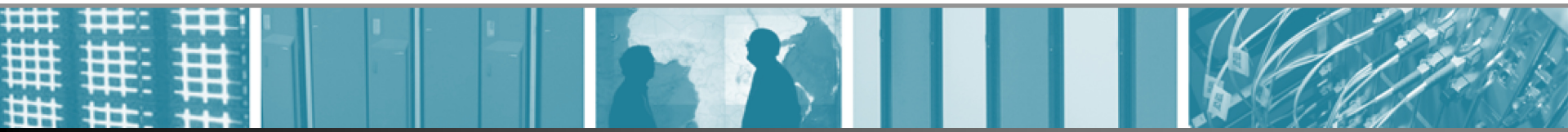
Accelerating the Computation of Detailed Chemical Reaction Kinetics for Simulating Combustion of Complex Fuels

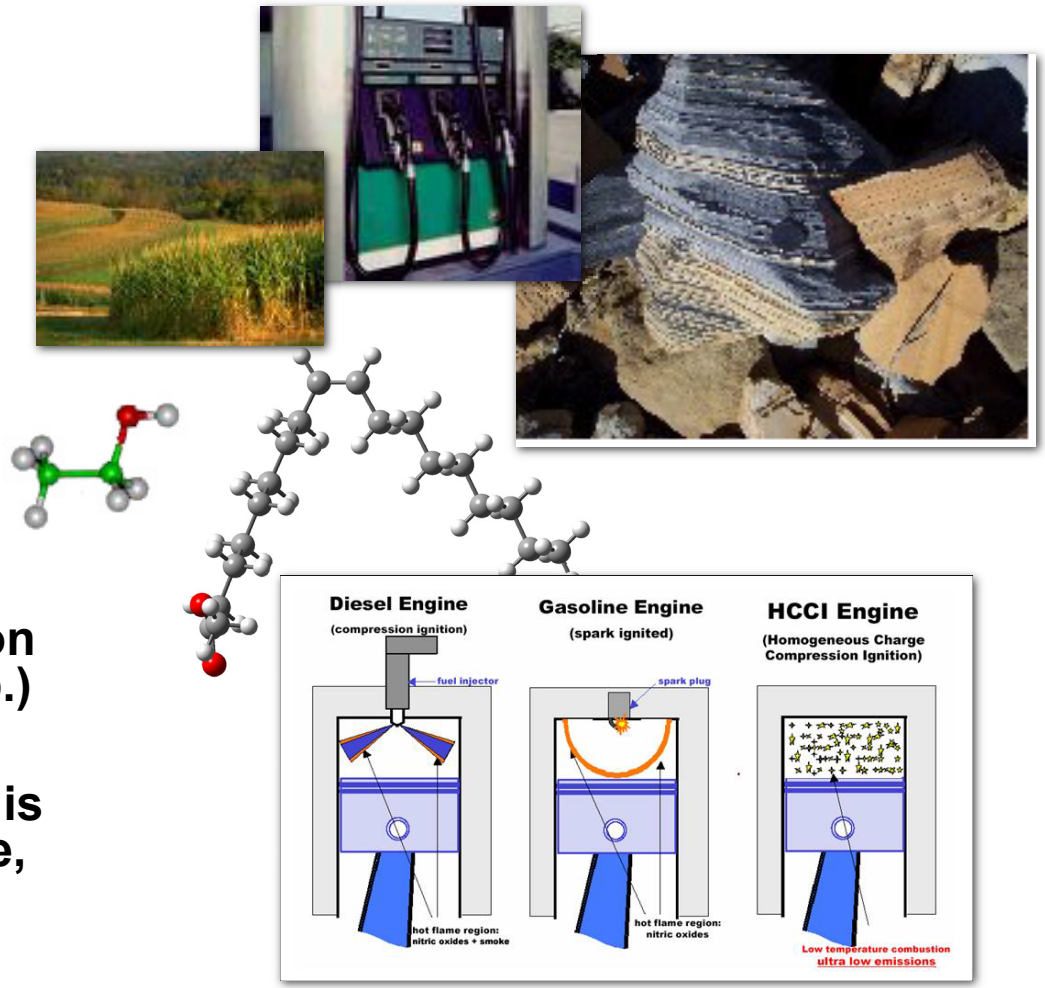
Ramanan Sankaran
Computational Scientist
Oak Ridge National Laboratory

Cray Technical Workshop on XK6 Programming (Oct 10th 2012)



Motivation: Changing World of Fuels and Engines

- Fuel streams are rapidly evolving
  - Heavy hydrocarbons
    - ✓ Oil sands
    - ✓ Oil shale
    - ✓ Coal
  - New renewable fuel sources
    - ✓ Ethanol
    - ✓ Biodiesel
- New engine technologies
  - Direct Injection (DI
  - Homogeneous Charge Compression Ignition (HCCI)
  - Low-temperature combustion
- New mixed modes of combustion (dilute, high-pressure, low-temp.)
- Sound scientific understanding is necessary to develop predictive, validated multi-scale models!



