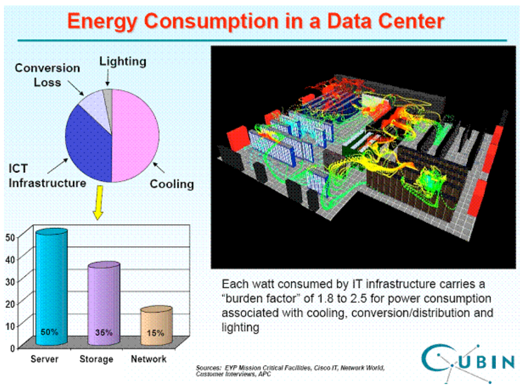
The Network I

Low Power Computing
David Andersen
Carnegie Mellon University

Themes

- Last two classes: Saving power by running more slowly and sleeping more.
- This time: Network intro; saving power by architecting for less power; optics
- Later lecture: Saving power on the network by running more slowly and sleeping

Energy in the DC: ~15% network?



Each watt consumed by IT infrastructure carries a "burden factor" of 1.8 to 2.5 for power consumption associated with cooling, conversion/distribution and lighting

Sources: EYP Mission Critical Facilities, Cisco IT, Network World, Customer Interviews, APC

Chart from CUBIN report (Melbourne)

Energy by Technology

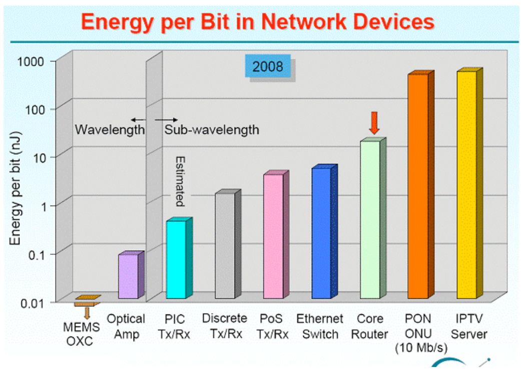
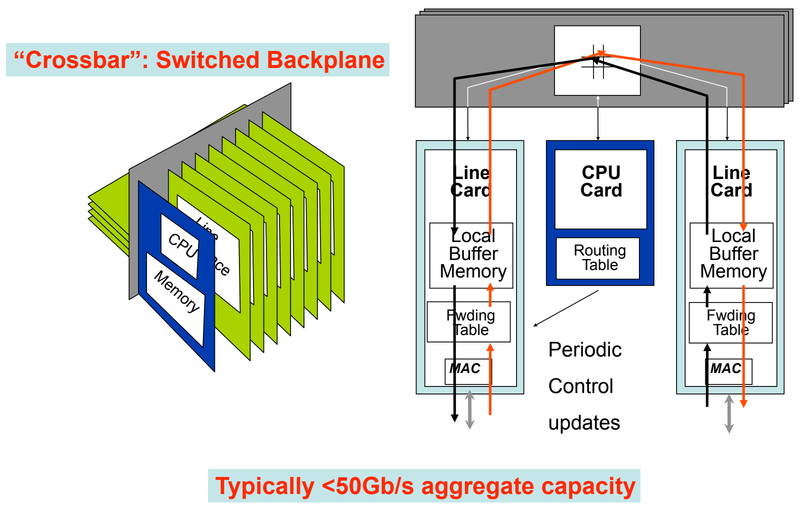


Chart from CUBIN report (Melbourne)

So...

- Network energy still dominated by server energy
  - But servers are going to get *a lot better*.
  - And faster. 10GE draws a lot of power today, and is starting to show up on motherboards.
- Good time to start figuring out how to solve the problem.

Third Generation Routers



Crossbars

- N input ports, N output ports
  - (One per line card usually)
- Every line card has its own forwarding table/ classifier/etc – removes CPU bottleneck

What's so hard here?

- Back-of-the-envelope numbers
  - Line cards can be 40 Gbit/sec today (OC-768)
    - Undoubtedly faster in a few more years, so scale these #s appropriately!
  - To handle minimum-sized packets (~40b)
    - 125 Mpps, or 8ns per packet
    - But note that this can be deeply pipelined, at the cost of buffering and complexity. Some lookup chips do this, though still with SRAM, not DRAM. Good lookup algos needed still.
- For every packet, you must:
  - Do a routing lookup (where to send it)
  - Schedule the crossbar
  - Maybe buffer, maybe QoS, maybe filtering by ACLs

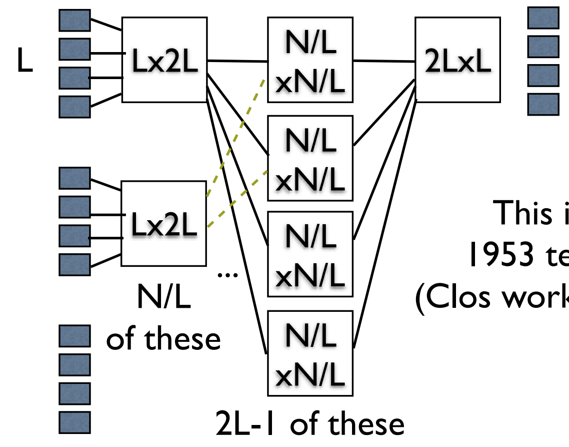
Crossbars get Tough

- Speed (reconfiguration rate): w/electronics, going past 40Gbps is quite tough.
- *N* - the number of ports on a single crossbar switch
  - Crossbar complexity is N^2
  - Scheduling becomes a big challenge
  - Which input port -> output port at time t

