Parallel Processors: Clusters, Grid Computing, Network, Performance

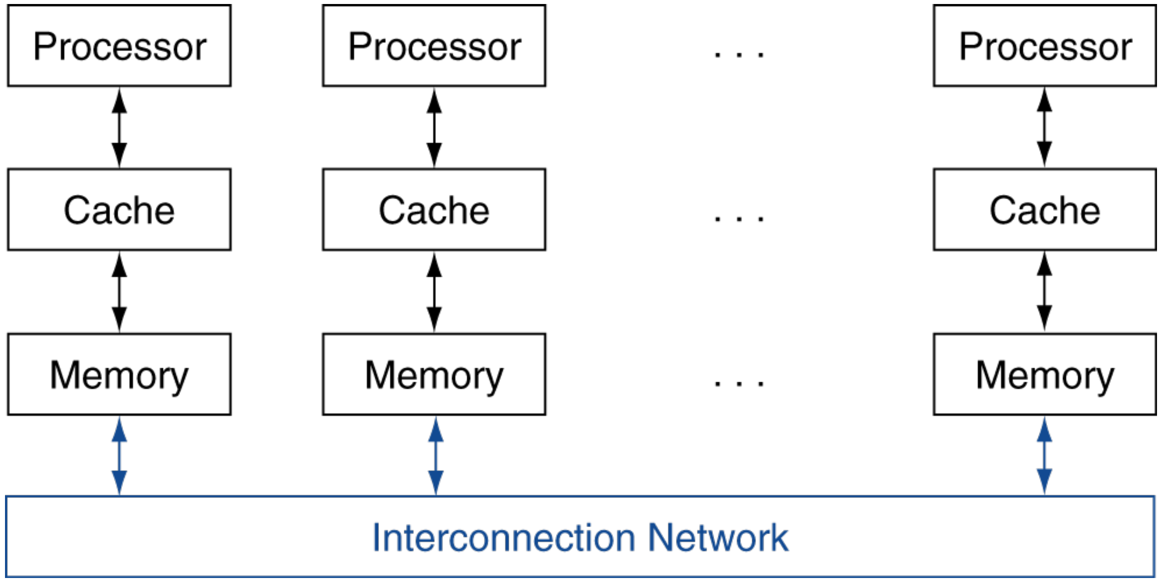
Dr. Vincent C. Emeakaroha

29-03-2017

vc.emeakaroha@cs.ucc.ie

Message Passing Multiprocessors

- Each processor has private physical address space
- Hardware sends/receives messages between processors



Loosely Coupled Clusters

- Network of independent computers
  - Each has private memory and OS
  - Connected using I/O system
    - E.g., Ethernet/switch, Internet
- Suitable for applications with independent tasks
  - Web servers, databases, simulations, …
- High availability, scalable, affordable
- Problems
  - Administration cost (prefer virtual machines)
  - Low interconnect bandwidth
    - c.f. processor/memory bandwidth on an SMP

