CSSI Element: **Fast Dynamic Load Balancing Tools for Extreme Scale Systems**

PI: Mark S. Shephard, Co-PI: Cameron W. Smith, Ph.D. Student: Gerrett Diamond
Scientific Computation Research Center (SCOREC) at Rensselaer Polytechnic Institute, Troy, NY, USA

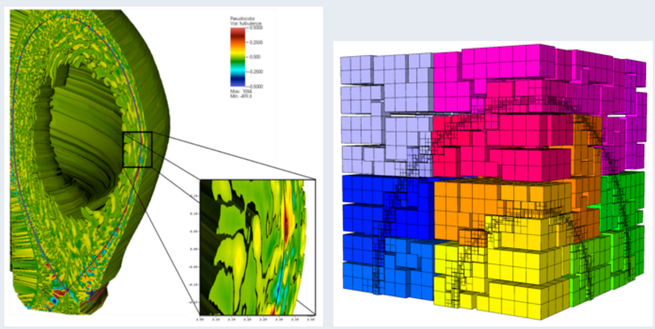
Award #: OAC-1533581

Motivation and Focus

GPUs provide >90% of the compute power on leadership systems.
Simulations with regions of physical interest that change can have

- complex relational structures,
- irregular forms of computational & communication costs, and
- evolving imbalance of work characterized by multiple criteria.

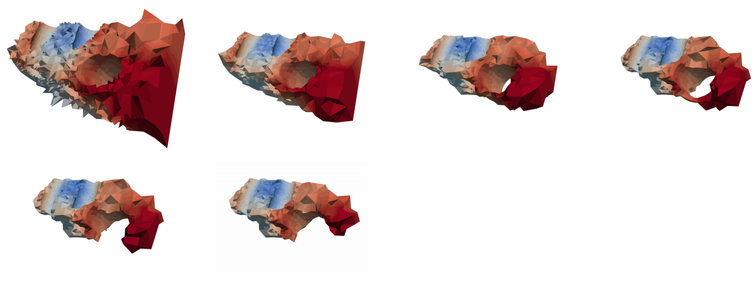
Provide fast dynamic load balancing on GPUs where simulation data exists.



XGC fusion plasma physics (left) and MFEM Laghos Sedov blast (right).

What is EnGPar?

- Provides a diffusive load balancing algorithm for partition improvement and supports multi-criteria partitioning.
- Complements existing multi-level and geometric methods.
- Utilizes a weighted multi(hyper)graph structure to represent data dependencies.
- Implemented to support efficient data parallel operations on accelerators



Multiple diffusive iterations (left to right) biased to migrate entities in order of descending distance (red to blue) from the topological center of the part (blue).

Unstructured Mesh Partition Improvement

Problem setup

- Billion element mesh of vertical tail structure.
- Run on the Mira BG/Q with one process per hardware thread.
- Target imbalance of 1.05.
- The imbalance of a given type (vtx, edge, face, or region) is defined as the max part weight divided by the avg part weight.

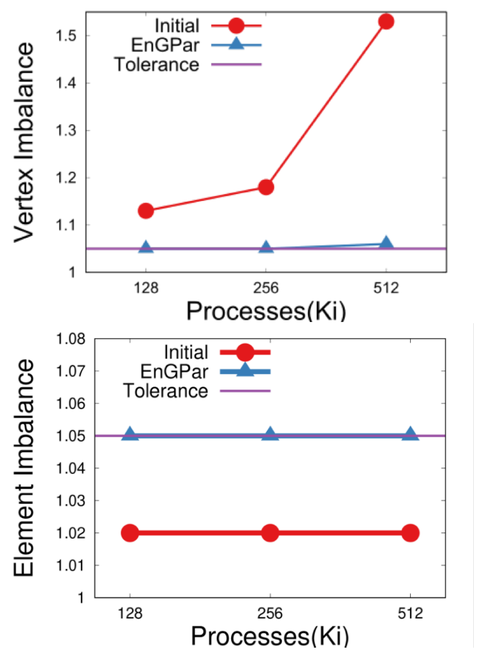
Initial partitions are built using:

- Global ParMETIS part k-way to 8Ki($8 * 2^{10}$) parts.
- Local ParMETIS part k-way from 8Ki to 128Ki, 256Ki, and 512Ki parts.

Element Partition: Mesh Vertex Imbalance Reduction

The partitions before using EnGPar:

Number of Parts	128Ki	256Ki	512Ki
Elements per part	9,836	4,918	2,459
Vertex imbalance	1.13	1.18	1.53
Element imbalance	1.02	1.02	1.02



Mesh vertex imbalances are reduced from 13% to 5% for 128Ki, 18% to 5% for 256Ki, and 53% to 6% for 512Ki. Edge cut is increased by 1%.

