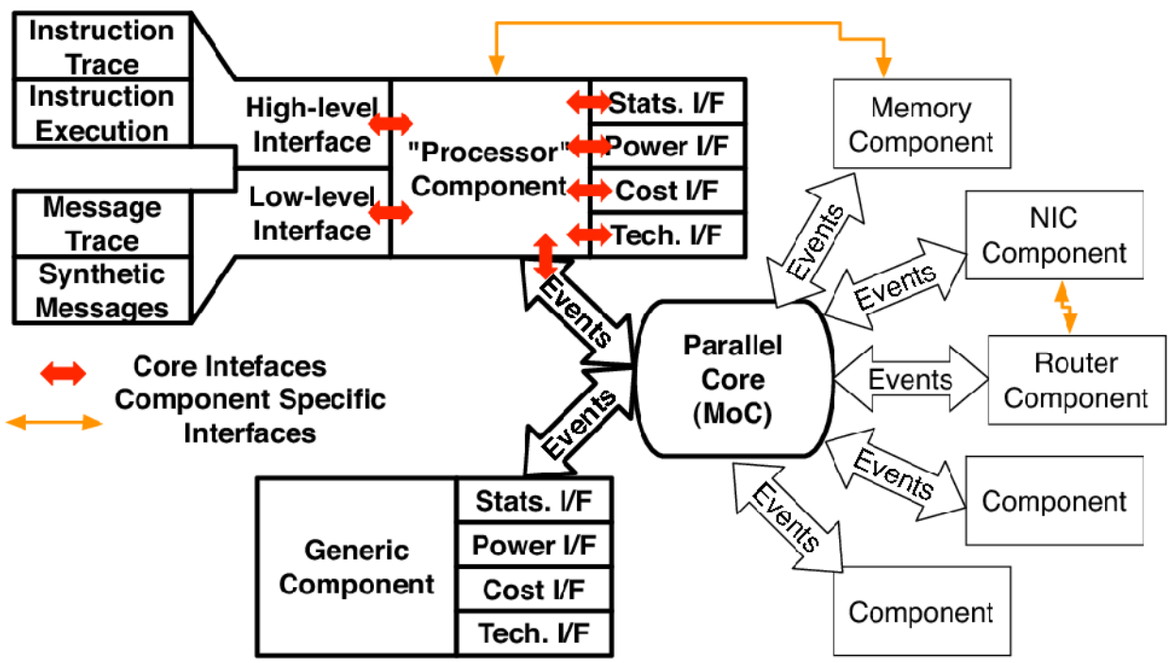
SST/macro: a macroscale simulator for predicating performance of large-scale parallel machines

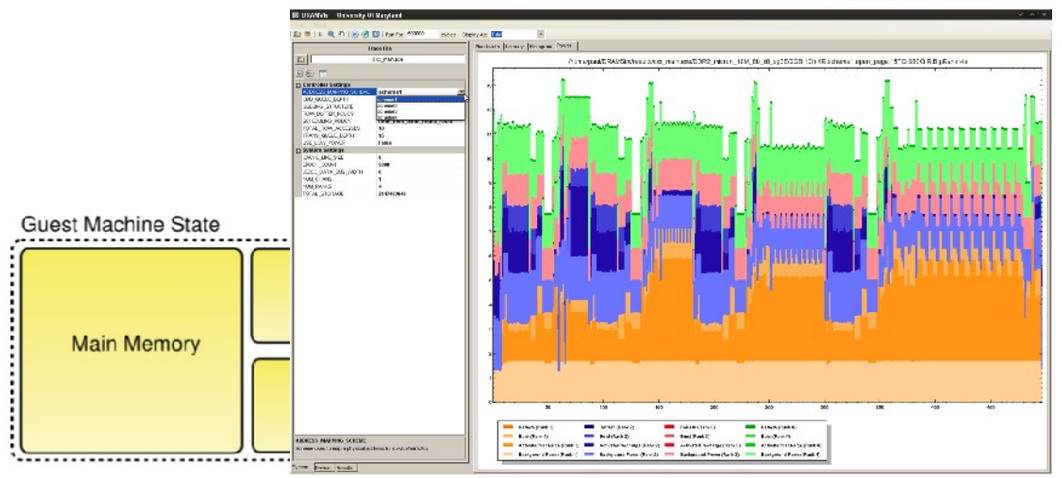
- Current approaches to architecture and application scaling have gaps
  - Rough estimates and guidelines are used for architecture specifications
  - Problems related to application scaling not addressed until delivery
- Simulation can assist in architecture and algorithm design
  - Trade-offs between various algorithm designs can be estimated without access to hardware or the need to write a complete application
  - Can inform procurements as well as the design of machines and algorithms.
- Several simulators exist—why another?
  - Current component simulators are isolated artifacts
  - Need flexibility to couple together a variety of components into a simulator of the desired architecture with the appropriate fidelity/cost.
  - SST/macro is developed in the context of Sandia's Structural Simulation Toolkit (SST)—SST has both microscale (clock-level accuracy) and macroscale components