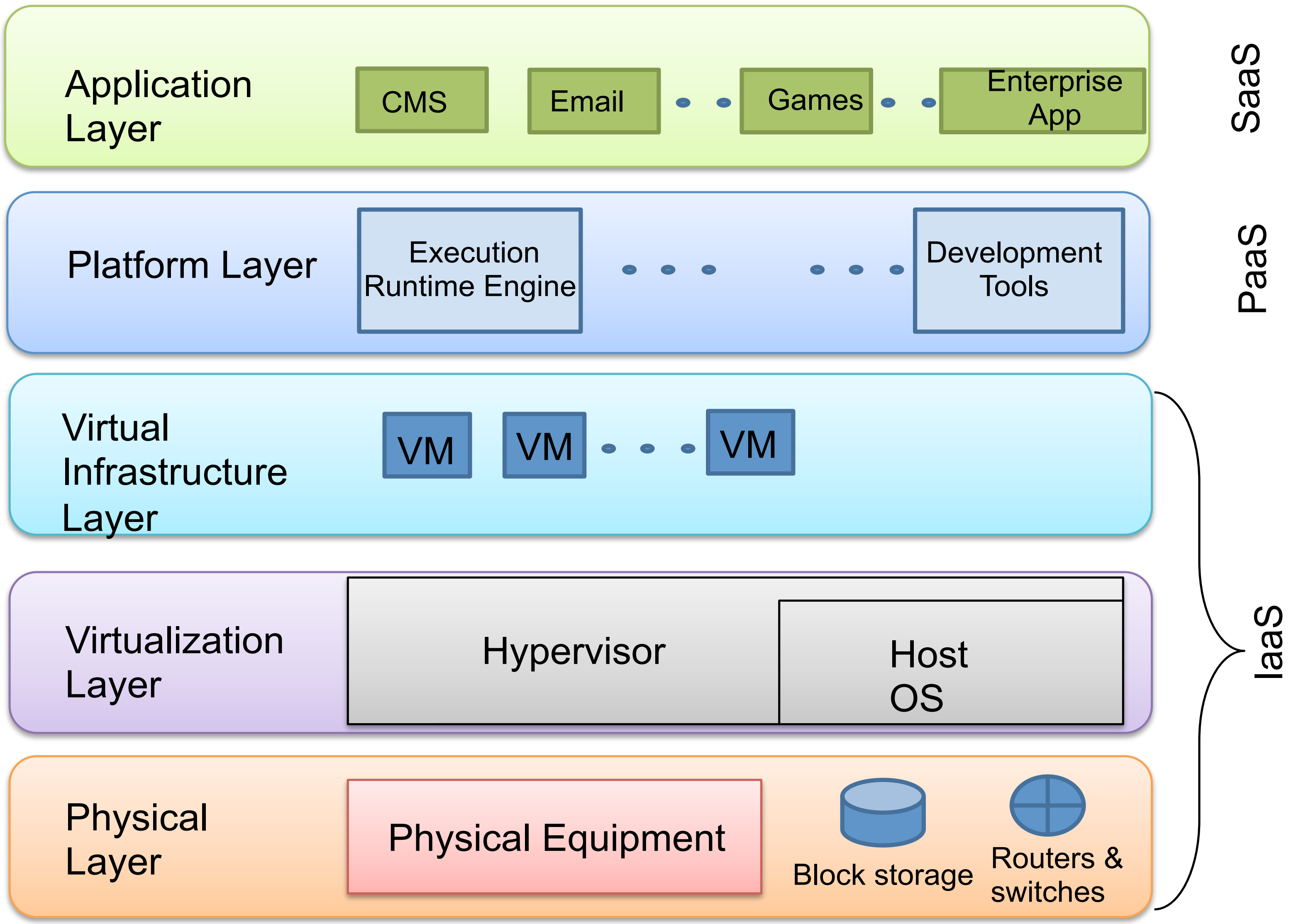
Grid Computing

- Separate computers interconnected by long-haul networks
  - E.g., Internet connections
  - Work units farmed out, results sent back
- Can make use of idle time on PCs
  - E.g., SETI@home, World Community Grid
  - Over 5 million computer users in more than 200 countries

Cloud Computing

- **Cloud Computing** is a general term used to describe a new class of network based computing that takes place over the Internet,
  - Basically storing, processing and accessing data over internet
  - Uses a collection/group of integrated and networked hardware, software and Internet infrastructure (called a platform).
  - Using the Internet for communication and transport provides hardware, software and networking services to clients
- These platforms hide the complexity and details of the underlying infrastructure from users and applications by providing very simple graphical interface or API (Applications Programming Interface).
- In addition, the platform provides on demand services, that are always on, anywhere, anytime and any place.