Switch Scheduling

- Variable length packets are segmented into fixed-sized cells
- Input linecards have queue of cells to send to output linecard(s)
- Crossbar switch - can make an arbitrary bipartite matching of input & output ports
- Scheduling *computes* this matching
  - Must run fast, be implementable in hardware
  - Must be fair over time (don't starve any inputs)
  - Must achieve 100% throughput
    - Example (not used): maximum weight bipartite matching. Runs in $O(N^3 \log N)$

Clos network, *N* line cards - non-blocking



This is old stuff - 1953 telecom world. (Clos worked for Bell Labs)

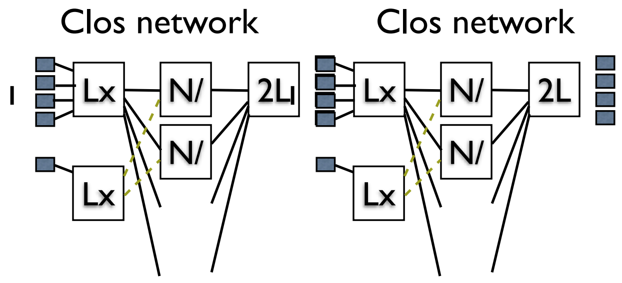
Clos isn't enough

- Definite improvement
  - Network of 676 line cards could be:
    - 26 groups of 26 line cards
    - 52 switches of 26x52
    - 52 switches of 26x26
    - We can build those switches...
- Scheduling a nonblocking Clos network like that is a beast.

Load Balanced Switching

- If scheduling is hard... don't!
- Input switches spread packets round-robin among middle-stage switches
- Middle stage switches direct to correct output switch
- Combining this idea with clos still works...

"Scaling" arch - re-imaged



"Intermediate" line cards maintain VOQs. But they're really just the same line cards.

Looking at one real router

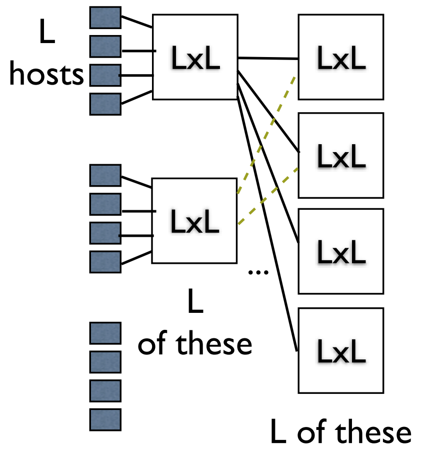
- Cisco CRS-1
- Largest router in the world
- Multiple racks, integrated with high-speed switching fabric
- Terabits of capacity
- Fabric chassis: ~10 kW
- Line card chassis: ~11 kW

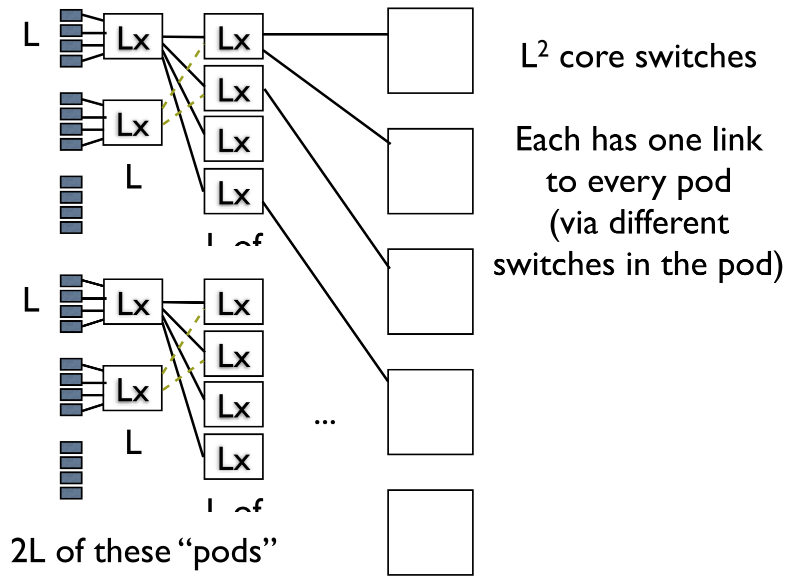
Cisco CRS-1

- Multi-rack system; capacity <= 92 Tbps (eventually).
- Looks like the arch from Keslassy: sends to middle switches in round-robin fashion, which relay to 3rd.
- Shelves interconnected with parallel optical links.
- Other power features: Line cards have a massively multi-core (188 cores, 32 bit RISC processors) processor to do work.
  - Packet processing is *very* parallelizable
- This is the "HFR" - prev generation was the "BFR". :)

Datacenter Ethernet

- Ethernet uses spanning tree to prevent loops.
- (Illustration)
- Only one link from each switch to root
- In a tree, 1/2 of traffic goes through the root. => bottleneck. Huge, fast switch.





Fat Trees / Clos networks

- A Fat Tree is a Clos network, with parameters chosen to maximize possible size for a given size switch
- This paper differs from the "Scaling" paper in the load balancing scheme.
  (One might imagine combining them, since the load balancing enabled passive optical links. Warning: MEMS are expensive. :)
- "...A central scheduler ... with global knowledge of all active *large* flows in the network."

Combining challenge: Packet Reordering

- Practically, networks must avoid re-ordering packets within a flow
  - TCP treats this as loss. Too much reordering kills throughput.
- The "Scaling" approach: bound mis-sequencing (might be hard in datacenter with heterogenous distances, etc) Re-sequencing buffer in third stage of bounded size
- Fat tree paper: Pinned route per flow. (Note relwork about thinking machines!)