The Impact of Phase Structure on Performance

Prof. Steven S. Lumetta

University of Illinois at Urbana-Champaign
Dept. of Electrical and Computer Engineering
Coordinated Science Laboratory

Architectures of the Future

- hierarchical organization
  - SMP base units
  - flexible interconnection
- cluster software
  - multiple peer operating systems
  - dynamic resource partitions
- hierarchical structure not well-defined
  - purchased/upgraded incrementally
  - execute on partitions of full machine

Parallel Applications?

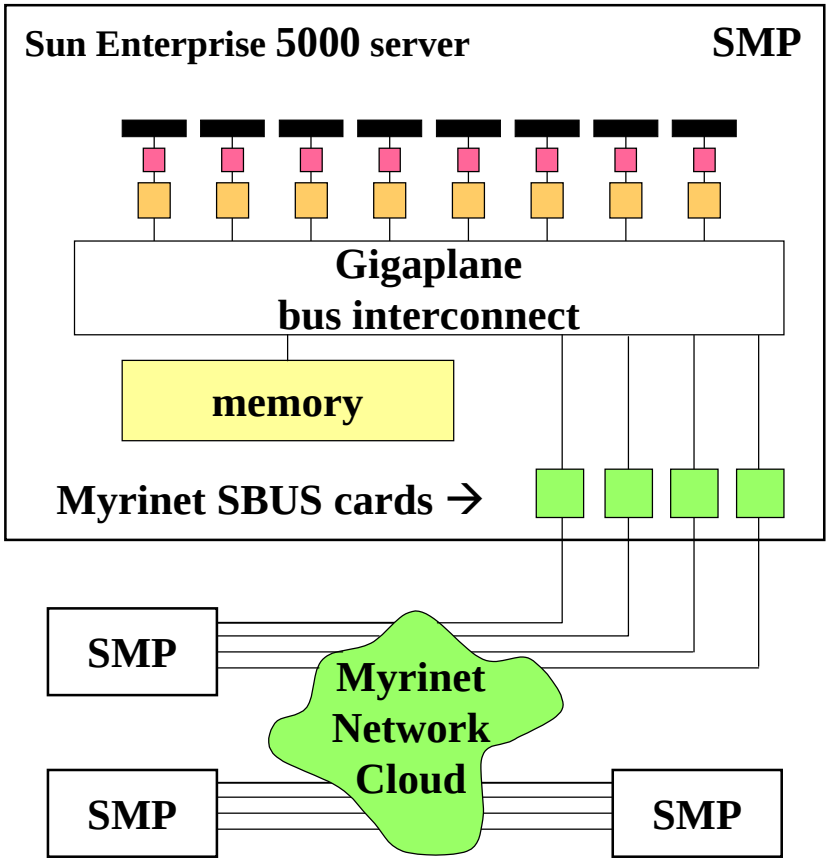
- important commercial applications
  - databases
  - internet services
  - collaborative and virtual environments
- environment suitable for parallelism
  - coarse-grained straightforward
  - communicate with single abstraction for fine-grained
    - uniform message-passing interface (my thesis)
    - distributed shared memory
- how does programming model interact with performance?

Outline

- motivation and question
- performance results
- phase-structured programming
- drawbacks of global structure
  - statistical fluctuations
  - load balance tuning
  - correlated communication
- conclusions

Hardware: A Sun Enterprise Clump

- four Sun Enterprise 5000 servers
- eight 167 MHz UltraSPARC's (four used in following performance data)
- 2 GB of memory
- four Myrinet SBUS network links



Application Run Profile

phase-structured Split-C applications

- SAMPLE: sample sort
  - all-to-all, fine-grained
- CON/comp: connected components, computation-bound input
  - localized, fine-grained
- CON/comm: communication-bound input
  - some all-to-one
- 3D-FFT: Fast Fourier Transform in 3D
  - all-to-all, bulk