Sandia National Laboratories

Institute for Advanced Architecture and Algorithms (IAA)

- SST is a component of a larger project, the IAA, which is a partnership between Sandia and Oak Ridge National Laboratories
- IAA seeks to enable extreme-scale computing
  - Focused R&D on key impediments to high performance in partnership with industry and academia
  - Foster the integrated co-design of architectures and algorithms
  - Partner with other agencies industry and academia
  - Impact vendor roadmaps
  - Deploy prototypes to prove the technologies
- IAA is locus for a community-wide architecture simulation project





DRAMSim II

Qemu

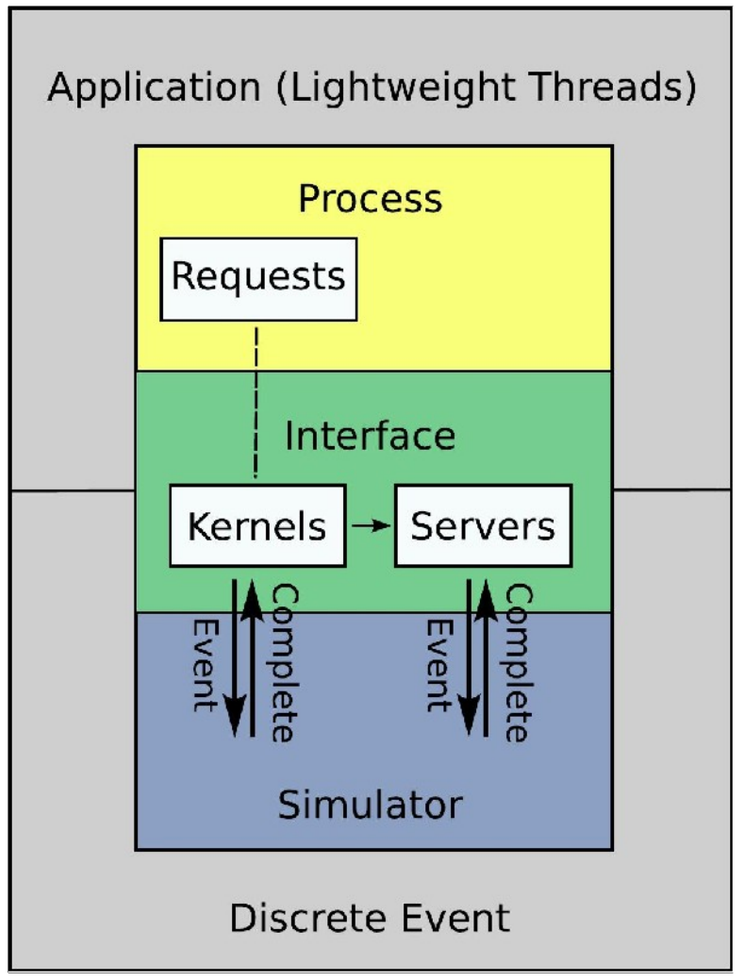
SST/macro project goals

Provide a SST simulation components that:

