



Latency, Bandwidth, and Concurrent Issue Limitations in High-Performance CFD

<http://www.mcs.anl.gov/petsc-fun3d>

William Gropp, Argonne National Laboratory

Dinesh Kaushik, Argonne National Laboratory & ODU

David Keyes, Old Dominion University, LLNL & ICASE

Barry Smith, Argonne National Laboratory

Organization of the Presentation

- Performance Issues
- Background of PETSc-FUN3D
- Optimizing for good memory performance
- Bottlenecks to parallel scalability
- Adapting to the hybrid (MPI/OpenMP) programming model
- Conclusions and future work

Factoring out the Parallel Performance

- Per-processor Performance
 - ⌚ Needs attention to the memory hierarchy
- Implementation Scalability
 - ⌚ Problem constrained scalability
 - ⌚ Memory constrained scalability
- Algorithmic Scalability
 - ⌚ Degrades as the number of processors increase

Three Fundamental Limiting Factors to Peak Performance: Vertical Hierarchy

- **Memory Bandwidth**
 - ⌚ Processor does not get data at the rate it requires
- **Instruction Issue Rate**
 - ⌚ If the loops are load/store bound, we will not be able to do a floating point operation in every cycle even if the operands are available in primary cache
 - ⌚ Several constraints (like primary cache latency, latency of floating point units etc.) are to be observed while coming up with an optimal schedule
- **Fraction of Floating Point Operations**
 - ⌚ Every instruction is not floating point instruction

Memory Performance - A Limitation

- Memory performance improvement rate (7% per year) is far behind the CPU performance growth (about 55% per year)
- The performance of many scientific computing codes is limited by the available memory bandwidth
- In shared memory programming, when the processors on a node compete for the memory bandwidth, there is added motivation to reduce the number of memory transactions (necessary or artificial).
- Memory performance models can be very helpful in understanding the observed performance of a code

Primary PDE Solution Kernels

- **Vertex-based loops**
 - ⌚ state vector and auxiliary vector updates
- **Edge-based “stencil op” loops**
 - ⌚ residual evaluation
 - ⌚ approximate Jacobian evaluation
 - ⌚ Jacobian-vector product (often replaced with matrix-free form, involving residual evaluation)
- **Sparse, narrow-band recurrences**
 - ⌚ approximate factorization and back substitution
- **Vector inner products and norms**
 - ⌚ orthogonalization/conjugation
 - ⌚ convergence progress and stability checks

Features of PETSc-FUN3D

- Based on “legacy” (but contemporary) CFD application with significant F77 code reuse
- Portable, message-passing library-based parallelization, run on desktop Pentium boxes through Tflop/s ASCI platforms
- Simple multithreaded extension (for SMP Clusters)
- Sparse, unstructured data, implying memory indirection with only modest reuse
- Wide applicability to other implicitly discretized multiple-scale PDE workloads - of interagency, interdisciplinary interest
- Extensive profiling has led to follow-on algorithmic research

Enhancing Locality

- Choose data layouts that enhance locality at every level of memory hierarchy
- Storage/use patterns should follow memory hierarchy
 - ⌚ **Blocks for Registers**
 - block storage format for multicomponent systems – cuts the number of **loads**
 - ⌚ **Interlaced Data Structures for Cache**
 - Choose
$$u1, v1, w1, p1, u2, v2, w2, p2, \dots$$
in place of
$$u1, u2, \dots, v1, v2, \dots, w1, w2, \dots, p1, p2, \dots$$
 - Cuts down **TLB and data** cache misses
 - ⌚ **Subdomains for Distributed Memory**
 - “Chunky” domain decomposition for optimal surface-to-volume (communication-to-computation) ratio