Combustion chemistry

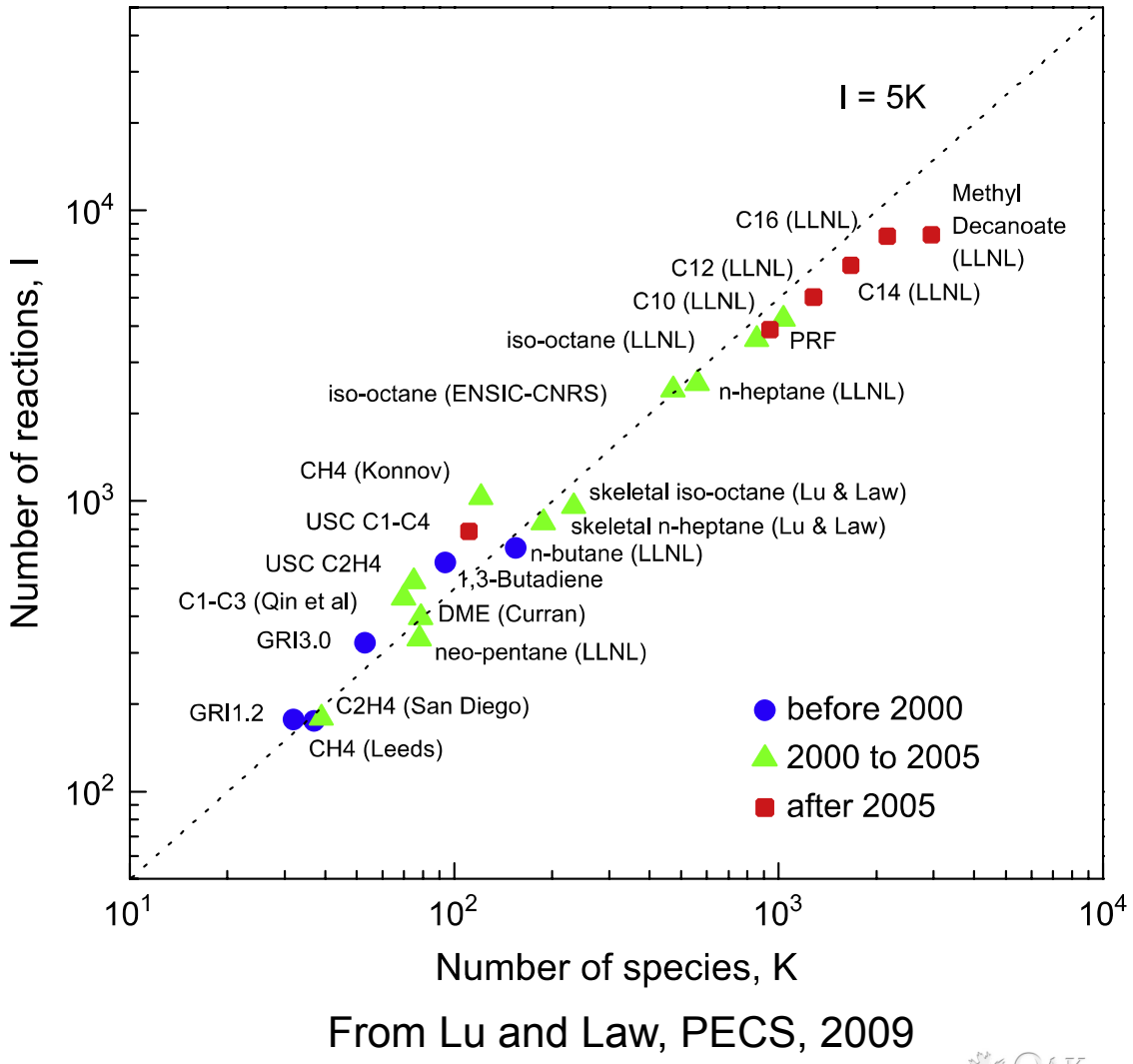
- Example, natural gas combustion

$$CH_4 + 2O_2 \Rightarrow CO_2 + H_2O$$

- Occurs through a reaction network producing and consuming intermediate species
  - CO, OH, H_2O_2, HO_2, CH_3, …
- Detailed chemical mechanisms are needed to compute
  - Flame structure and stability
  - Emissions
  - Validate reduced reaction mechanisms

Detailed chemical kinetics are expensive

- Chemical source term evaluation is computationally intensive
- Thousands of elementary reaction steps accumulated to global species reaction rates
- Often the target for model reductions or algorithmic improvements
- How fast can we compute detailed chemical kinetics on accelerators?



