How to present your Experimental Results?

Pao-Ann Hsiung
Embedded System Laboratory, National Chung Cheng University, TAIWAN
Contents

- How to do experiments?
- How to analyze your results?
- How to present your results?
- Conclusions
How to do experiments?

- Implementation Platform
- Implementation Details
- Examples
Implementation Platform

- **Fix** one platform for implementation
  - All experiments should be performed in the SAME platform
    - Otherwise, you need to explicitly mention which were performed in which platform and **WHY??**

- **Describe your platform in your Thesis**
  - Machine: OS, CPU, RAM, …
  - Language: C/C++/Java/VHDL/Verilog/SystemC (versions)
  - Tools: Compiler, Synthesizer, Profiler, Linter, …
  - Libraries: Graphics, GUI, …
Implementation Details

- Use only **standard** language versions: ISO C/C++, etc.
- Use the **latest** up-to-date functions
  - Don’t use obsolete functions: gets(), …
- Ensure **compatibility** across machines
  - Windows, Linux, FreeBSD, …
- **Measure** the following
  - CPU Time Usages, Memory Usages (getresources)
- **Parameterize** everything!!!
  - Don’t use “constants” in your program statements! Use #define or variables.
Implementation Details

- Perform **error checking!!**
  - Input files, wrong data input, enough memory, buffer overflow, …

- Variable **naming**
  - Variable names should be **consistent** with that in your Thesis!!

- **Last but not the least:** /* COMMENTS */
  - Add comments to your code wherever possible, especially in all the **data structure definitions** in header files
  - Use **English**, (preferably no Chinese!)

- One more please!!!
  - **Makefile:** that would save a lot of efforts!!!
Examples

- **Toy Example**
  - To illustrate the important **steps/concepts** in your method, algorithm, architecture, design, implementation
  - Run it both manually and using your programs!

- **Large Real-World Examples**
  - To illustrate how your method, algorithm, architecture, design **scales** to complex and large examples in the real-world

- **Random Examples**
  - To illustrate how your method, algorithm, architecture, design handles **future** systems
  - To show the **statistics**!!!
How to analyze your results?

- **Goals**
  - **To show the advantages** of your method
    - Novelty, time/space efficiency, scalability, simplicity, robustness, adaptivity, …
  - **To discover the limitations** in your method
    - Functional: Cannot do something …
    - Non-functional: Poor in doing something …
  - **To compare** your method with other existing methods
    - A naïve method
    - The most similar method(s)
    - Other methods
How to analyze your results?

- **The Expected**
  - Do you see what you *expected*?
    - Yes: Congratulations! You got what you wanted.
    - No: Find the cause!
      - Found: Congratulations! You got what you wanted.
      - Not found: Well, …
        - Was your expectation correct?
        - Was your design and implementation correct?

- **The Unexpected**
  - Do you see something *unexpected*?
    - No: Mmmm….
    - Yes: Explore further, may be you found something worth investigating!
How to analyze your results?

- Try to be as thorough as possible!
  - Don’t leave out any cases!!! (How many cases are there?)
    - Example: 6 features ➔ at least 6 different sets of experiments!
  - Don’t take the results for granted!!! (Think! Think! Think!)
- Be in the shoes of the authors with whom you are comparing!
  - Would you like to be criticized or deemed inferior without solid evidences? No!!!
How to present your results?

- **Use different formats**
  - **Tables**
    - For toy example and illustration
  - **Graphs**
    - For statistics and scalability
- **Use tools such as spreadsheets and graph plotters**
  - **MS Excel** (to collect your results)
  - **Matlab** (to co-relate your results)
  - **Gnuplot** (to plot your results)
Conclusions

- The way you do and the way you present your experimental results have a great impact on what the readers conclude about your work
- Be confident about your advantages
- Be humble about your limitations
- Be sure about your future work