Application Performance on 16 Processors

execution time in seconds	Enterprise 5000 Clump (4x4)	16-way UltraSPARC 170 NOW
SAMPLE	1.76 sec	2.01 sec
CON/comp	1.128 sec	1.110 sec
CON/comm	4.18 sec	4.76 sec
3-D FFT	0.504 sec	0.518 sec

- one network interface per processor
- roughly equivalent results
- faster for all-to-one communication (e.g., CON/comm)
  - 1/4 travels through shared memory

Phase-Structured Programming

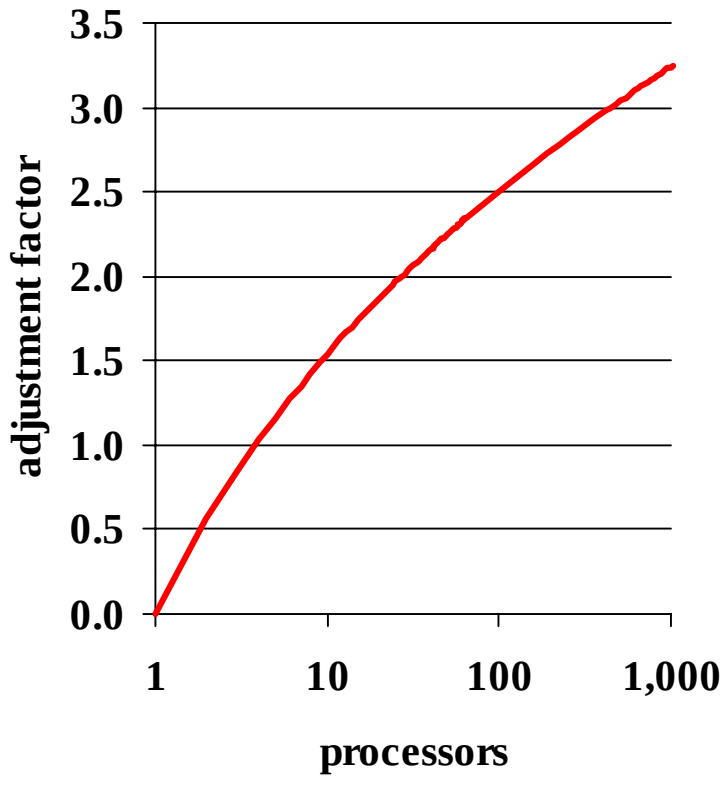
- common approach for parallel applications
  - split into phases
  - global synchronization between each phase
  - generalize to several processor subgroups
- benefits of phase structure
  - global structure similar to sequential program
  - parallel analysis/debugging one phase at a time
- the price of the abstraction
  - wait for slowest processor to finish
  - requires good load balance

Drawbacks of Global Structure

- statistical fluctuations
  - more important on large machines
  - limits expected efficiency
- load balance tuned to multi-level hierarchy
  - uneven processor communication rates
  - change in hierarchy breaks balance
- correlated communication
  - want economy of scale
  - aggregate demands as variable as individual