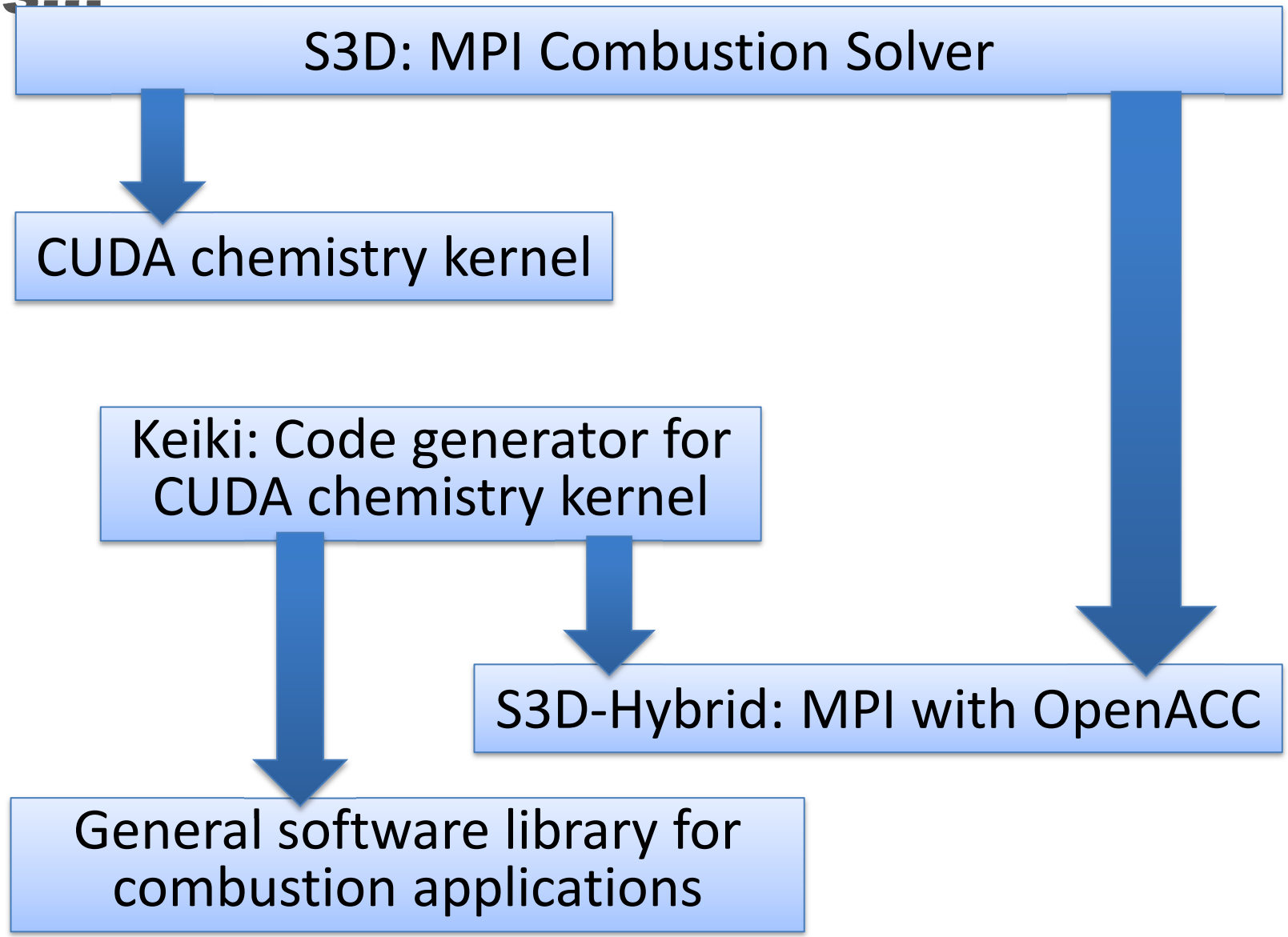
From Lu and Law, PECS, 2009

Chemistry Kernels

- Reaction rates, thermodynamic properties and transport coefficients account for 55% of time.
  - Complex chemical kinetic models needed to address multi-stage ignition and flame dynamics
- Point-wise functions that are independent of DNS software's mesh data structure and MPI-layer
  - Uses Chemkin API
- Used across other combustion codes in the community.
  - Impacts other HPC and workstation-scale combustion applications.
- Accelerator library targets the DNS chemistry needs and beyond