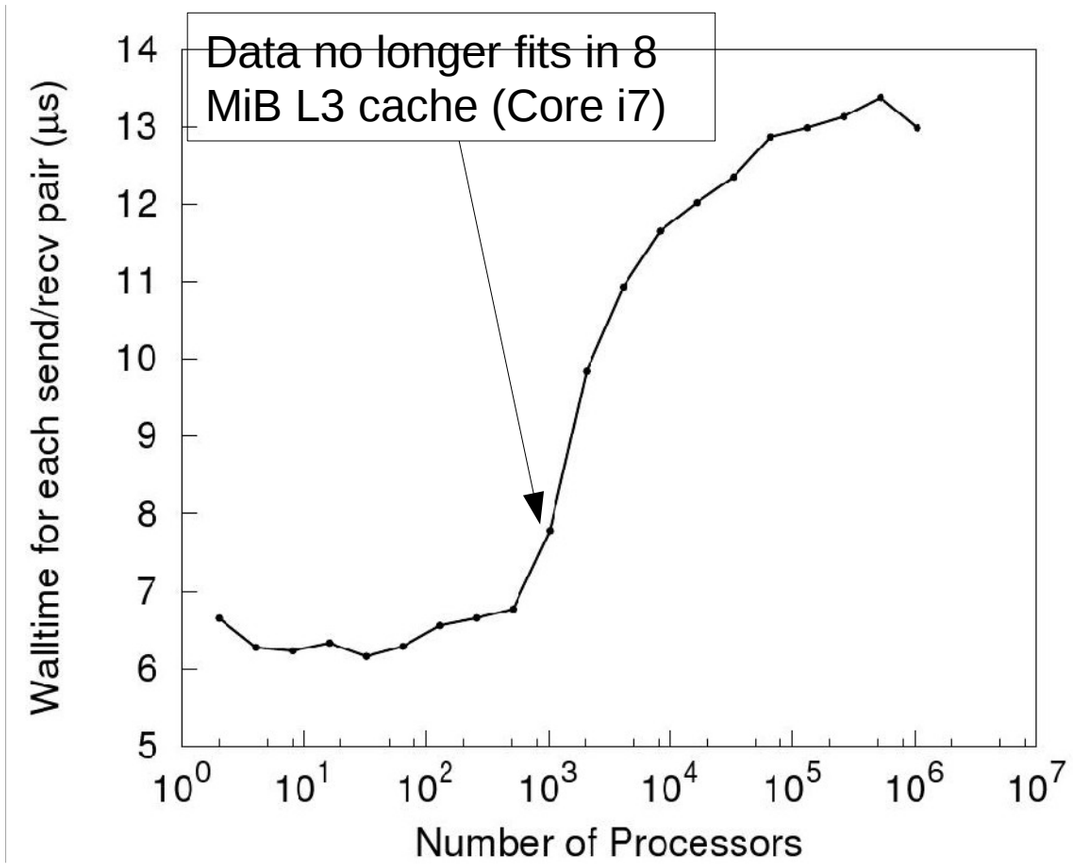
- Takes into account the communication and computation coupling that applications exhibit
  - Trace files that record an actual application run
  - Skeleton applications that mimic application behaviour
- Can run very large simulations
  - 1,000,000's of cores
  - Full multi-physics simulations
- Allow investigation of effects of
  - Topology and process placement
  - Changes in the routing algorithm
  - Changes in the network latency and bandwidth
  - Having many cores share the same network interface
  - Interactions between different jobs running on the same machine
  - Modifications to the MPI layer
  - Modifications of the application

SST/macro design

- Event interface is now used: permits integration into the hybrid multi-scale SST framework.
- Extremely lightweight events: > 150,000 MPI ping pong round trips per second



SST/macro architecture.