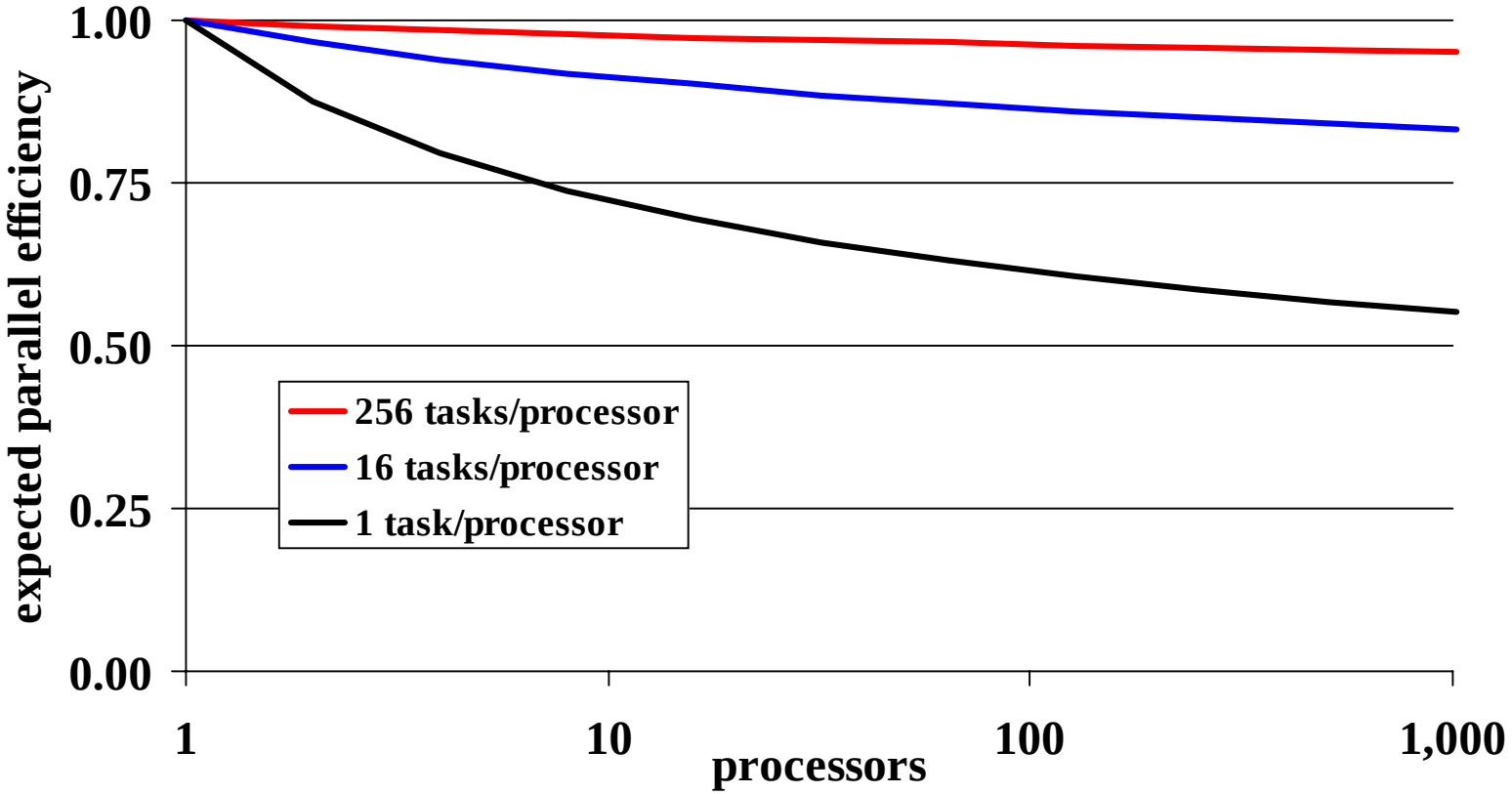
Statistical Fluctuations

- consider
  - P-processor system
  - processor workloads are random variables
- execution time is maximum of P variables
- adjust mean by amount linear in standard deviation (value from graph to right)