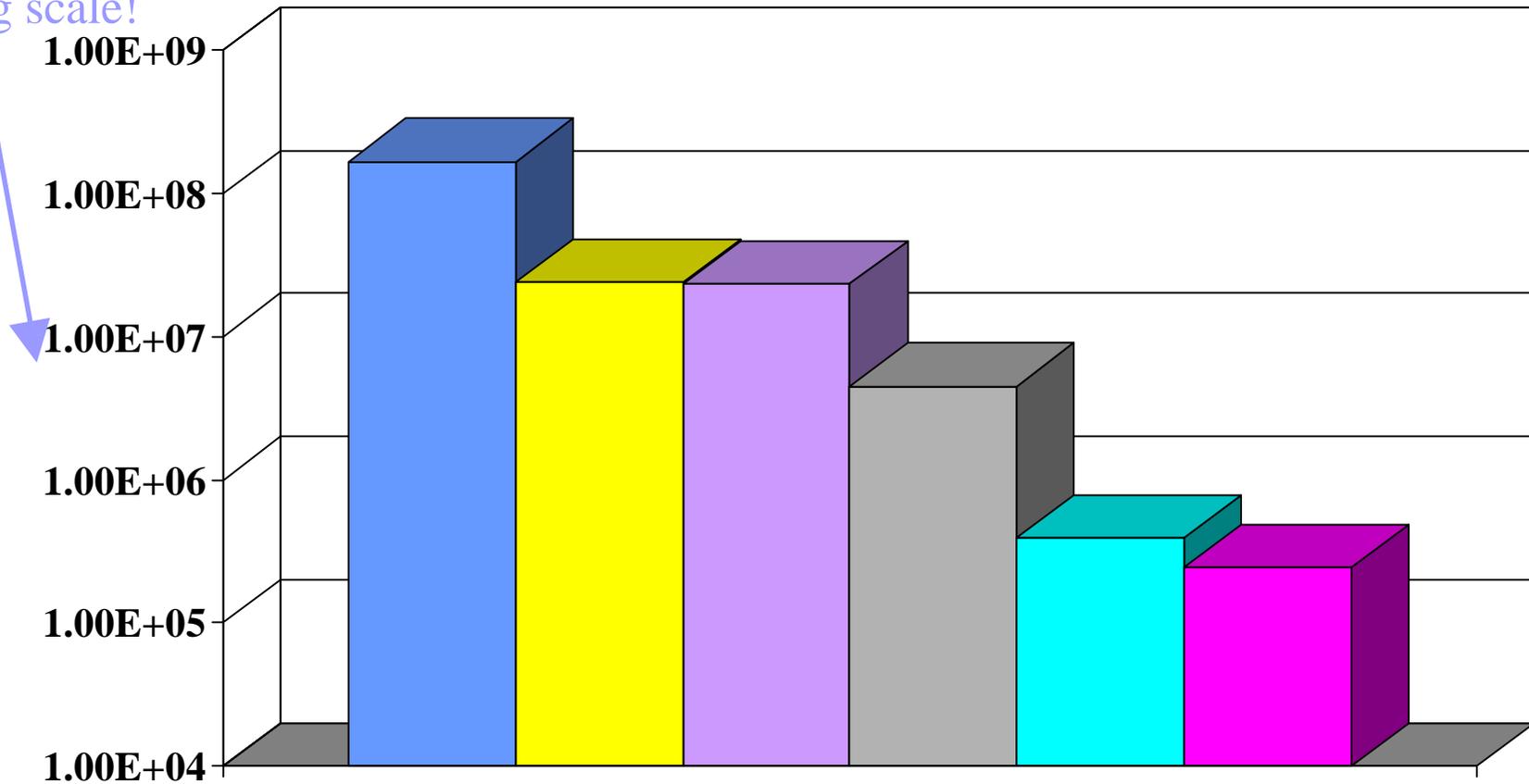
Data Layouts - Reorderings

- Edge Reordering
 - ⌚ sort the nodes at either ends of the edges
 - ⌚ effectively transforms an edge based loop into a node based loop
 - ⌚ enhances temporal locality
- Node Reordering
 - ⌚ Bandwidth reducing orderings will reduce the TLB and cache misses by referring to data items that are close in memory.
 - ⌚ Our experience is with RCM and Sloan