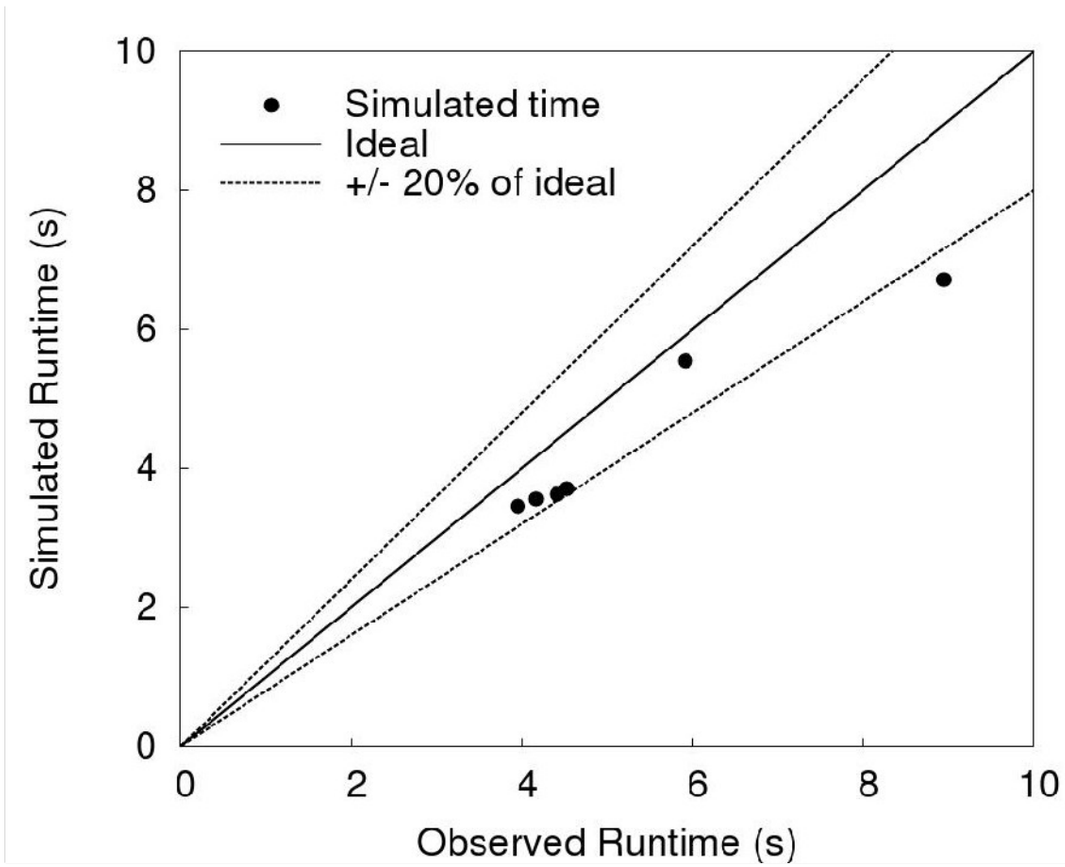
SST/macro performance.

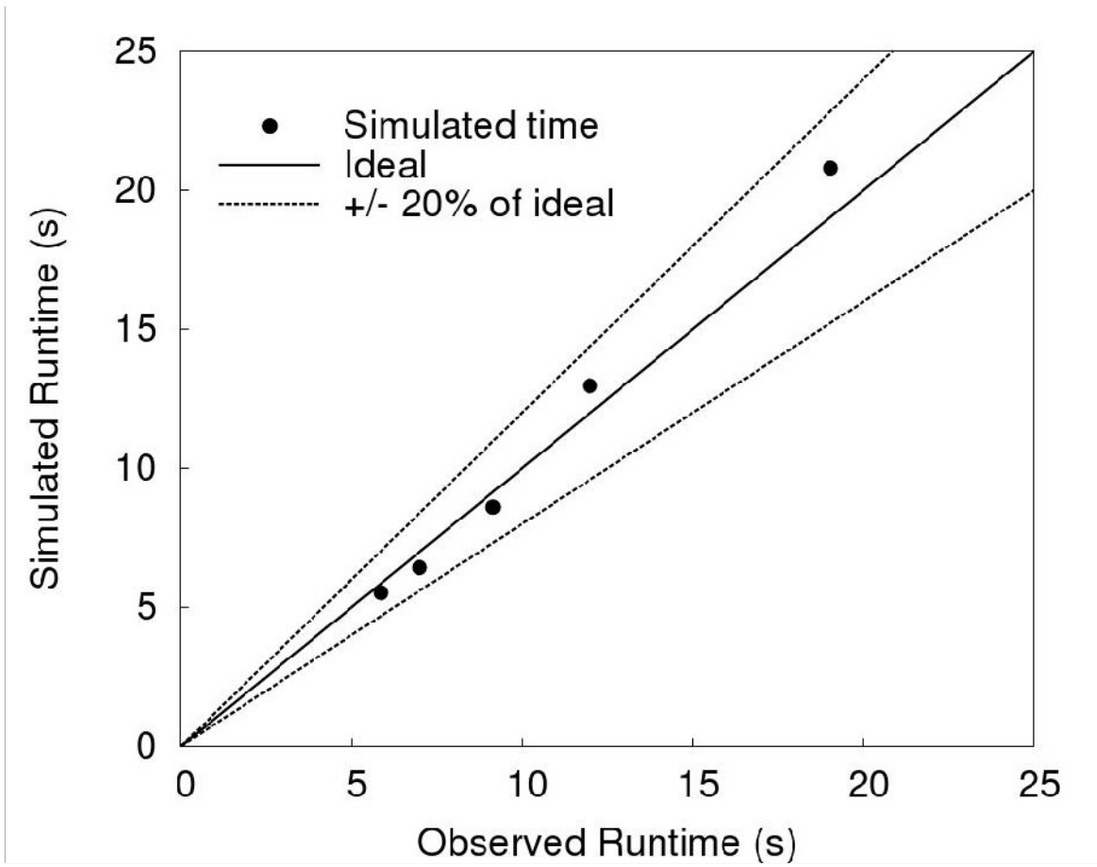
National Nuclear Security Administration