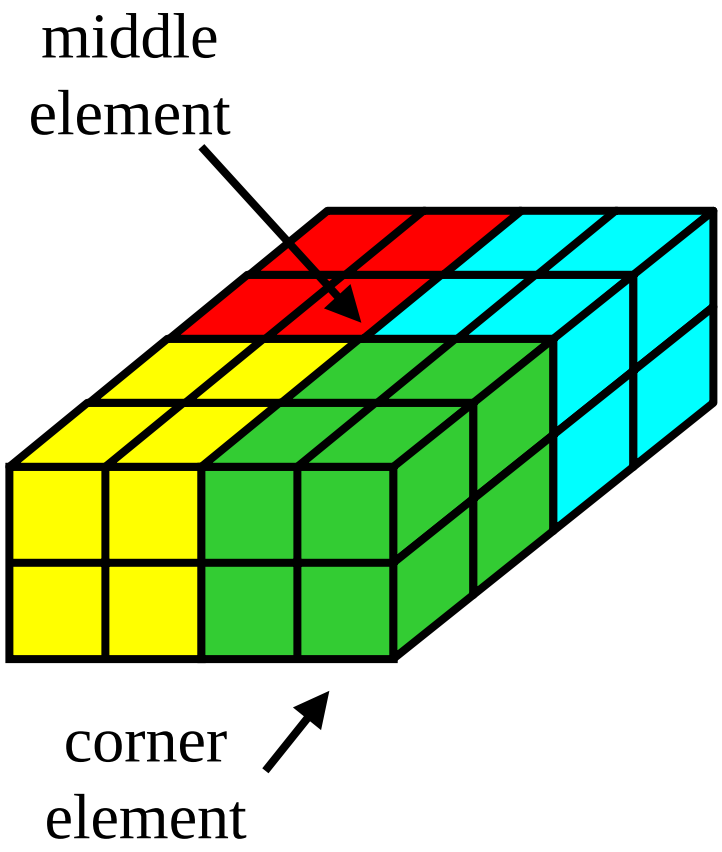
Expected Parallel Efficiency

per-task variability is 0.25



Uneven Communication Demands

- EM3D electromagnetic wave propagation
- physical 3D mesh partitioned between processors
- “corner” elements do not use network
- “middle” elements use network for 40% of communication



Tuned Load Balance

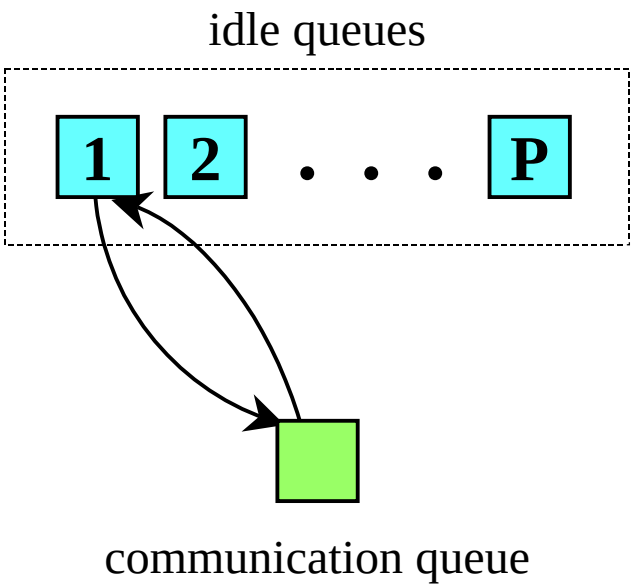
- tuning load balance
  - address hierarchy by repartioning workload
  - balance breaks when hierarchy changes
- measured ratio between fastest and slowest processors in EM3D time step
  - 1.74 on 32-node network of workstations
  - 7.84 on 32-processor Clump (4x8-processors)
  - load balance is very poor on Clump!
- a related problem: subpartitions of hierarchy

Correlated Communication

- want economy of scale
  - processors aggregated in hierarchy (*e.g.*, SMP's)
  - want to provide fewer resources per-processor
  - but are aggregate demands smoother?
  - only if independent
- application breaks into phases
  - computation phase
  - communication phase
- communication demands correlated!