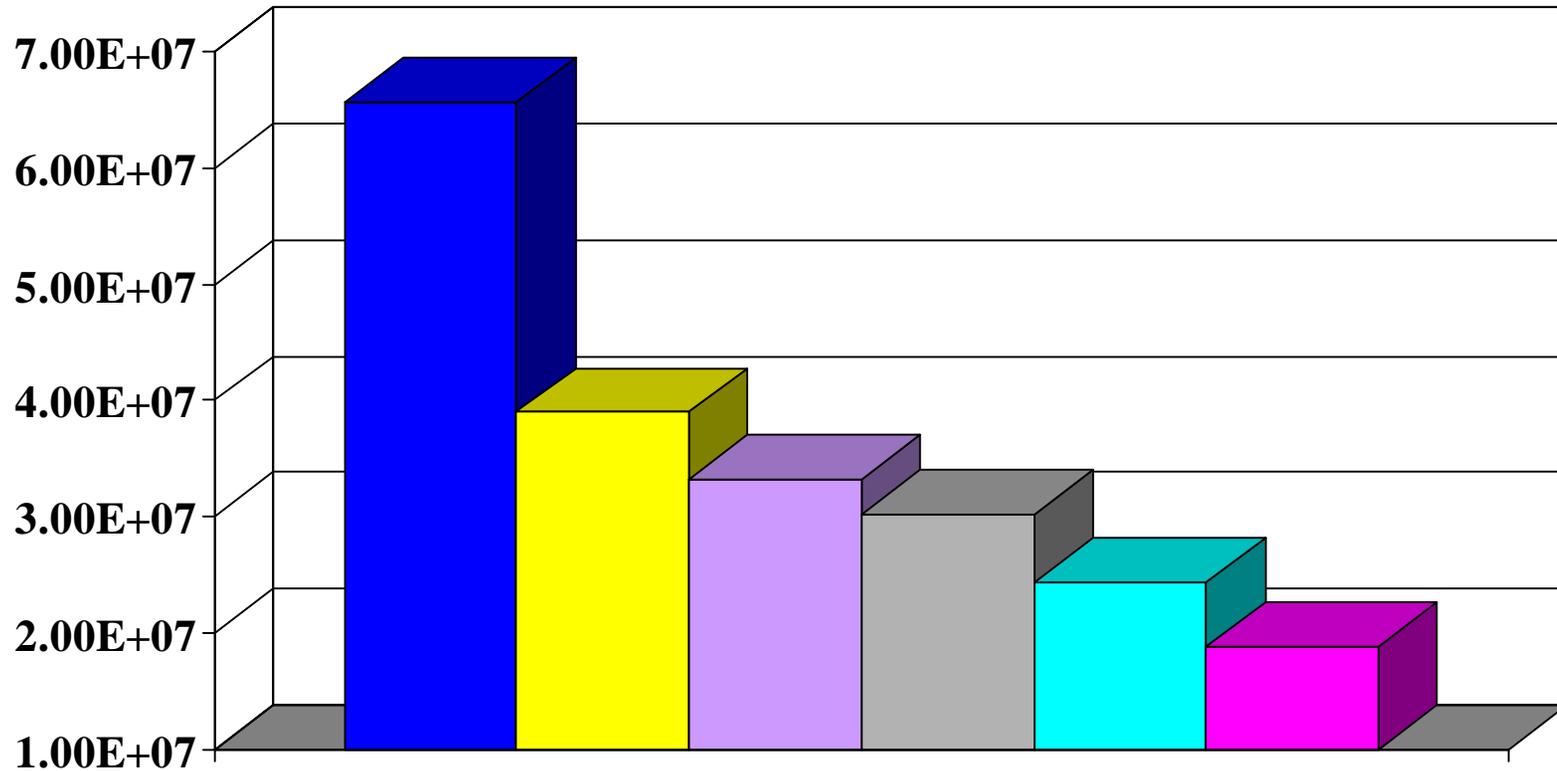
TLB Misses: Measured Values on Origin



Log scale!



Secondary Cache Misses: Measured Values on Origin



MPI: Parallel Performance on ASCI Red

ONERA M6 Wing Test Case, Tetrahedral grid of 2.8 million vertices (about 11 million unknowns) on up to 3072 ASCI Red Nodes (each with dual Pentium Pro 333 MHz processors)

Nodes	Iter	Time in seconds	Speedup	Parallel Efficiency		
				Overall	Algorithmic	Implementation
128	22	2,039	1.00	1.00	1.00	
512	26	638	3.20	0.80	0.85	0.94
1024	29	362	5.63	0.70	0.76	0.93
2048	32	208	9.78	0.61	0.69	0.89
3072	34	159	12.81	0.53	0.65	0.82