Kyle Spafford (ORNL) et al., "Accelerating S3D: A GPGPU Case Study," in Seventh International Workshop on Algorithms, Models, and Tools for Parallel Computing on Heterogeneous Platforms (HeteroPar 2009). Delft, The Netherlands, 2009

Background: In the beginning there was…



Accelerator library for combustion kinetics

- Conservation equation in a typical combustion application

$$\frac{\partial Q}{\partial t} + u\frac{\partial Q}{\partial x} + \cdots = \dot{\omega}_Q$$

$$\dot{\omega}_Q = f\left(\dot{\omega}_C\right)$$

where $\dot{\omega}_C$ is the rate of chemical kinetics

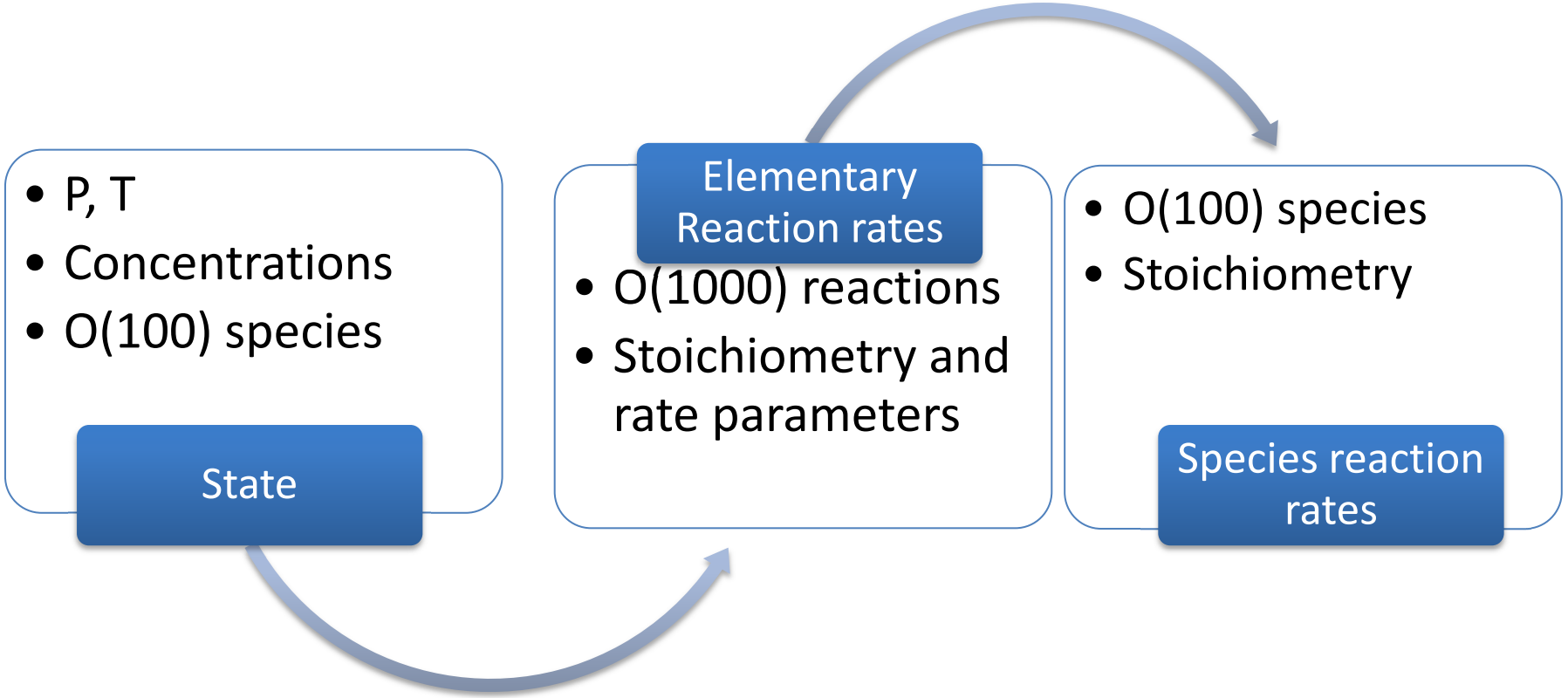
- Chemistry kernel evaluates the chemical kinetics for large mechanisms.
- Well optimized on CPUs and achieves more than 20% of peak on AMD opterons
- Porting to GPU and larger chemistry requires higher levels of parallelism

