CS529
Multimedia Networking

Experiments in Computer Science

Introduction

- Some claim computer science is not an experimental science
  - Computers are man-made, predictable so computer science is a theoretical science (like Mathematics)
  - But too many components (including humans!) to be predictable – need experiments to study
- Some claim system development is computer science
  - Building an OS or a federated database
  - Rather, that is *computer engineering*, and science comes after

Theory and Engineering

- Computer theory can only take you so far

  "Beware of bugs in the above code; I have only proved it correct, not tried it."

  - Donald E. Knuth

- Computer engineering can only take you so far
  - While building *apparatus* is skillful, unless grants new knowledge it is wasted
  - Need *science* to increase knowledge
- Use **Experiments** to evaluate theory or apparatus!

Experiments in Computer Science

"*The fundamental principle of science, the definition almost, is this: the sole test of the validity of any idea is experiment*"

— Richard P. Feynman

- Tried and true experimental scientific methodology from Physics, Biology, Chemistry ...
  - Often (?) *not* followed in Computer Science
- Let's be better Computer S*cientists*!

Scientific Methodology

- Observe
  - (Devise solution)
- Hypothesize
- Design
- Experiment
- Analyze
- Report

Methodology: Observe and Understand

- Find Problem
  - Test: *Netscape Audio*
  - Build: *Audioconference*
  - Read: *Kevin Jeffay says…*
- Understand Relationships
  - *UDP loses packets*
  - *TCP increases delay*
  - *P-frames depend upon I-frames*
  - (From background in **this** class!)

Methodology: Devise and Hypothesize

- Devise Solution (but empirical only is ok!)
  - *Claypool buffering algorithm*
  - *Claypool Reliable Audio Protocol (CRAP)*
- Make Hypothesis
  - Generalization about relationships
  - *Processor load induces jitter*
  - *Java virtual machine induces jitter*
  - Needs to be tested (not proven)

Methodology: Experiment

- Design Experiment
  - Variable: *processor workload*
  - Control: *baseline workload*
- Run Experiment

  "Whoa! That's not what I expected!"
  - Bug in code (*two processes sharing socket*)
    - Back to "Run"
  - Uncontrolled event (*system backup*)
    - Back to "Design"
  - Insufficient understanding *(Unix scheduling)*
    - Back to "Understanding"

Methodology: Analyze

- Interpretation and Evaluation
  - Statistical significance
    - mean, confidence intervals, correlation, goodness of fit
  - Does data *support* or *reject* hypothesis?
  - Explanation of other phenomena
    - *Processor load makes my phone call difficult*
    - *Java inadequate for interactive multimedia*

Methodology: Report

- Disseminate at an appropriate level
  - Workshops, Conferences, Journals
- Peer reviewed
  - Unlike "White" papers or Tech reports (and some Conferences!)
- Clarify in writing important
  - Reproducibility
- Includes analysis of complex data
  - Graphs! Statistical significance!
- Value presentation as a form of dissemination

Dirty Little Secrets

- Mini-experiments (no, "Pilot Tests")
- Hypotheses after the fact
  - Running yields understanding
- Results *here* mean results *there*
- Controlled system still says meaningful things about the real world
- Observing a system will not change it

Groupwork

- Create a CS hypothesis
- Describe how you would test it
- Is your work useful if your hypothesis is:
  - Accepted?
  - Rejected?