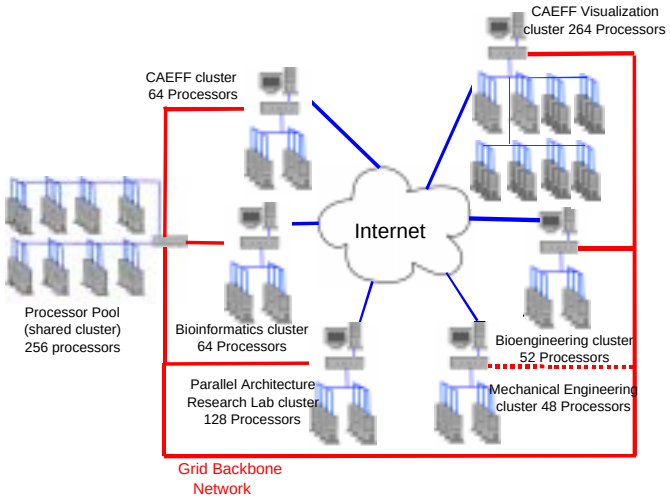
META-SCHEDULING FOR MINI-GRIDS

Dan Stanzione and Walt Ligon
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Computational Mini-Grids

A computational mini-grid is a campus-wide collection of high-performance computing resources inter-connected via dedicated network links. Mini-grids allow for the exchange of computing power between the various computer systems which comprise the grid. Analogous to power grids, computational grids allow users a local access point with a single authentication which can reach all the resources of the grid.

Clemson Computational Mini-Grid



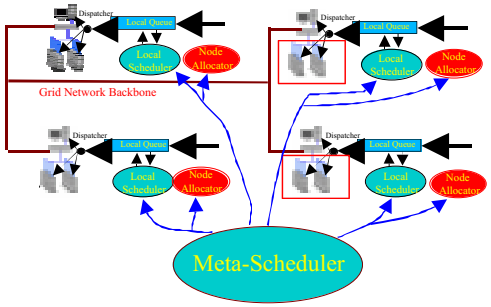
What Is Meta-Scheduling?

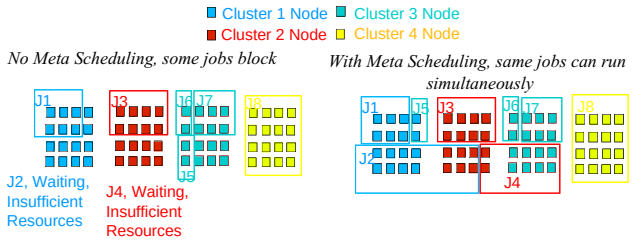
Each compute cluster which comprises the grid can accept jobs from users. Each cluster has its own local queue, scheduler, scheduling policy, and resource allocator, which decide in which order jobs will run. Meta-scheduling is a higher-level distributed scheduling process, through which the actions of all the local schedulers are coordinated on a grid-wide basis.

Why Use Meta-Scheduling?

- Greater total system throughput (Idle nodes in one cluster can be used for another cluster's workload)
- Lower average turnaround time to users
- Cluster resources can be aggregated to run jobs too large for any single cluster

Meta-Scheduler Relationship to Cluster System Software





Research Approach

- Meta-scheduling research involves changes to system software grid-wide. To minimize the impact of algorithm development on daily grid operation, a cluster/grid simulator has been developed.
- Node allocation software has been developed and deployed throughout grid.
- Resource management software has been deployed on each grid cluster.
- Meta-scheduling will be integrated with node allocator and grid management later this year.

Intercluster Node Allocation

Balloc, the Beowulf node allocator, has been developed to provide a node loaning/sharing capability. Balloc allows any compute node in any cluster on the grid to "move" from one cluster to another. With Balloc in place, clusters can be made to virtually grow to encompass all the processors in the grid. Balloc is a distributed set of daemons that run on all grid clusters, and communicate among each other to satisfy requests for idle nodes. Nodes can be allocated in a shared mode, or exclusively to a single user or job. Balloc is in operation today, and has been used to run jobs of up to 512 processors. Currently, node loaning with Balloc is scheduled manually (by the system administrator). Several large, exclusive allocations have been made to meet Center deadlines.

BeoSim – Beowulf Simulator

- Beosim is an event-driven simulator of Beowulf cluster and mini-grid performance. Beosim is functional and the current testbed for meta-scheduling algorithms. Beosim features include:
  - Fully configurable cluster/grid topology
  - Ability to generate random workloads or simulate from trace files of parallel jobs
  - Modular construction allows easy replacement of:
    - * Local cluster scheduling algorithm
    - * Meta-scheduling algorithm
    - * Node allocation algorithms
    - * Node performance models

Simulation Results

- Single cluster vs. analytic model
  -Verify simulator output against mathematically tractable case, one cluster, FCFS scheduling, Poisson distributed workload, 100,000 jobs

			Model			Simulator		
Arrival Rate λ	Service Rate m	# of Processors *m*	Mean Wait Time W	Mean Service Time T	Processor U	W	T	U
0.004	0.0067	1	225.00	375.00	.60	226.45	376.03	.60
0.004	0.0067	2	14.84	164.84	.30	14.99	164.57	.30
0.004	0.0067	3	1.54	151.54	.20	1.64	151.21	.20
0.005	0.0059	1	963.30	1133.30	.850	1010.50	1180.10	.852
0.005	0.0059	2	37.50	207.50	.425	38.14	207.70	.426
0.005	0.0059	3	4.80	174.80	.283	4.98	174.60	.284

- Multi-cluster with and without meta-scheduling

	Meta-Scheduling off Average Wait Time (sec)	Meta-Scheduling on Average Wait Time (sec)
Cluster 1 (1000 Jobs)	374,003	93,711
Cluster 2 (1000 Jobs)	374,009	280,440
Cluster 3 (1000 Jobs)	374,005	467,197
Cluster 4 (5000 Jobs)	1,873,060	1,029,170
Total Execution Time (sec)	3,750,000	1,500,000

- Meta-scheduling reduces overall execution time by 60% (for this unbalanced load)

Analysis and Conclusions

- Meta-scheduling can have significant impact on throughput and job turnaround time
- More sophisticated algorithm design with scheduler proceeding
- Bring meta-scheduling online with current resource managers

Acknowledgements: This work was supported by the ERC Program of the National Science Foundation under Award Number EEC-9731680.

CAEFF