Parallelizing reaction kinetics (CKWYP)

$$\dot{\omega}_M = \sum_N \Omega(P, T, Y_M)$$

- Grid-level parallelism (several independent states)
  - Will provide MPI parallelism
  - In some cases, also SMP-like parallelism
- Grid-level vectorization does not provide sufficient performance
  - 32(states) * 4000 variables * 8bytes = 1000 kB
- Current capacity in shared memory/L1 cache = 64kB
- Need to go deeper for vector parallelism
  - Equation level parallelism

Data flow in the rates kernel



- Data movement should be minimized while also vectorizing
- Expose concurrency (independent blocks) within the reaction network
- Redundant computation to achieve parallelism

Partitioning at species/reaction level

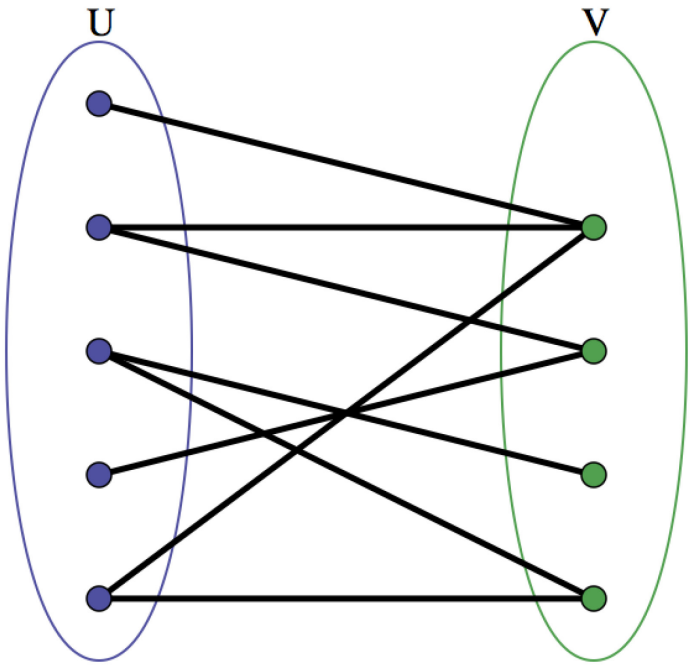
- Similar to partitioning the grid for distributed memory parallelism (MPI)
- Why partition the computation at species/reaction level?
  - Asynchronous execution to hide latencies and data transfers (memcpy across PCI)
  - Distribute work to multiple accelerators assigned to a single host
  - Allow finer grained parallelism at the chemistry level to multiply the scalability of the flow solver
- **Keiki** treats the chemical kinetics as a graph and partitions it to minimize edgecut and maximize parallel performance

Reaction network as a graph

- Chemical reaction network is a bi-partite graph between two sets of vertices
  - The species form one set
  - The reactions form the second set
  - Stoichiometry of the reaction network defines the graph
- The adjacency matrix of the graph is