MPI: Scalability Bottlenecks on ASCI Red

Nodes	Percentage Times for			Scatter Scalability	
	Global Reduc-tions	Implicit Synchro-nizations	Ghost Point Scatters	Total Data Sent per Iteration (GB)	Application Level Effective Bandwidth per Node (MB/s)
128	5	4	3	3.6	6.9
512	3	7	5	7.1	6.0
1024	3	10	6	9.4	7.5
2048	3	11	8	11.7	5.7
3072	5	14	10	14.2	4.6

Motivation for Hybrid Model

- **Given**
 - ⌚ a scalable MPI based code
- **Goal**
 - ⌚ use hybrid model to achieve better performance than MPI alone
- **Methodology:**
 - ⌚ assign one subdomain to one MPI process
 - ⌚ use OpenMP within a subdomain that gets mapped to a node (with 2 or more processors)
- **Advantage**
 - ⌚ take advantage of shared memory programming within a subdomain
 - ⌚ results in bigger subdomains as more than one thread can work on a subdomain as compared to pure MPI case

Our View of the Hybrid Model

- **MPI Extreme**
 - ⌚ the user manages the memory updates
- **OpenMP Extreme**
 - ⌚ the system manages the memory updates
- **Hybrid MPI/OpenMP**
 - ⌚ Some memory updates are managed by the user and the rest by the system

Performance Issues for OpenMP

- Overhead of thread management
- Redundant storage and work
- Sequential reduction phase, which tend to be memory bandwidth bound
- Simplicity goes away when user takes care of memory updates (similar to MPI model)

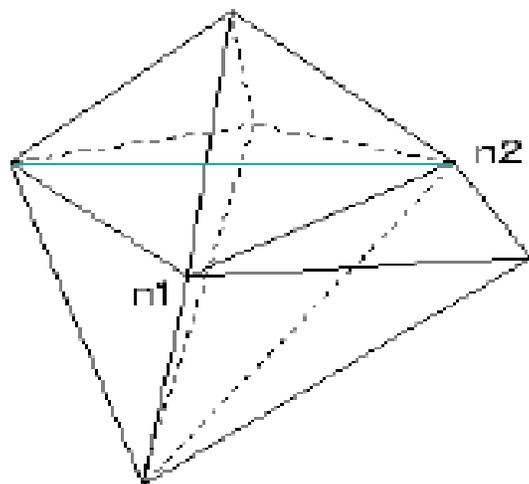
Hybrid Model: Implementation Issues

- Data Distribution
 - ⌚ False sharing
 - ⌚ Cache locality
- Work Division
 - ⌚ Compiler or User
 - ⌚ Static or dynamic
- Updates of the Shared Data
 - ⌚ Private data but initialization and reductions are memory bandwidth bound
 - ⌚ Shared data but updates need to be synchronized

Hybrid Model: Three Implementation Strategies

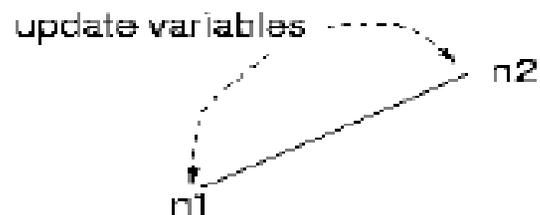
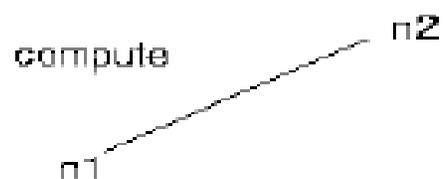
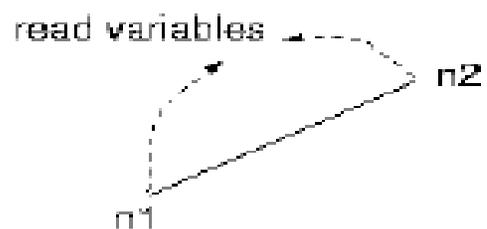
- Edge Coloring
 - ⌚ Poor cache locality
 - ⌚ Compiler divides the work
 - ⌚ Updates are independent
- Edge Reordering
 - ⌚ Excellent cache locality
 - ⌚ Compiler divides work
 - ⌚ Updates are a problem
- Manual Work Division
 - ⌚ Each MPI process calls MeTiS to further subdivide the work among threads
 - ⌚ Boundary data is replicated for each thread
 - ⌚ “Owner computes” rule is applied for every thread