Cloud Layers and Delivery

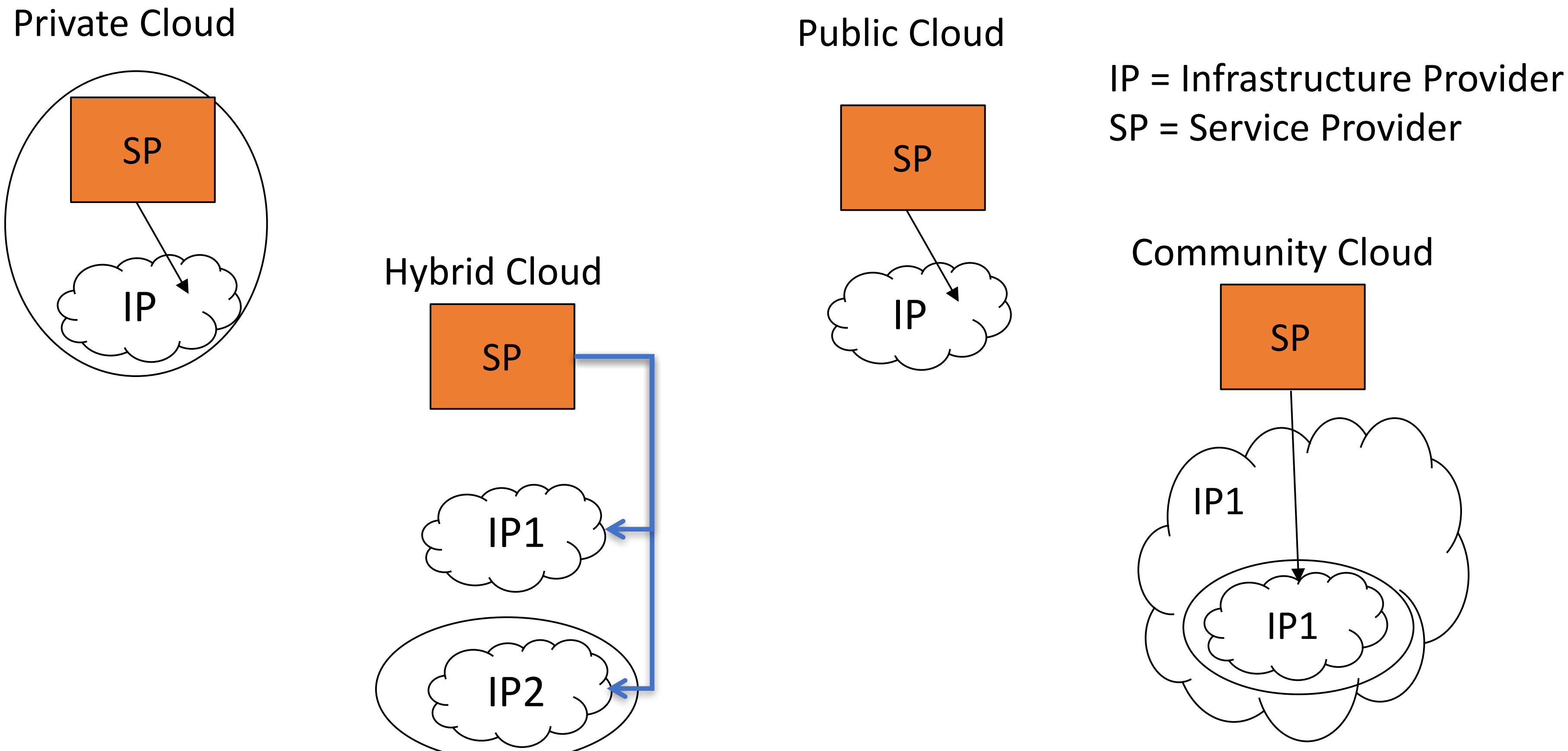


Actors/Stakeholders in Cloud

- Infrastructure providers
  - E.g., Amazon AWS, Cisco
- Service providers
  - E.g., Microsoft Azure, Google, Amazon EC2, EMC
- Service consumers / end users
- Service Brokers
  - E.g., Appirio, Cloud compare, Cloudmore

