L1: Intro to Computer Systems: Complexity

Frans Kaashoek and Nickolai Zeldovich

6.033 Spring 2011

http://web.mit.edu/6.033

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

http://web.mit.edu/6.033

- Schedule has all assignments
  - Every meeting has preparation/assignment
  - First deliverable is due tomorrow
  - Read the Therac paper for Friday
- Return sign-up sheet at the end of lecture(if you didn’t do so yesterday)
  - We will post sections assignment tonight

Monday	Tuesday	Wednesday	Thursday	Friday
feb.2 ***Registration Day***	feb.3 REC 1 Worse is Better ***First day of classes***	feb.4 LEC 1 **Intro to systems** **Preparation: Read 1.1, 1.2, 1.3 (click on link)**	feb.5 *REC 2* **The Architecture of Complexity** Preparation: Read Simon paper	feb.6 Writing Program Recitation Preparation: Read Therac-25 paper and Writing Assignment **Assigned: One-pager #1**

Today

- What is a system?
  - Challenge: controlling complexity
- Problem types due to complexity
  - Emergent properties, propagation of effects, incommensurate scaling
- Examples of problem types
- Approach to coping

What is a system?

- 6.033 is about the design of computer systems
- System = *Interacting set of components w. a specified behavior at the interface with its environment*
- Examples: Bank ATMs, Web, Linux
- Much of 6.033 will operate at design level
  - Relationships of components
  - Internals of components that help structure

Challenge: coping w. complexity

- Hard to define; symptoms:
  - Large # of components
  - Large # of connections
  - Irregular
  - No short description
  - Many people required to design/maintain
- Technology rarely the limit!
  - Indeed tech opportunity can be a problem
  - Limit is usually designers' understanding

Problem Types in Complex Systems

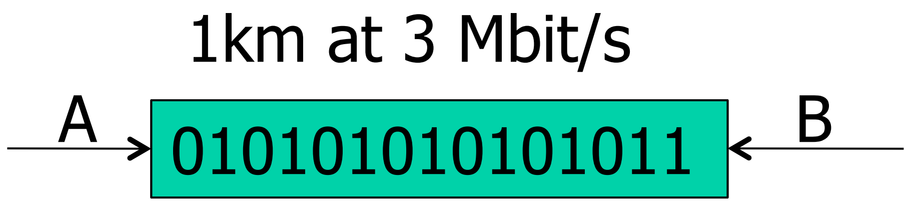
- Emergent properties
  - surprises
- Propagation of effects
  - Small change -> big effect
- [Incommensurate] scaling
  - Design for small model may not scale

- Problems show up in non-computer systems

Emergent Property Example: Ethernet

- All computers share single cable
- Goal is reliable delivery
- Listen before send to avoid collisions

Will listen-while-send detect collisions?



- 1 km at 60% speed of light = 5 microseconds
  - A can send 15 bits before bit 1 arrives at B
- A must keep sending for 2 * 5 microseconds
  - To detect collision if B sends when bit 1 arrives
- Minimum packet size is 5* 2 * 3 = 30 bits

3 Mbit/s -> 10 Mbit/s

- Experimental Ethernet design: 3Mbit/s
  - Default header is: 5 bytes = 40 bits
  - No problem with detecting collisions
- First Ethernet standard: 10 Mbit/s
  - Must send for 2*20 μseconds = 400 bits
  - But header is 14 bytes
  - ➢ Need to pad packets to at least 50 bytes
- ➢ Minimum packet size!

Propagation of Effects Example (L. Cole 1969)

- WHO attempted to control malaria in North Borneo
- Sprayed villages with DDT
- Wiped out mosquitoes, but ….
  - Roaches collected DDT in tissue
  - Lizards ate roaches and became slower
  - Easy target for cats
  - Cats didn’t deal with DDT well and died
  - Forest rats moved into villages
  - Rats carried the bacillus for the plague