Flux Evaluation in PETSc-FUN3D



Variables at each node:
density,
momentum (x,y,z),
energy,
pressure

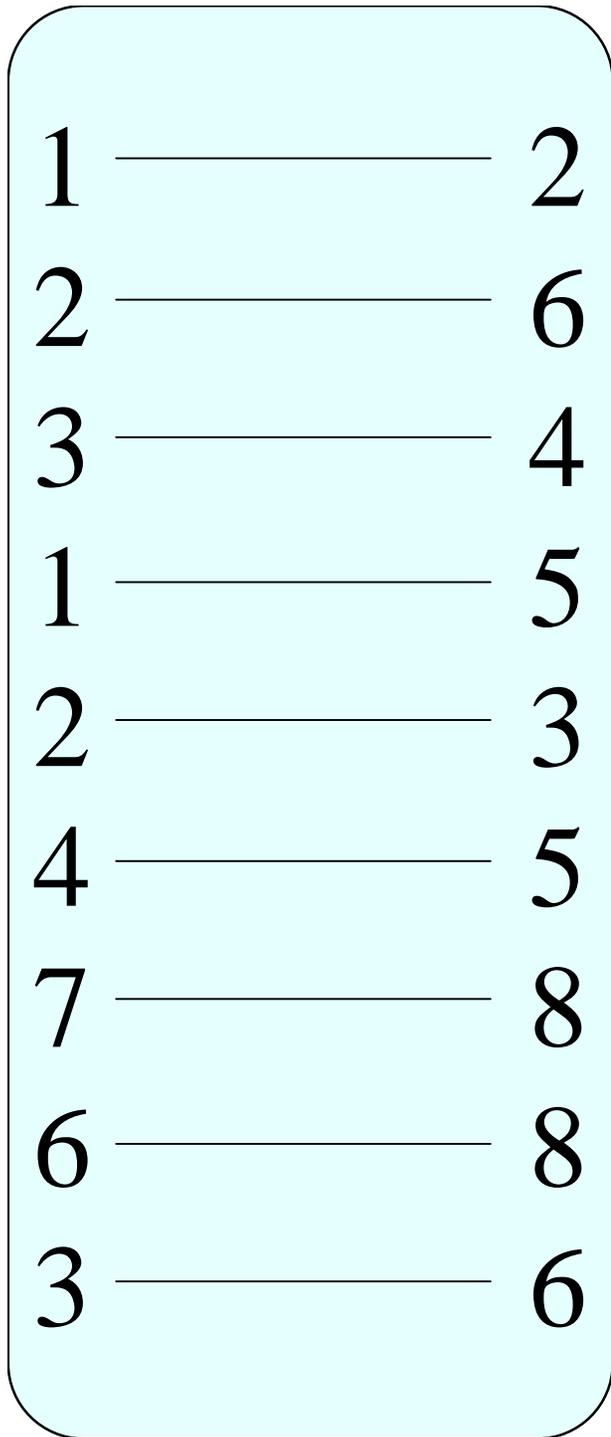
Variables at edge:
identity of nodes,
orientation(x,y,z)