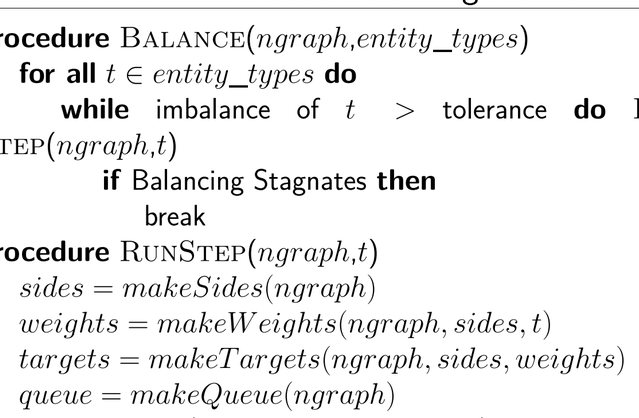
Diffusive Partitioning

Algorithm 1 Diffusive Load Balancing Framework

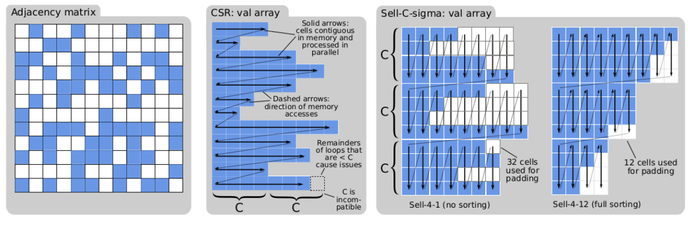
```
1:  procedure BALANCE(ngraph,entity_types)
2:      for all t ∈ entity_types do
3:          while imbalance of t > tolerance do RUNSTEP(ngraph,t)
4:              if Balancing Stagnates then
5:                  break
6:  procedure RUNSTEP(ngraph,t)
7:      sides = makeSides(ngraph)
8:      weights = makeWeights(ngraph, sides, t)
9:      targets = makeTargets(ngraph, sides, weights)
10:     queue = makeQueue(ngraph)
11:     plan = select(ngraph, targets, queue)
12:     ngraph.migrate(plan)
```

Queue

One inward BFS and one outward BFS to compute graph distance; the traversal order.



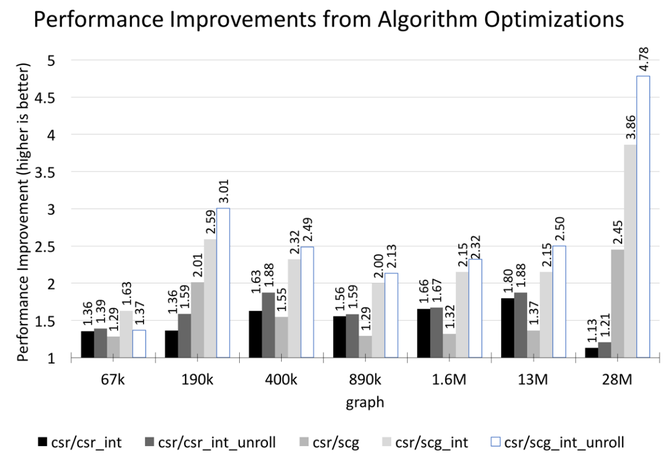
(left) The distance from each vertex to the boundary and (right) the distance from the core vertex (marked with a zero near the bottom left corner).



Sell-C-Sigma structure (Besta and Merending, et al.)

Accelerating BFS

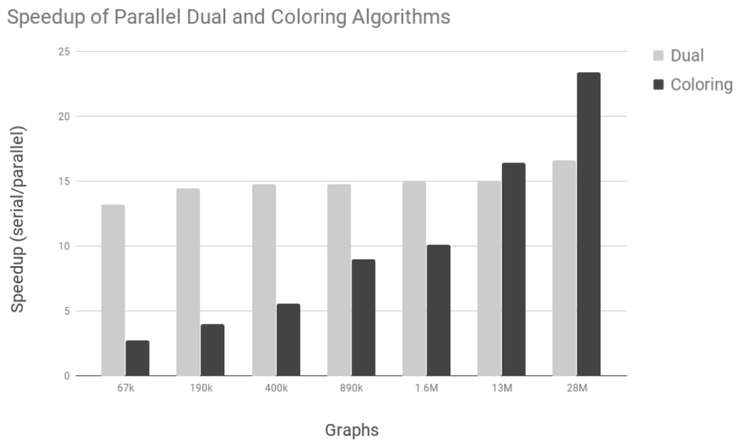
Hypergraphs created from unstructured tet mesh of automotive part
Timing comparison on NVIDIA 1080ti - includes data transfers, but not JIT; average of three runs shown
`scg_int_unroll` is 4.78 times faster than csr on 28M graph and up to 11 times faster than serial push on Intel Xeon (not shown)
Memory coalescing is critical; csr vs. scg



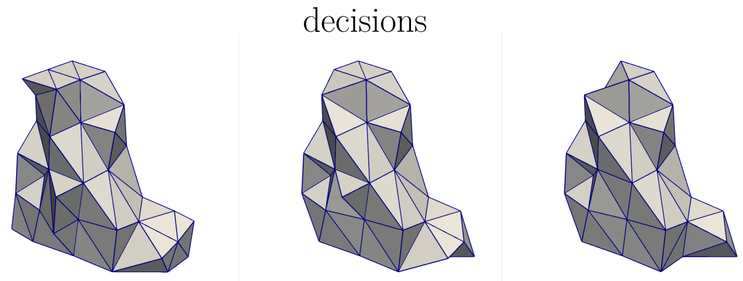
`push`: C++ serial push, `pull`: C++ serial 'pull', `csr`: OpenCL 'pull' on CSR, `scg`: OpenCL 'pull' on Sell-C-Sigma, `*_int`: 4B int, `*_unroll`: unroll the vtx-to-hyperedge loop

Selection using Kokkos

Operate on non-overlapping cavities to avoid race; color the boundary
A cavity is selected for migration if it satisfies color, target, and size criteria



Speedup of parallel vs. serial coloring and dual computation on NVIDIA 1080ti Making good decisions



Initial, GPU Selection, CPU Selection Bias selection towards cavities with highest topological distance.

Closing Remarks and Future Work

- Using Kokkos for improved portability and ease of use over OpenCL
- Porting of BFS to Kokkos is underway
- Focused on balancing particles in XGCm accelerated unstructured mesh PIC

More Info

EnGPar - `github.com/SCOREC/EngPar`
Mark S. Shephard - `shephard@rpi.edu`
Cameron W. Smith - `smithc11@rpi.edu`