Sandia National Laboratories

Validation of simulator

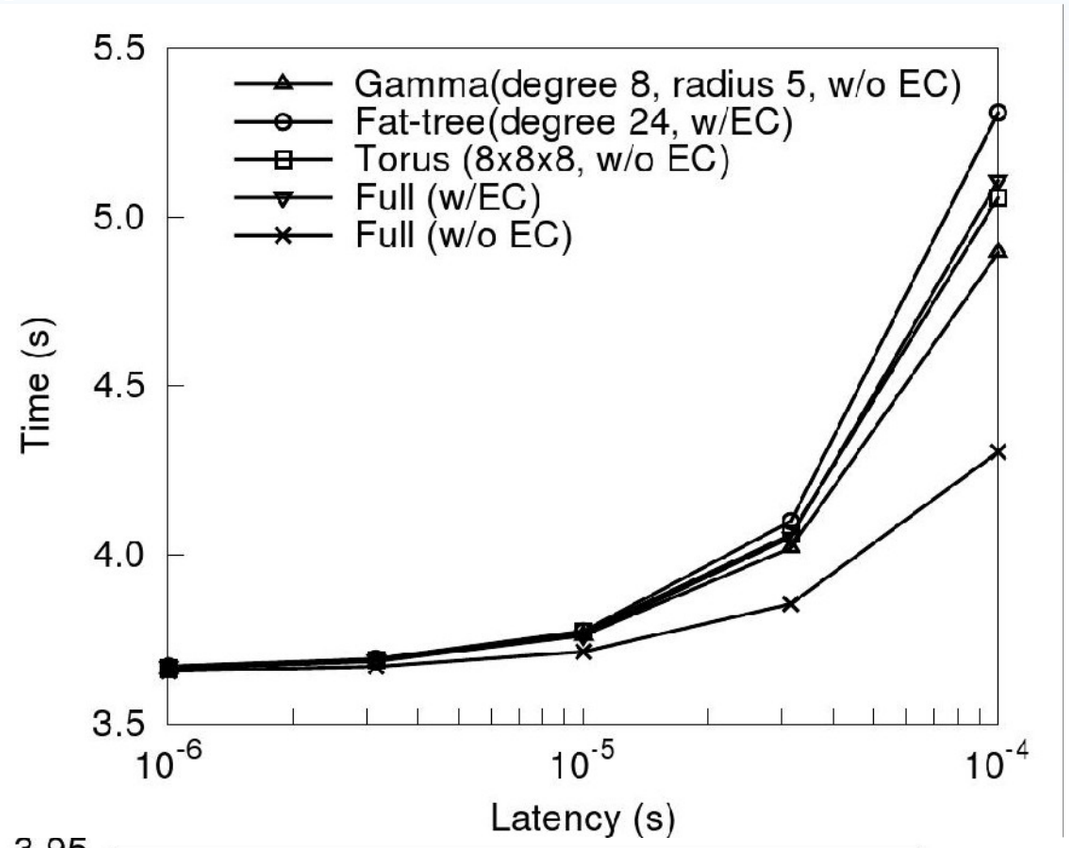
- Used AMG2006: part of NNSA ASC/Sequoia acceptance tests and being considered for Zia
- Collected traces with dumpi and played back through simulator