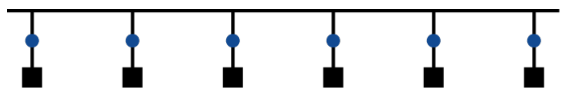
Cloud Deployment Models

- There are 4 basic deployment models

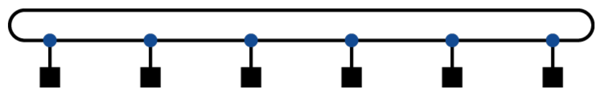


Interconnection Networks

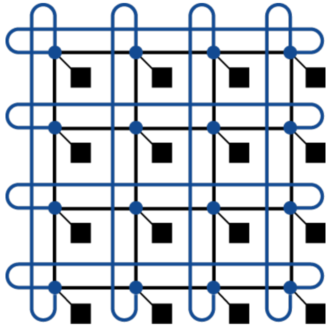
- Network topologies
  - Arrangements of processors, switches, and links



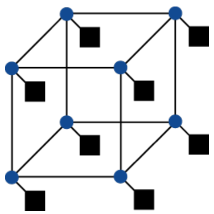
Bus



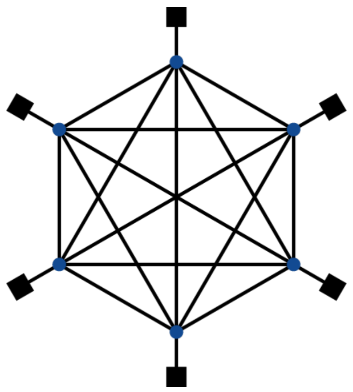
Ring



2D Mesh



N-cube (N = 3)



Fully connected

Network Characteristics

- Performance
  - Latency per message (unloaded network)
  - Throughput
    - Link bandwidth
    - Total network bandwidth
    - Bisection bandwidth
  - Congestion delays (depending on traffic)
- Cost
- Power
- Routability in silicon

Parallel Benchmarks

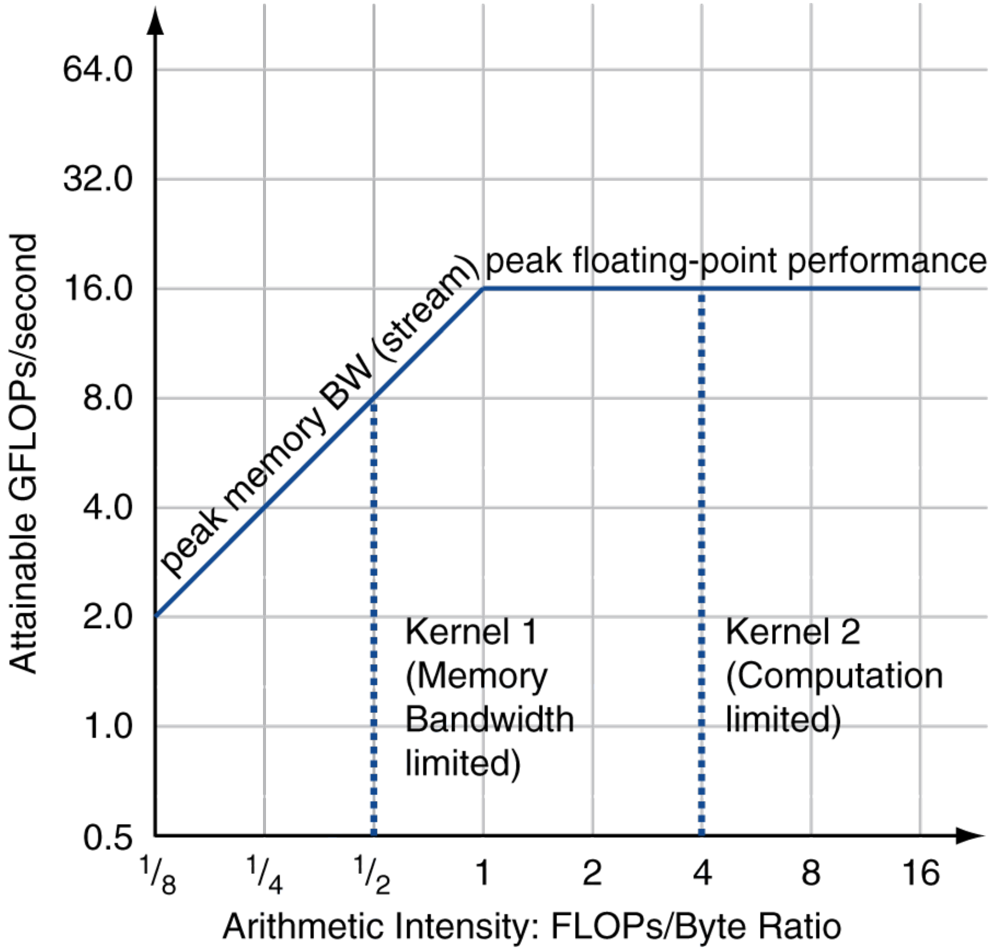
- Traditional benchmarks
  - Fixed code and data sets
- Linpack: matrix linear algebra
  - Performance (world fastest computer)
- SPECrate: parallel run of SPEC CPU programs
  - Job-level parallelism (throughput)
- SPLASH: Stanford Parallel Applications for Shared Memory
  - Mix of kernels and applications, strong scaling
- NAS (NASA Advanced Supercomputing) suite
  - computational fluid dynamics kernels, weak scaling
- PARSEC (Princeton Application Repository for Shared Memory Computers) suite
  - Multithreaded applications using Pthreads and OpenMP

