Data Centers

Growth of Big Services

- Big data: web grows, data collection grows
- Big customers: number of people connecting grows, bandwidth grows, time connected grows
- Big competition: more advanced uses of data improve customer experience

Data center as a computer

- Big applications treat a whole data center (or set) as a computer upon which to run
  - Service layer provides abstractions above the OS

Data Center Hardware

- Driven by price/performance for workload

Architecture Overview

- Driven by price/performance for workload
- Server: 1 u rack
- Rack: 40 servers + gigabit ethernet switch
  - 1 gb ethernet for efficiency; locality matters a lot
- Storage: either NAS/SAN or local disks
  - NAS/SAN easier to program, ignore locality, but hard to scale (and expensive), must be made highly reliable
  - GFS (local disks) easier to scale, can take advantage of locality, more reliable

Data Center Properties

- Homogeneous platform
  - Clusters tend to all be similar machines to make purchasing/management more efficient
  - Also makes coding more efficient: don't need to worry about widely varying capacities
- Small # platform types
  - Storage light: 2-4 cores, boot disk
  - Storage heavy: 16-24 disks

Scaling Google Down

- Why is WSC important outside Google/Microsoft/ Amazon?
  - Economics of low-end hardware apply everywhere
  - Small organizations might grow big
- Do private clouds make sense for most organizations?
  - Perhaps not: can lose benefits of multi-tenancy that makes it pay off (non-correlated peak loads), low capital expense
  - Power costs a lot more for small clouds, as do operational staffs

What's different about WSC

- Poorly connected compared to super computers
- Fault behavior important
  - Instead of keeping all nodes running, keep enough running but let others fail
- Energy consumption important
  - 50% of cost of data center over time is power

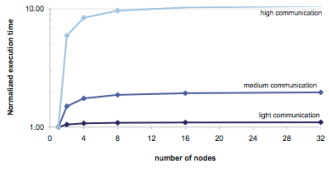
Warehouse compared to big SMP

- When does a big SMP (128 cores) help?
  - Lots of fine-grained communication
  - Does this exist for data center workloads?
    - sometimes; could make search a lot faster if it ran on one machine
- Challenge: workloads (at Google) can exceed the capacity of a single machine
  - still pay latencies of networking, lose much of the single machine benefit

Cost Efficient Hardware

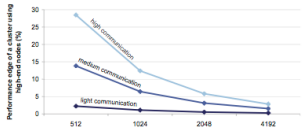
- Key observation 1: if problem fits on a machine, can get great speedup via large SMP if fine-grained parallelism is needed (can be cost effective)
- Key observation 2: if problem is larger than a machine, a big SMP doesn't help much, costs more, and doesn't simplify coding
- so: go for commodity parts, pushed down by sales at BestBuy

Scalability of nodes



Light communication: scales well to lots of nodes
Heavy communication: Scales poorly once you Have a few nodes

benefit of large nodes declines for larger clusters, is small for light communication



Processor sizing

- What processor is needed?
  - High-end? e.g. top-of-the-line Core i7/xeon?
    - cost/performance can be high
    - Code is memory bound; higher clock speeds and larger caches may not help
  - Medium? Google says yes …
  - Low end: Atom, Via Nano, Arm?
    - Better price/performance, performance/watt
    - But: worse
      - scheduling: harder to schedule at fine grain, more accurate load balancing needed
      - More networking needed to connect everything
      - Latency matters for some workloads; it isn't all memory and I/O
    - Related work: Gordon, FAWN push this for I/O bound lookup workloads

Latency within a Datacenter

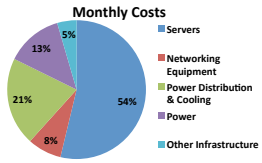
- Disk is the worst thing to access
  - Local memory best
  - Then memory of other node in rack
  - Then memory of other node in data center
  - Then disk
  - Disk bandwidth similar to bandwidth from other nodes in cluster

Failures

- Scale of data center leads to failures
  - 1.2 – 16 restarts per server per year
  - 4% of disks fail every year

Power & Related Costs [Will] Dominate

- **Assumptions:**
  - Facility: ~$88M for 8MW facility
  - Servers: Roughly 46k @ $1.45k each
  - Server power draw at 30% load: 80%
  - Commercial Power: ~$0.07/kWhr
  - PUE: 1.5



3yr server, 4yr net gear, & 10 yr infrastructure amortization

- **Observations:**
  - 34% costs functionally related to power (trending up while server costs down)
  - Networking high at 8% of costs & 19% of total server cost

Updated from: http://perspectives.mvdirona.com/2008/11/28/CostOfPowerInLargeScaleDataCenters.aspx

Power Usage in Datacenters

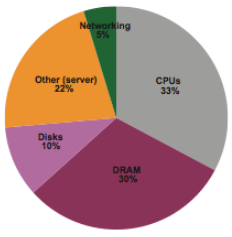
- Power is a large cost
  - Pay large cost for peak power – must build equipment, utility must guarantee to provide
  - 20% of facility cost is power redundancy (backup
  - Actual power use not as expensive
- Often reported as PUE – power usage effectiveness
  - All computing hardware = 1.00
  - Everything else: adds on
    - if same power for cooling, power distribution, then PUE is 2
    - If only 20% more needed, then PUE is 1.2
  - State of the art: 1.2
    - Local batteries on computers (avoid big UPS)
    - Better power distribution
    - Efficient cooling: build in cold places

Where Does the Power Go?