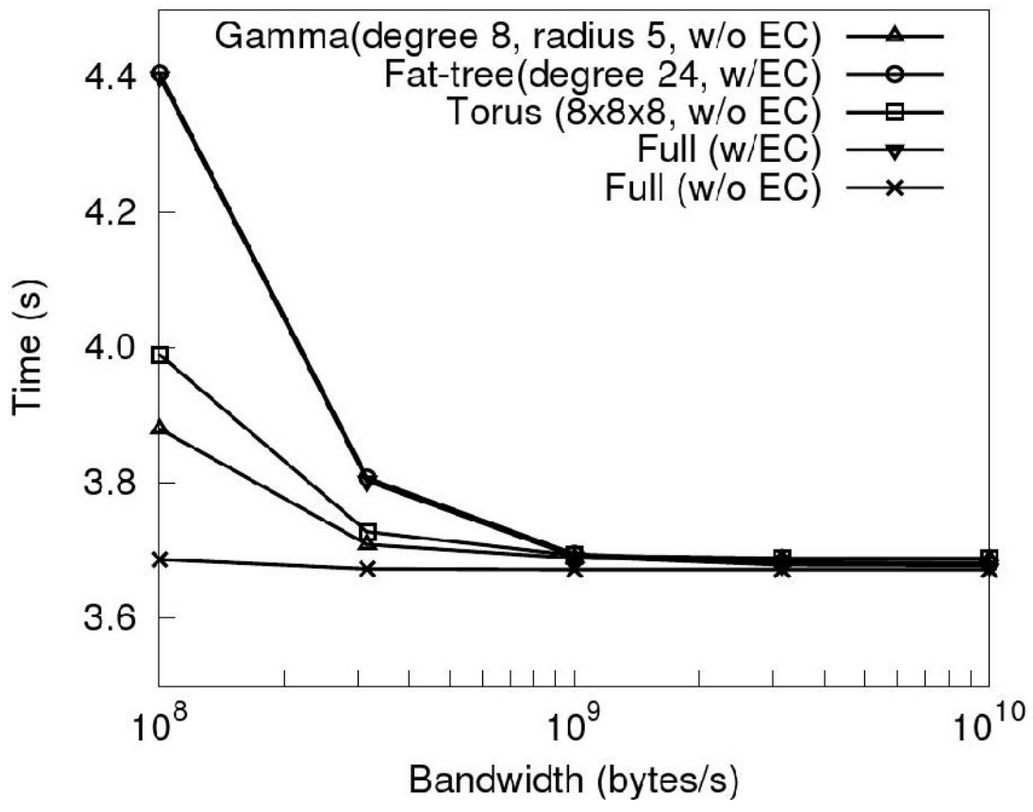


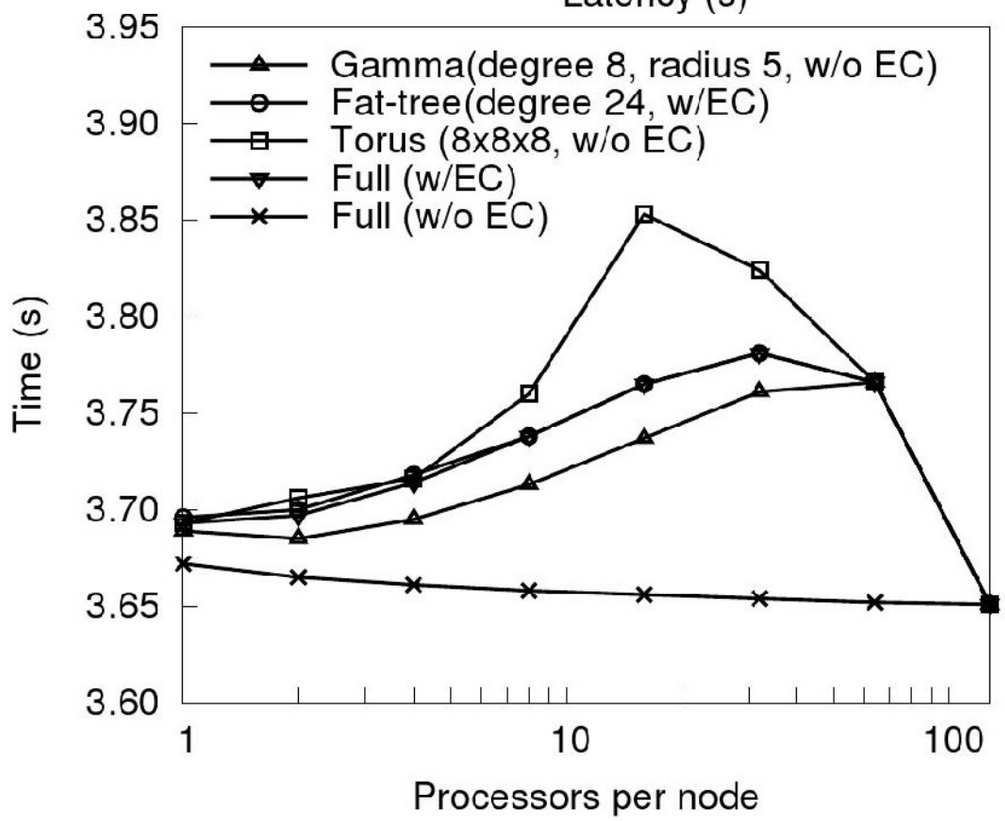


Sensitivity of AMG2006 to architecture parameters

- Examined simulated time to solution as bandwidth, latency, and processors per node are varied for several topologies.