Modelling Performance

- Architectural diversity – multithreading, SIMD, GPUs
  - Need for simple performance model for all architecture types
- Parallel computers peak floating-point performance
  - Depends on kernels speed on a given computer
- Multicore chip peak floating-point performance
  - Collective peak performance of all the cores on the chip
- Arithmetic intensity: ratio of floating-point operations per byte of memory accessed by a program
  - Memory system demand
- Stream benchmark provides peak memory performance

Roofline Diagram

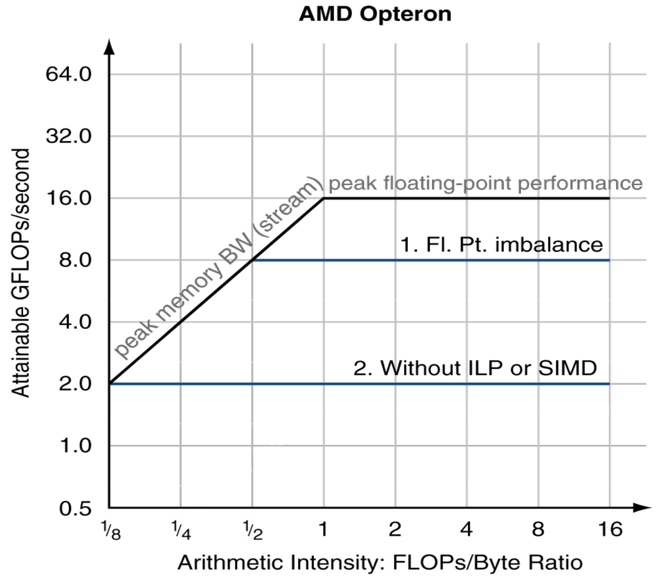
- Ties together
  - Peak floating-point performance
  - Arithmetic intensity
  - Peak memory performance
- Defines an upper bound to performance
- Data based on AMD Opteron X2
  - Dual cores @ 2GHz

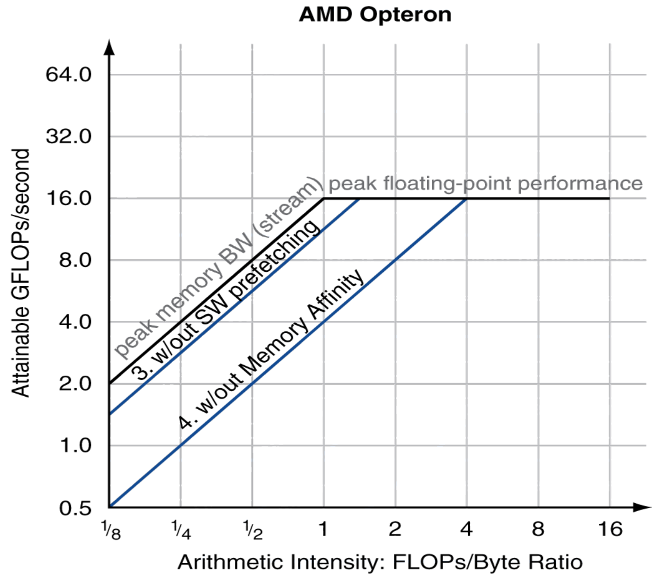


Attainable GPLOPs/sec
= Max (Peak Memory BW × Arithmetic Intensity, Peak FP Performance)

