UCL Department of Computer Science
CS M038/GZ06: Mobile and Cloud Computing
Spring 2013
Kyle Jamieson and Brad Karp

Paper Presentation Guidelines

A presentation should consist of the following sections:

- **motivation and problem definition**
- **state main contributions of work**
- **description of central design (or in some cases, design of measurements)**
  - may not necessarily have time to present *every* detail of an algorithm or system; if so, present those that are most important:
    - to understanding how/why the system/design/algorithm works
    - to understanding results in the experimental evaluation
  - clarity very important here; describe in a "top-down" fashion, starting with the overall problem, identifying the parts of the solution, then identifying the sub-parts of those parts, &c.
- **experimental evaluation**
  - what *questions* do the authors ask?
  - what is the authors' hypothesis for each question? why?
  - may not have time to present *all* results in paper. Present most important ones if so.
  - for any graph you show:
    - start by stating the axes
    - explain the overall trend in the graph: *why* does the system behave as it does? justify your explanation by referring to relevant details of the system's design and experiment's design.
    - does anything in the graph seem "untidy" or odd? (examples: outlier points, high variance/wide confidence intervals in measurements, authors claim a trend but some of the data don't fit it, &c.) if so, note it and try to explain it.
- **related work**
  - what are the most closely related other systems/results? how are they similar? how are they different? is the difference between the work you are presenting and the related work significant?
  - we expect you to find and read the papers cited by the authors as related (you don't need to read them in as much detail as the paper you are presenting! but in enough detail to understand how the work they describe differs from that in the paper you are presenting.)
  - we expect you to search for more recent related work published after (or perhaps simultaneously with) the paper you are presenting
  - no need to claim the work you are presenting is "better" or "worse" than a particular piece of related work (though you may of course do so if you feel that way!); often it is simply that the two pieces of work are *different*–but you must articulate the precise difference (e.g., "these other offers solve a slightly different problem...")

- **future work**
  - successful scientific research requires scoping the problem: while a good paper contributes new knowledge, there are invariably more questions than can be answered in a single paper.
- **conclusion**
  - one slide!
  - review problem and why it's hard
  - review main contributions
  - offer your final critical assessment:
    * what are the strengths of the work?
    * what are its weaknesses?
    * what important questions are left unanswered?

Advice on giving a good talk:

- Number of slides: Most speakers presenting CS research find that they speak for around 1.5 to 2 minutes per slide (on average). So you should expect to have 18 - 24 slides for a 35-minute talk. (Note that if you introduce "animations" by repeating a slide with one element moved or added per slide repetition, this practice will inflate the number of slides–for the purposes of counting slides, consider the animation steps as a single slide.)
- Rehearse your talk many times: Most of the best scientific talks are rehearsed several times (for presenters learning to give talks, 10 or more rehearsals isn't unusual). Pay careful attention to length: you may not run over the allotted 35 minutes (we will stop you), and in the real world, running over allotted time is inconsiderate and selfish.
- Help one another present clearly: When you rehearse, listen to your groupmate's part critically. Try to envision that you are an audience member who hasn't read the paper as carefully as you (as the presenters) have, and ask yourself if you would be able to follow your groupmate's presentation. Take notes as your groupmate presents, noting what's unclear or what could be improved.
- Use examples to explain difficult ideas: If you need to explain a complicated algorithm or design, it's often useful and time-efficient to give a simple example of the problem, then show how the algorithm or design works on that example.
- Be constructively critical throughout: There can be weaknesses in almost any aspect of a piece of research: in the motivation, problem definition, assumptions made, design details themselves, experimental design, or even conclusions drawn (usually if there's a problem there, it relates to a problem elsewhere, perhaps with the assumptions made or experimental design). You aren't presenting the work as the author–you are presenting it as an impartial scientific peer. So you shouldn't view your job as "defending" the work; you should look at it objectively, and identify shortcomings and strengths.

Good guides to giving computer science talks:

- Willy Zwaenepoel's advice on presenting papers in a class: `http://www.cs.rice.edu/~alc/comp520/presentation.html`
- Dave Patterson's and Mike Dahlin's advice on how to give a *bad* talk (note: these slides describe things you should **not** do!): `http://www.cs.utexas.edu/users/dahlin/professional/badTalk.pdf`