➢ WHO replaced malaria with the plague

Galileo in 1638

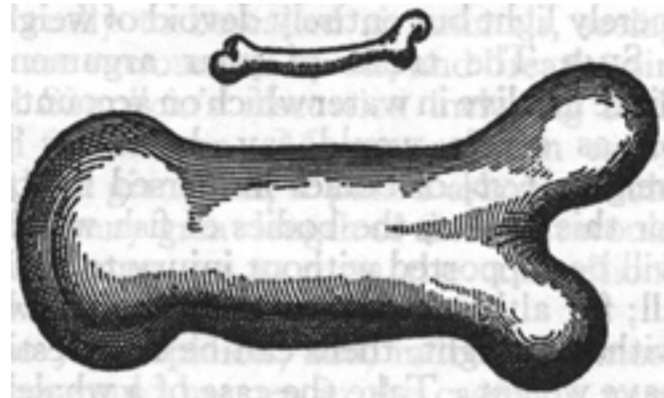


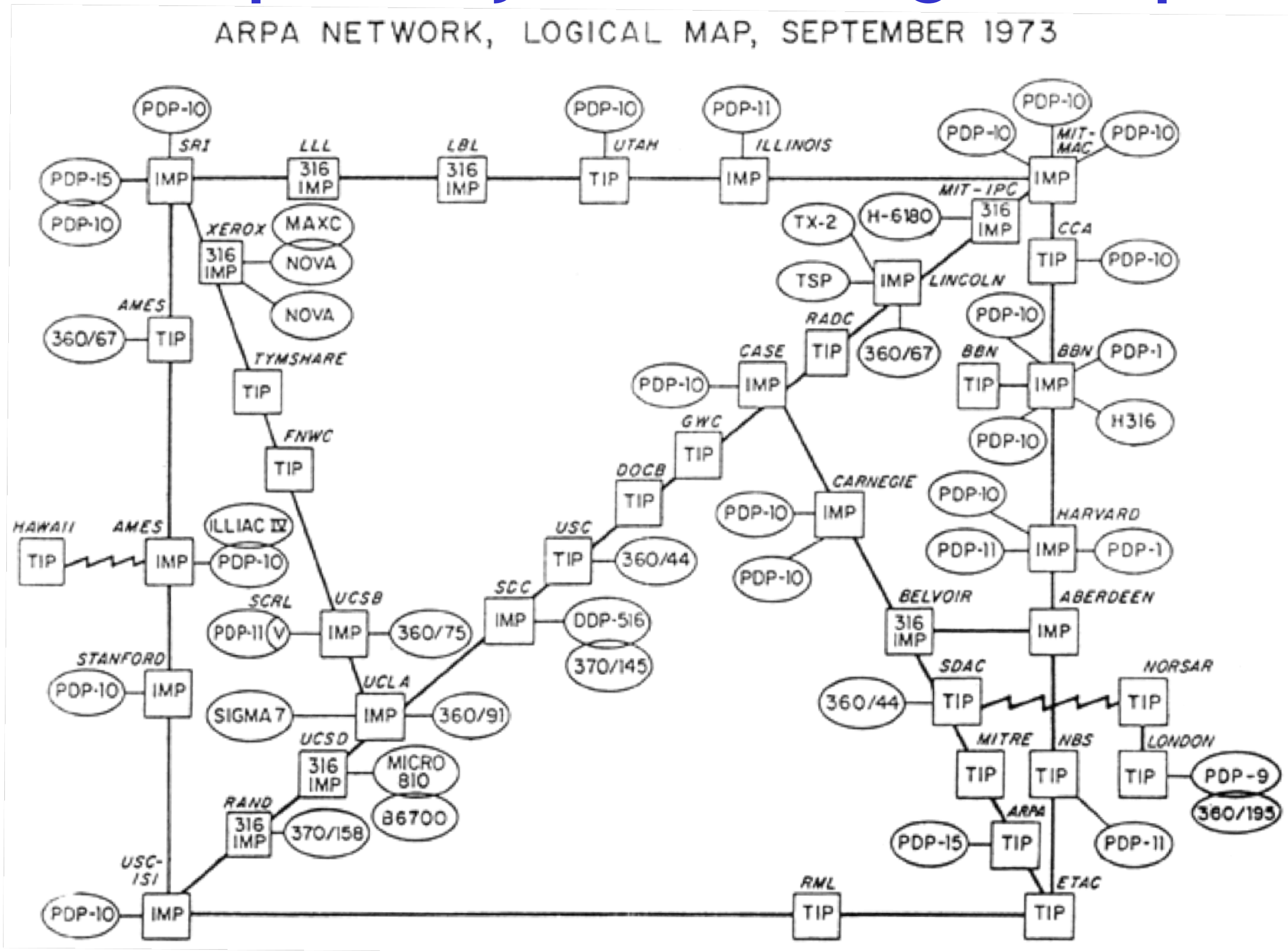
Fig. 27

“To illustrate briefly, I have sketched a bone whose natural length has been increased three times and whose thickness has been multiplied until, for a correspondingly large animal, it would perform the same function which the small bone performs for its small animal. From the figures here shown you can see how out of proportion the enlarged bone appears. Clearly then if one wishes to maintain in a great giant the same proportion of limb as that found in an ordinary man he must either find a harder and stronger material for making the bones, or he must admit a diminution of strength in comparison with men of medium stature; for if his height be increased inordinately he will fall and be crushed under his own weight. Whereas, if the size of a body be diminished, the strength of that body is not diminished in the same proportion; indeed the smaller the body the greater its relative strength. Thus a small dog could probably carry on his back two or three dogs of his own size; but I believe that a horse could not carry even one of his own size.” [Dialog Concerning Two New Sciences, 2nd Day]

Mouse -> Elephant (Haldane 1928)