Optimizing Performance

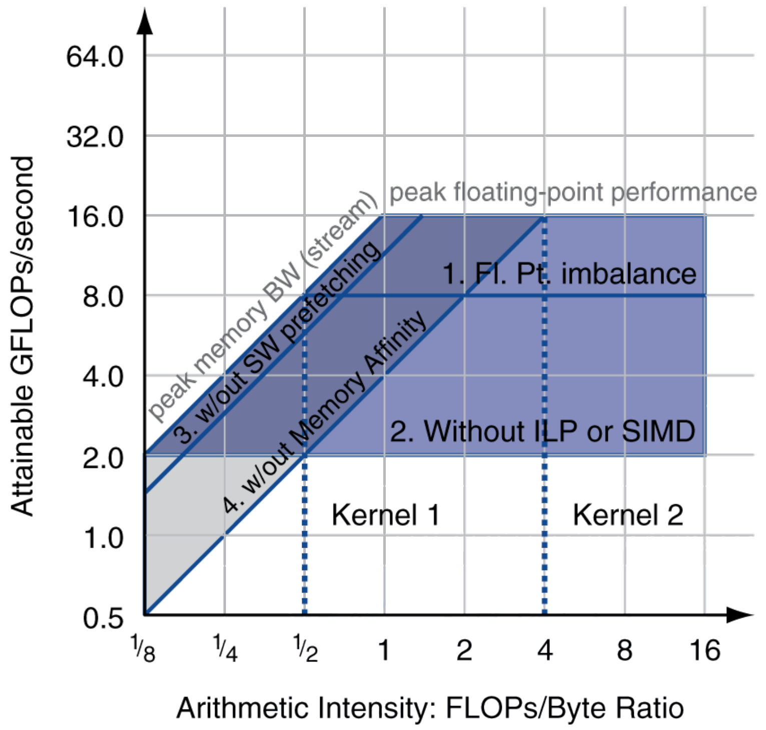
- Optimize FP performance
  - Balance adds & multiplies instructions
  - Improve superscalar ILP and use of SIMD instructions
    - Loop unrolling
- Optimize memory usage by reducing bottlenecks
  - Software prefetch
    - Avoid load stalls
  - Memory affinity
    - Allocate thread and data on same processor
    - Avoid non-local data accesses





Optimizing Performance

- Choice of optimization depends on arithmetic intensity of code



- Arithmetic intensity is not always fixed
  - May scale with problem size
  - Caching reduces memory accesses
    - Increases arithmetic intensity

Fallacies

- Peak performance tracks observed performance
  - Marketers like this approach!
  - In multiprocessor, they simply multiply
  - Need to be aware of bottlenecks that limits performance
- Amdahl’s Law doesn’t apply to parallel computers
  - Since we can achieve linear speedup
  - But only on applications with weak scaling

Pitfalls

- Not developing the software to take account of a multiprocessor architecture
  - Example: using a single lock for a shared composite resource
    - Serializes accesses, even if they could be done in parallel
      - Silicon Graphic Operating System
  - A possible solution
    - Use finer-granularity locking

Concluding Remarks

- Goal: higher performance by using multiple processors
- Difficulties
  - Developing parallel software
  - Devising appropriate architectures
- SaaS importance is growing and clusters are a good match
- Performance per dollar and performance per Joule drive both mobile and WSC

Concluding Remarks

- SIMD and vector operations match multimedia applications and are easy to program

- For x86 we expect
  - Two cores per chip every two years
  - Double SIMD width four yeas