Shared Resource Model

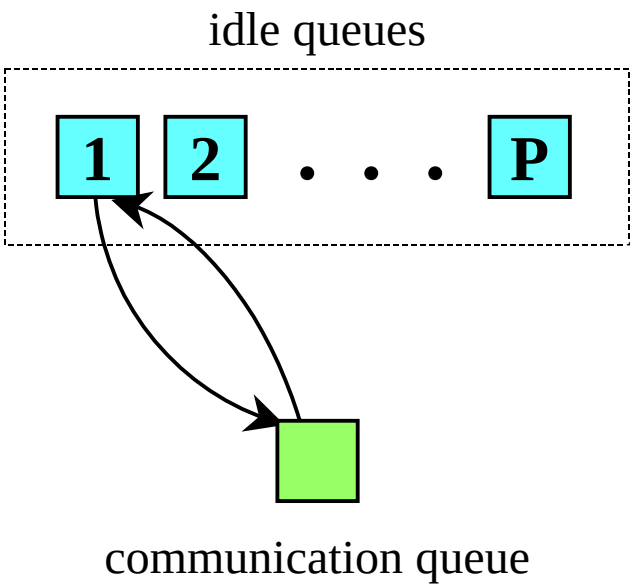
- consider aggregation of P processors with shared communication resources
- compare with unaggregated system
- model: processors alternate between two queues
  - private idle queue
  - shared communication queue