$$A = \begin{pmatrix} 0 & B \\ B^T & 0 \end{pmatrix}$$

- Where B is the M x N stoichiometry matrix

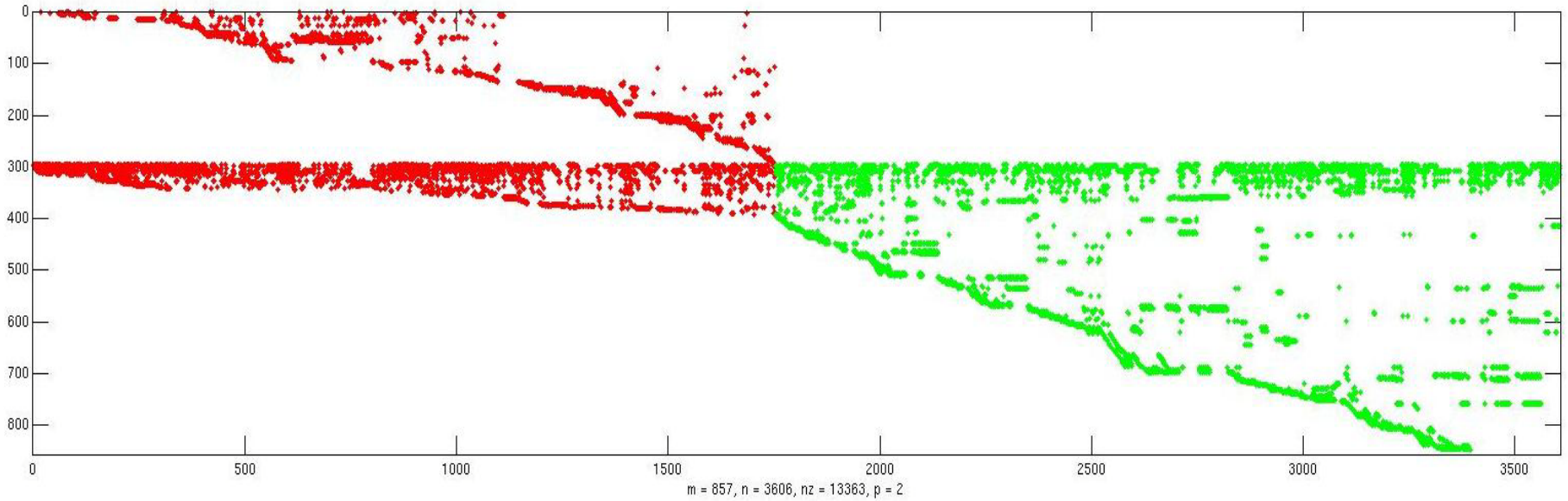


Partitioning the graph

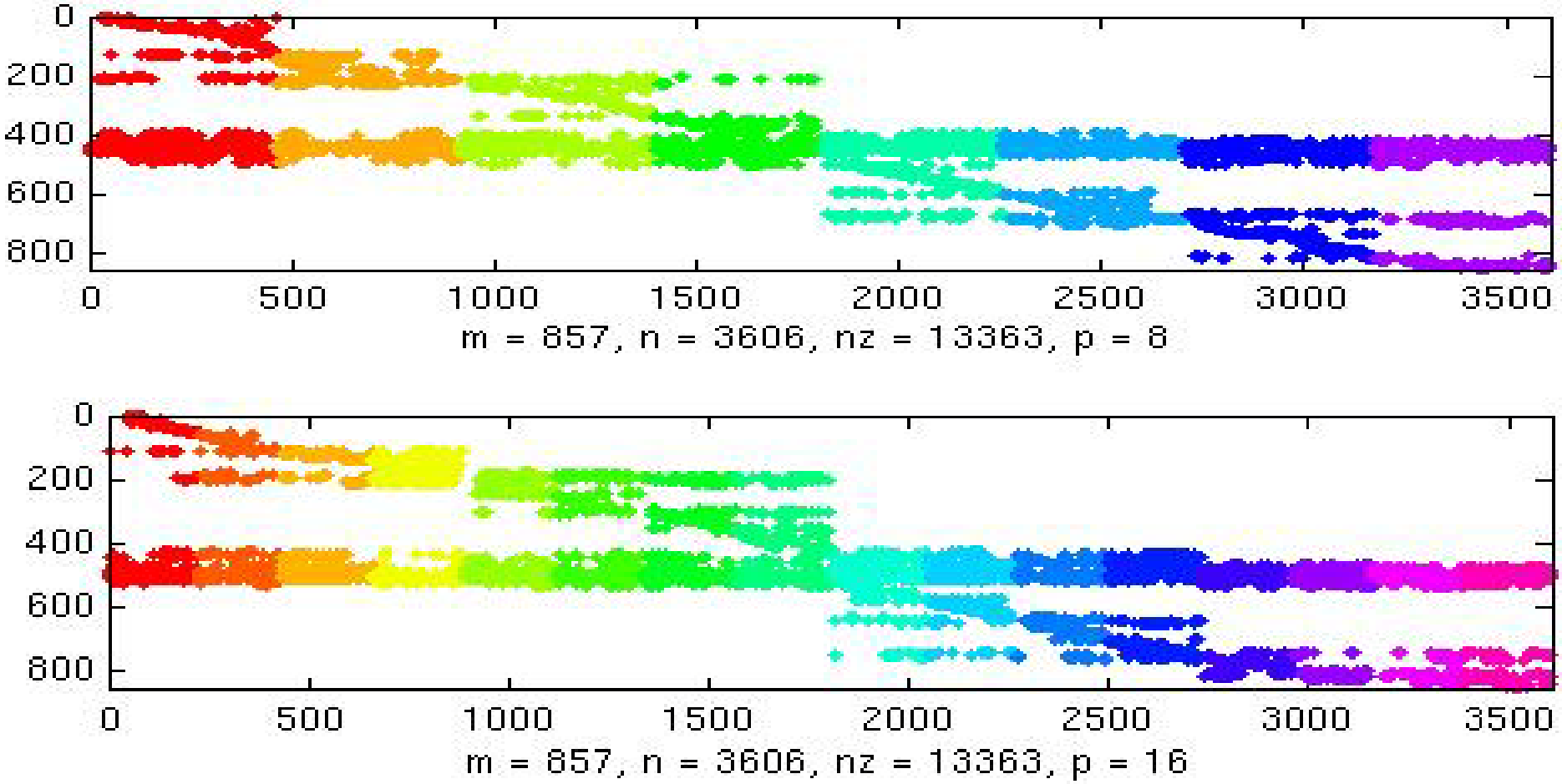
- Graph partitioning software Metis and PaToH were used to partition the bi-partite graph
  - A good quality partition minimizes edge-cut with maximum load balance
  - Reorders the network, without changing the answers
- Edge-cut induces redundant computation or synchronization points
- Partitions should be sized to meet the vector length and memory requirement
  - Large enough to have enough number of threads per thread block
  - Control shared memory requirement to obtain high occupancy
- Need a sufficient number of partitions that can execute concurrently