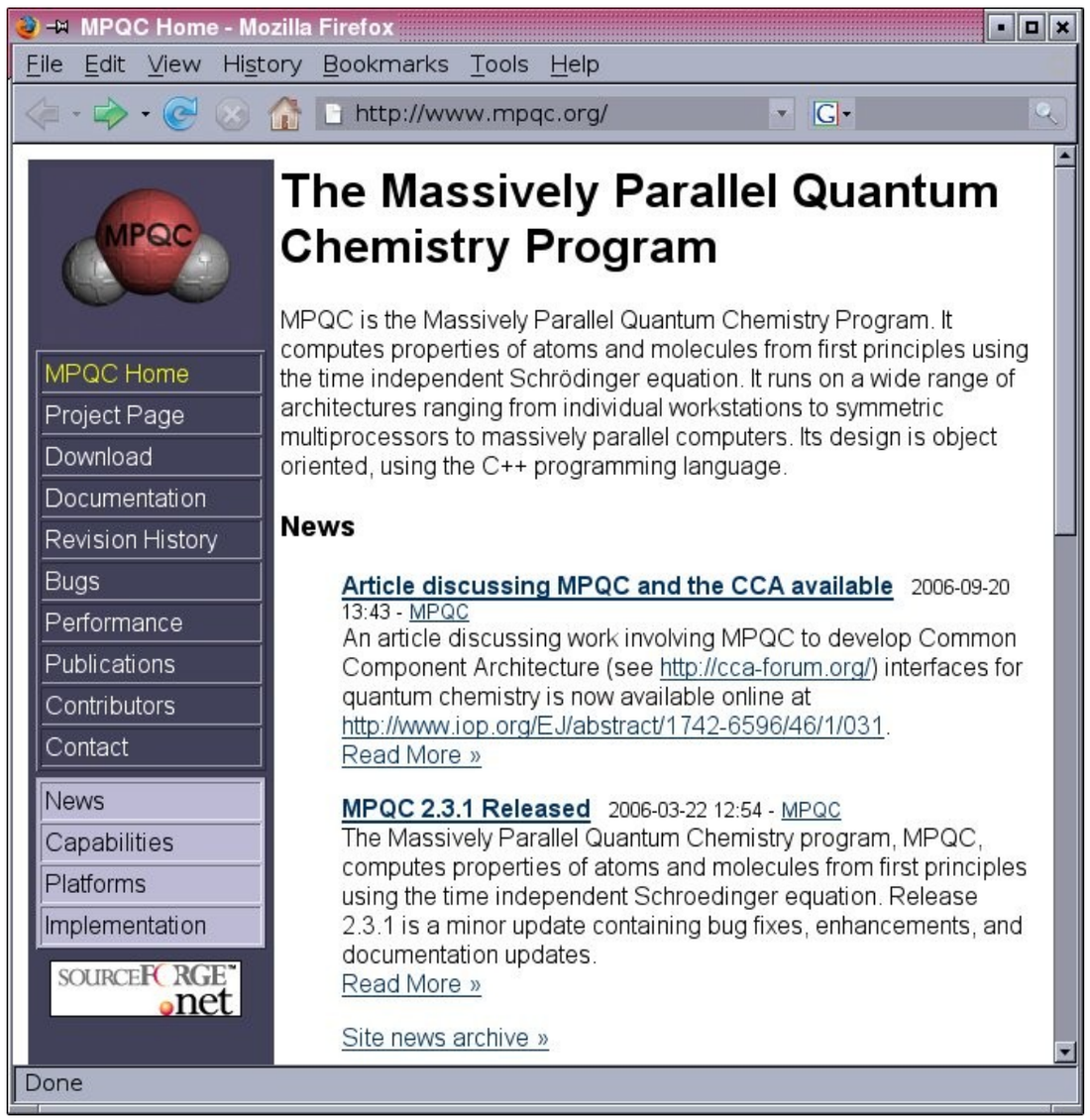


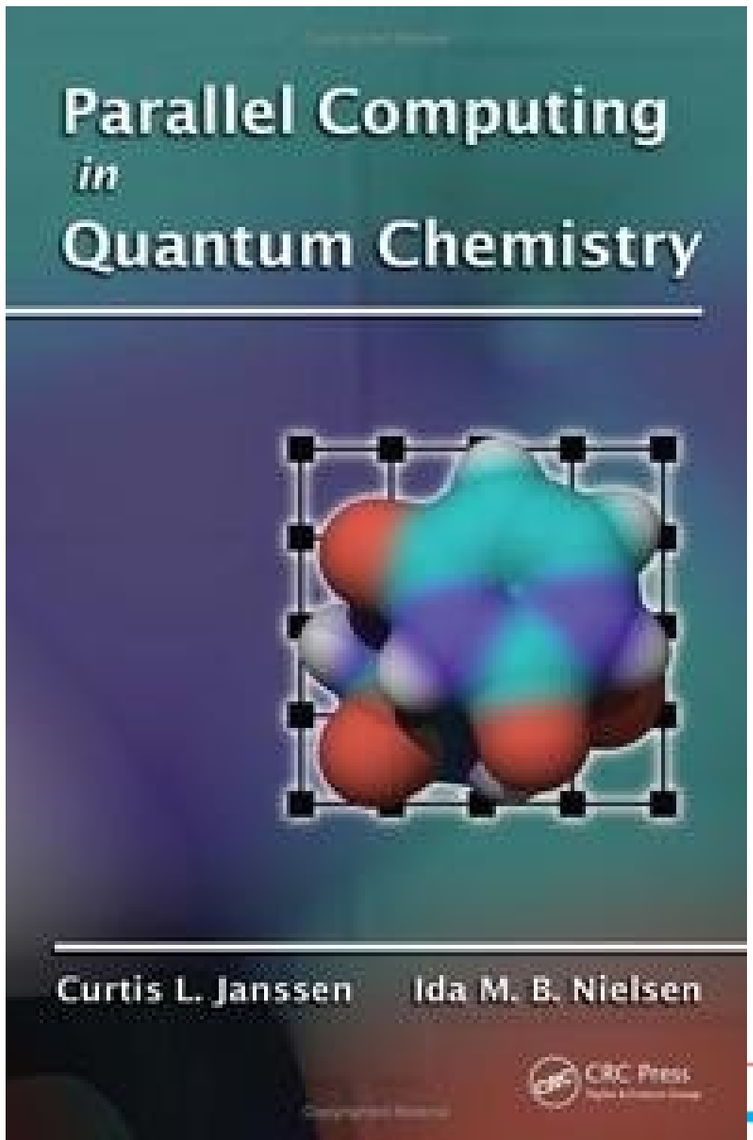


Sandia National Laboratories

Additional information

- Contact info: Curtis Janssen, cljanss@sandia.gov
- MPQC Home Page: http://www.mpqc.org
- Parallel QC background: *Parallel Computing in Quantum Chemistry*, Janssen and Nielsen, CRC Press, April 2008





Sandia National Laboratories

End

Sandia National Laboratories