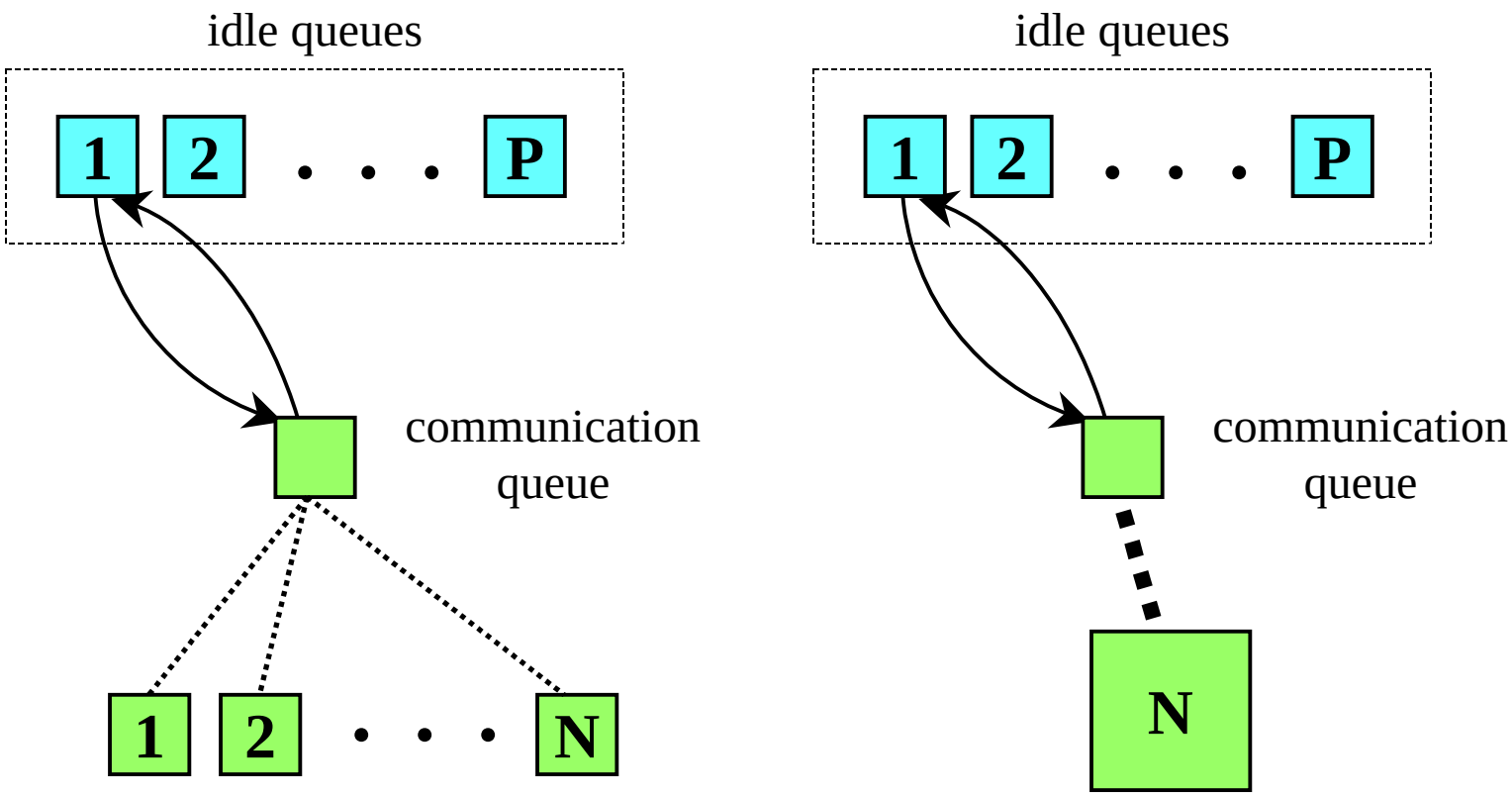
Shared Resource Model

- communication queue
  - single server
  - server-sharing discipline

- processor characterization
  - utilization **u** (from 0 to 1)
  - duty cycle when P=1



Communication Queue Scaling

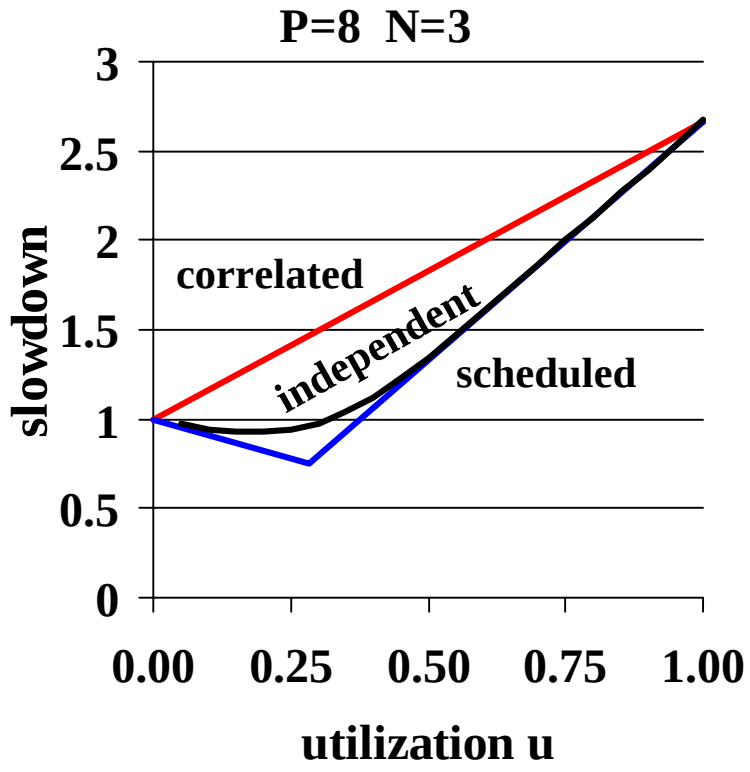


many small resources

one large resource

Application Slowdown Metric

- three regimes
  - correlated: worst case
  - independent: speedup at low utilization
  - scheduled: maximum benefit



Alternatives to Global Structure

- dynamic load-balancing
  - use function-level parallelism?
  - meshes well with object-based approach
  - reduces impact of drawbacks

- tradeoff
  - overhead for dynamic execution
  - non-determinism makes debugging harder

Conclusions

- future hardware is hierarchical
  - can engineer high-performance communication layer
  - can use prediction, *etc*. to hide latency
- performance limited by programming model
  - statistical effects
  - load balance more uneven on Clump
  - phase structure correlates communication
- dynamic approaches: help or hindrance?