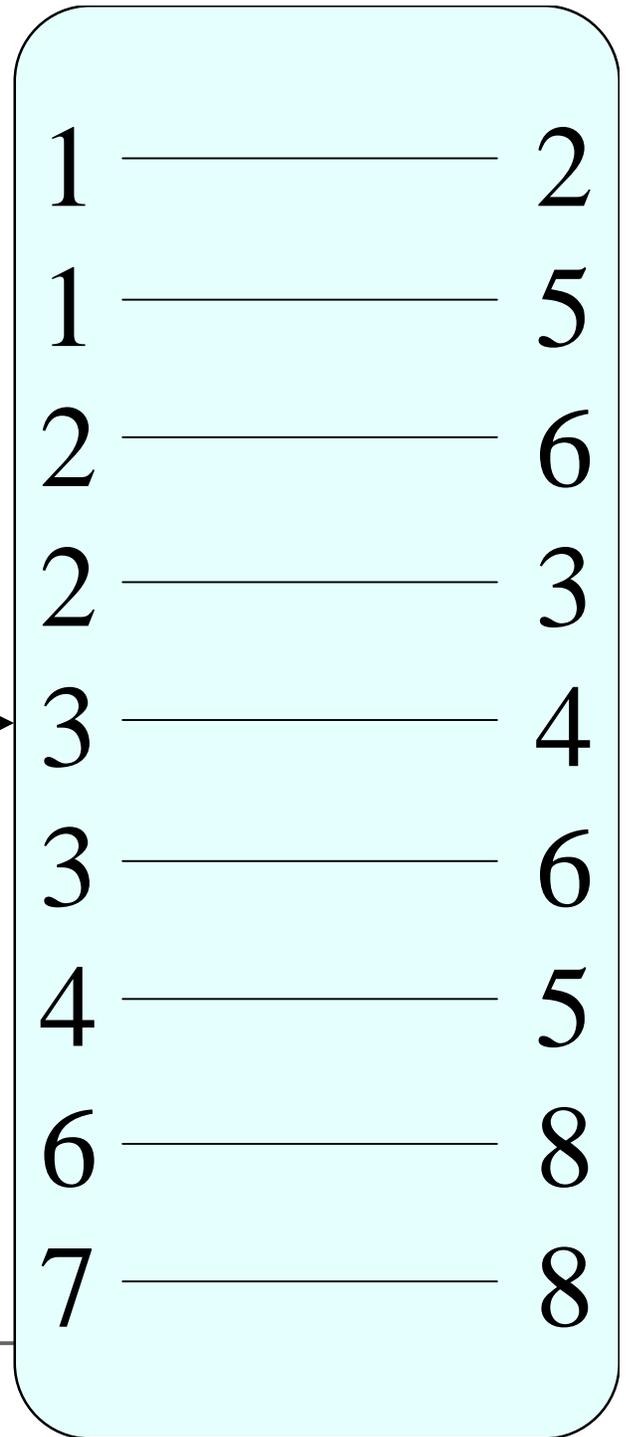
1	2
2	6
3	4
1	5
2	3
4	5
7	8
6	8
3	6

Edge Coloring

1	2
3	4
7	8
2	6
1	5
2	3
6	8
4	5
3	6



Edge Reordering



MPI/OpenMP in PETSc-FUN3D

- Only in the flux evaluation phase, as it is not memory bandwidth bound
- Gives the best execution time as the number of nodes increases because the subdomains are chunkier as compared to pure MPI case

Nodes	MPI Processes Per Node		MPI/OpenMP 2 Threads Per Node		
	1	2	Edge Coloring	Edge Reordering	MeTiS Divided
256	510	332	423	314	293
1024	183	136	130	116	109
3072	93	91	63	62	63

Conclusions

- PDE codes can run well on distributed hierarchical memory machines, with attention to partitioning, vertex ordering, component ordering, blocking, and tuning
- Parallel scalability is not hard, but attaining high per-processor performance for sparse problems gets more challenging with each machine generation
- Hybrid MPI/OpenMP achieves good overall performance but should be used only in the phases that are not memory bandwidth limited
 - ⌚ Results in bigger subdomains
 - Faster convergence rate
 - Less network transactions

Acknowledgments

- Accelerated Strategic Computing Initiative, DOE
 - ⌚ *access to ASCI Red and Blue machines*
- National Energy Research Scientific Computing Center (NERSC), DOE
 - ⌚ *access to large T3E*
- SGI-Cray
 - ⌚ *access to large T3E*
- National Science Foundation
 - ⌚ *research sponsorship under Multidisciplinary Computing Challenges Program*
- U. S. Department of Education
 - ⌚ *graduate fellowship support for D. Kaushik*

Related URLs

- Follow-up on this talk
<http://www.mcs.anl.gov/petsc-fun3d>
- PETSc
<http://www.mcs.anl.gov/petsc>
- FUN3D
<http://fmad-www.larc.nasa.gov/~wanderso/Fun>
- ASCI platforms
<http://www.llnl.gov/asci/platforms>
- International Conferences on Domain Decomposition Methods
<http://www.ddm.org>
- International Conferences on Parallel CFD
<http://www.parcfd.org>