Partitioning iso-octane chemistry

- LLNL's detailed mechanism for gasoline surrogate composed of 858 species and 3606 reactions

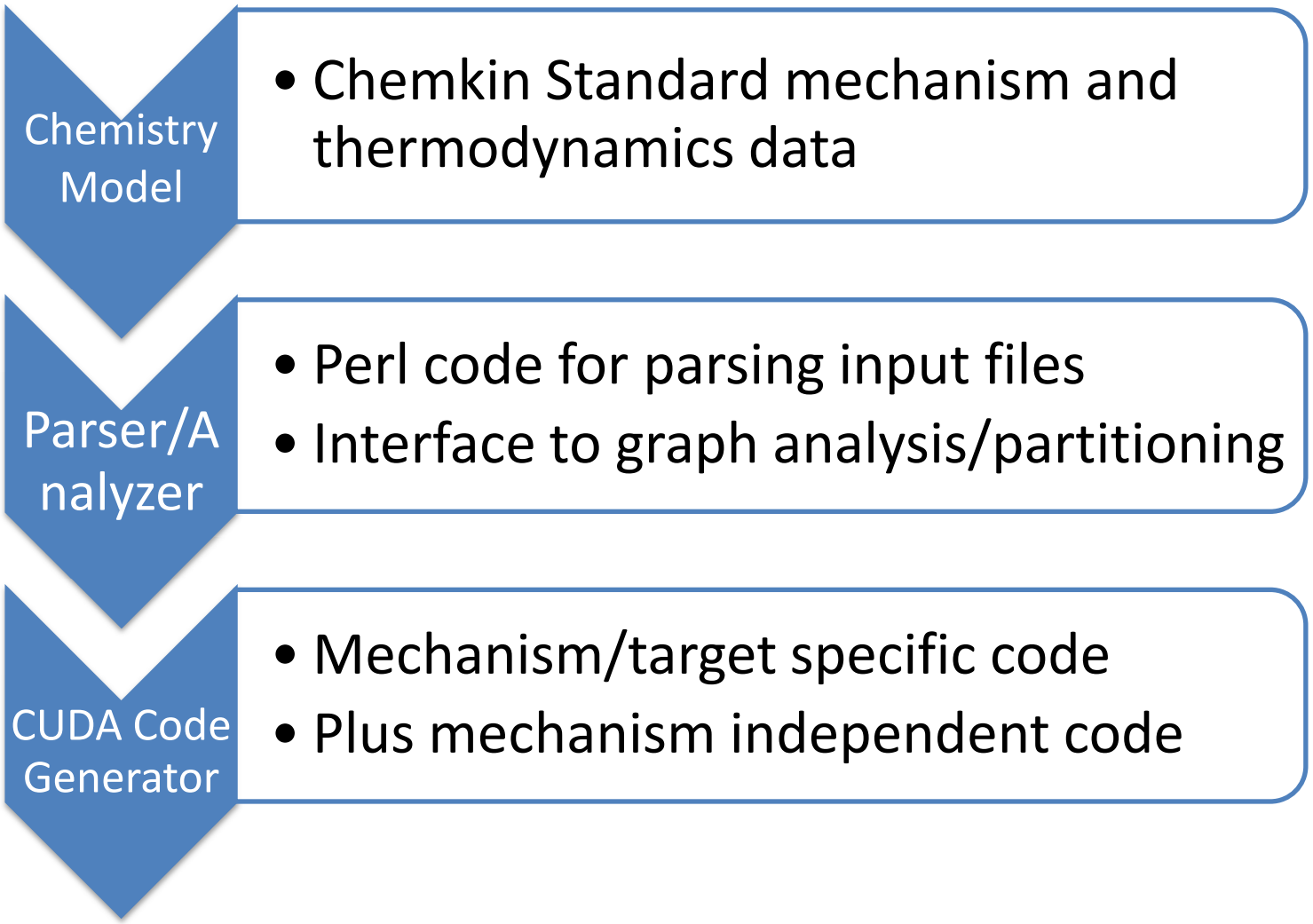


Partitioning iso-octane chemistry (contd)



- The quality of partitioning gets better as the chemistry model gets bigger

Keiki – Code Generator

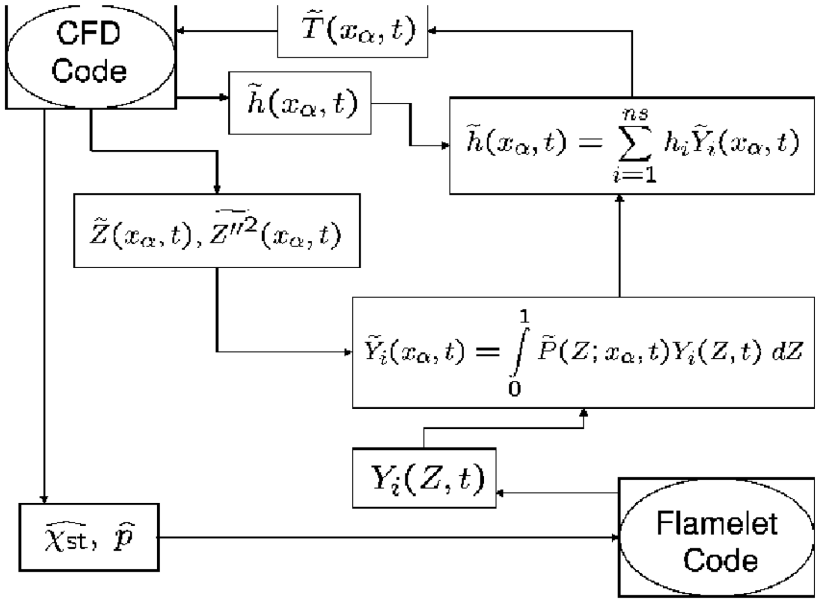


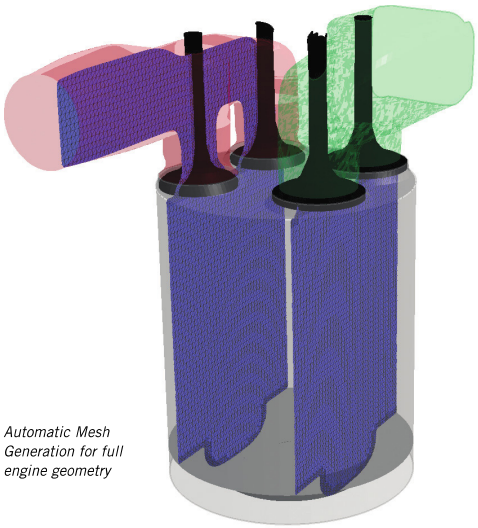
Performance results

- Performance on dual 6-core Opteron CPU and Fermi GPU were compared for 52-species n-heptane and 858 species iso-octane chemistry
  - CPU peak = 2*62.4 = 125 GF
  - GPU peak = 515 GF
- The CPU code was well optimized and tuned for performance
- The execution times on GPU were 3X faster than the CPU
- Work in progress to measure and tune performance on Kepler

GPU library coupled to combustion CFD

- Work in progress:
- A flamelet equation solver is being developed around the CUDA library
- CUDA library for chemical kinetics is being coupled to Forte in partnership with Reaction Design
  - Forte ported to Jaguar (Cray XK6)
  - Software linking and API are being explored





Automatic Mesh Generation for full engine geometry

Summary

- New software and techniques were developed to enable the computation of combustion chemistry on GPU accelerators using the CUDA programming model
- Significant potential to accelerate the computation of very large detailed mechanisms
- What started out as an effort to accelerate S3D has been extended to much larger chemical mechanisms.