- Mouse has a particular skeleton design
  - Can one scale it to something big?
- Scaling mouse to size of an elephant
  - Volume ~ $O(n^3)$
  - Bone strength ~ cross section ~ $O(n^2)$
  - Mouse design will collapse
- Elephant needs different design than mouse

A computer system scaling example



Incommensurate scaling

- Scaling the Internet
  - Size routing tables (for shortest paths): $O(n^2)$
    - Hierarchical routing on network numbers
    - Address is 16 bit network # and 16 bit host #
  - Limited networks (2^{16})
  - Network Address Translators and IPv6
- Scaling Ethernet's bit-rate
  - 10 mbit/s: min packet 64 bytes, max cable 2.5 km
  - 100: 64 bytes, 0.25 km
  - 1,000: 512 bytes, 0.25 km
  - 10,000: no shared cable

Sources of Complexity

- Many goals/requirements
- Interaction of features
- Performance

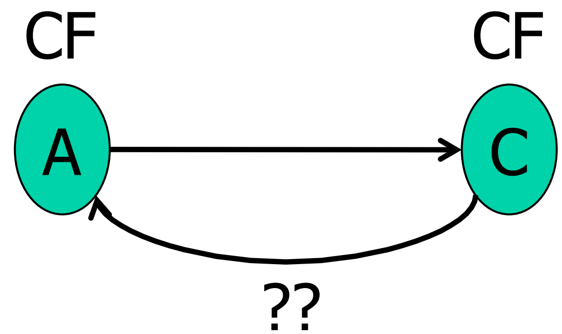
Example: more goals, more complexity

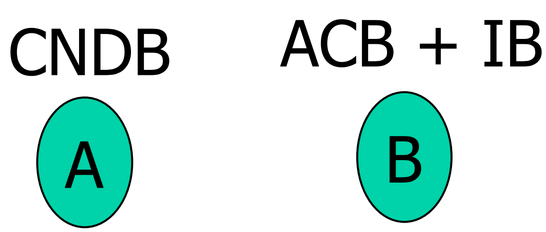
- 1975 Unix kernel: 10,500 lines of code
- 2008 Linux 2.6.24 line counts:

85,000	processes
430,000	sound drivers
490,000	network protocols
710,000	file systems
1,000,000	different CPU architectures
4,000,000	drivers
7,800,000	Total

Example: interacting features, more complexity

- Call Forwarding
- Call Number Delivery Blocking
- Automatic Call Back
- Itemized Billing





- A calls B, B is busy
- Once B is done, B calls A
- A's number on appears on B's bill

Interacting Features hidden

- Each feature has a spec.
- An interaction is bad if feature X breaks feature Y.
- ...
- The point is not that these bad interactions can't be fixed.
- The point is that there are so many interactions that have to be considered: they are a huge source of complexity.
- Perhaps more than n^2 interactions, e.g. triples.
- Cost of thinking about / fixing interaction gradually grows to dominate s/w costs.
- The point: Complexity is super-linear

Example: more performance, more complexity

- One track in a narrow canyon
- Base design: alternate trains
  - Low throughput, high delay
  - Worse than two-track, cheaper than blasting
- Lower delay w/ a siding and two trains
  - Precise schedule
  - Risk of collision / signal lights
  - Siding limits train length (a global effect!)
- Point: performance cost super-linear

Summary of examples

- Expect surprises
- There is no small change
- 10x increase $\Rightarrow$ perhaps re-design
- Just one more feature!
- Complexity is super-linear
- Performance cost is super-linear

Coping with Complexity

- Simplifying insights / experience / principles
  - E.g., “Avoid excessive generality”
- Modularity
  - Split up system, consider separately
- Abstraction
  - Interfaces/hiding, e.g. standard size windows
  - Helps avoid propagation of effects
- Hierarchy
  - Reduce connections
  - Divide-and-conquer
- Layering
  - Gradually build up capabilities

6.033 Approach

- Lectures/book: big ideas, technology, examples
- Recitations: papers, discussion
  - Design examples
  - Writing examples: core prob/soln vs detail
  - Learn how to read a paper, skim vs meat
- Design projects: practice designing and writing
  - Design: choose problem, tradeoffs, structure
  - Writing: explain core ideas concisely
- Exams: focus on reasoning about system design
- Ex-6.033 students: papers and recitations

Example 6.033 Readings

- Therac-25

  bad design, at many levels. detailed post-mortem
- UNIX

  “New Jersey” design
- MapReduce

  Elegant system design for processing much data
- End-to-End Argument

  Captures a non-obvious and useful design argument
- System R

  Recovery from crash at any point

Class plan

- Next lecture: computer systems are different
- Naming: gluing modules together
- Client/server: enforced modularity
- Networks: hard boundaries between modules
- Reliability and transactions: handing failures
- Security: handling malicious failures