- **Assuming a good data center with PUE ~1.5**
  - Each watt to server loses ~0.5W to power distribution losses & cooling
  - IT load (servers & storage): 1/1.5 => 67%
  - Network gear <4% total power (5.8% of IT load)
- **Power losses are easier to track than cooling:**
  - Power transmission, conversion, & switching losses: 11%
  - Cooling losses the remainder:100-(67+11) => 22%
- **Observations:**
  - **Utilization & server efficiency improvements very highly leveraged**
  - **Cooling costs unreasonably high**
  - **PUE improving rapidly**

Where does power go?

- Within useful equipment:
  - CPU is largest
  - DRAM is next
  - Disks lower (not used as much)
- CPU power has been reduced by Intel, but memory not as much



Data Center Software

- Platform software:
  - bios, kernel, os services (e.g. DNS)
- Cluster infrastructure – distributed systems services
  - RPC
  - MapReduce
  - Dryad
  - Hadoop
  - GFS, BigTable, Chubby Sawzall
- Applications (SaaS)
  - GMail, search

Data center workloads

- Interactive workloads:
  - Lots of request parallelism
    - millions of people submitting independent requests
  - Lots of data parallelism
    - Indexes of the whole web
  - Short-running requests for low latency
    - Simplifies scheduling/resource consumption
- Batch workloads: typically data processing
  - Lots of data parallelism: cached data, logs

Data Center Workloads

- Request-level parallelism
  - Independent requests
  - Challenge: take advantage of parallelism, don't lose it through serialization
- Data Parallelism
  - Big data sets, read-mostly
  - Gmail: smaller individual data items, but typically not shared
- Workload churn:
  - SaaS allows frequent upgrades: replace pieces every week/month/year depending on level (infrastructure is the slowest to be replaced …)

Core Reliability/Performance techniques

- All seen before:
  - Replication for perf/avail
  - Partitioning for perf/avail
  - load balancing for perf
  - Health check/watchdogs for avail
  - Integrity checks for avail – asserts, checksums
  - Compression for perf
  - Eventual consistency for perf/avail
- Note: redundant execution not widely used because too expensive: double the number of machines

Infrastructure Software

- Resource management:
  - scheduling allocation across a DC: priorities, quotas, task mgmt
  - Abstractions
  - Automation
  - Specify job requirements (CPU, disk, memory, network) as input

Hardware abstraction

- Like OS: not set of disks, but GFS
  - GFS: files
  - Chubby: locks
  - Dynamo: key/value
  - Message passing – protocol buffers, RPC

Management

- Deploy software to a cluster
  - Copy bits
  - Launch services
  - Upgrade software
- Monitoring
  - Watchdogs, heart beats
  - Performance monitors:
    - fault tolerance causes failures to look like loss in performance
- Key idea: developing software means developing a deployment/management infrastructure
  - Cost of software is development + operations, want to keep operations low
  - Automate as much as possible

Monitoring Infrastructure

- Service level dashboard: online monitoring
  - operator can figure out how a service is behaving, look at rates & derivatives to see disruptions
  - Platform monitoring: is hardware running?
    - Can be masked by fault tolerant software
- Performance Debugging: offline/testing
  - Black box: analyze network, look for statistical inferences
  - Instrumentation: modify code to log, annotate data packets

Programming Frameworks

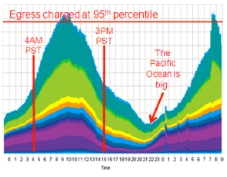
- Simplify job of common programming tasks
  - Map/Reduce for parallel data analysis
  - BigTable/Dynamo handle data partitioning
  - ProtocolBuffers handle data serialization
- Allow graceful upgrade
  - support version n + n-1 + n+1 at the same time (or two of the three) for rolling upgrades

Utilization & Economics

- Server utilization problem
  - 30% utilization VERY good &10% to 20% common
    - Expensive & not good for environment
  - Solution: pool number of heterogeneous services
    - Single reserve capacity pool far more efficient
    - Non-correlated peaks & law of large numbers
  - Example:
    - I/O bound workloads + memory/CPU bound workloads
- Charge back models drive good application owner behavior
  - Cost encourages prioritization of work by application developers
  - High scale needed to make a market for low priority work

Resource Consumption Shaping

- Resourced optimization applied to full DC
- Network charge: base + 95th percentile
  - Push peaks to troughs
  - Fill troughs for "free"
  - Dynamic resource allocation
    - Virtual machine helpful but not needed
  - Symmetrically charged so ingress effectively
- Power also often charged on base + peak
  - Push some workload from peak into "free" troughs
  - S3 (suspend) or S5 (off) when server not needed
- Disks come with both IOPS capability & capacity
  - Mix hot & cold data to "soak up" both resources
- Incent priority (urgency) differentiation in charge-back model
  - Charge application groups based on their resource usage + power



Research Problems

- Power:
  - as workload varies, how do you make power consumption vary proportionally
    - Processors can be turned down, but disks take a long time to spin down
    - Memory consumes power for refresh
- Utilization:
  - Interactive servers have low utilization to reduce queuing
  - Idle periods are short (1-100 ms)
- Solutions:
  - Consolidate workload onto fewer machines
    - but moving data expensive …