Advice for Bachelor students for *PR with Bachelor Thesis – Structural Details*[1]

Putting together a thesis is hard. To make your life a little easier we have put together some general guidelines on structural details. This contains (1) a typical structure of how CS Bachelor theses look like and (2) some recommendations on how to conduct and report your research.

These recommendations are not set in stone, but are rather meant as a set of guidelines to help you prepare, conduct, and write your BSc thesis. We strongly recommend following these guidelines.

(1) Suggested structure for your thesis

While the structure of a thesis often depends on the specific area of work one is pursuing, there is a general common structure.

Abstract

Briefly summarize the main contributions of your work. Don't have a lengthy introduction (that's the job of the next section!). See it as the elevator-pitch! :) See also https://www.lightbluetouchpaper.org/2007/03/14/how-not-to-write-an-abstract/

1. Motivation

- motivate your topic
- clearly say what the problem is and why other related techniques/tools don't solve it
- clearly state the goals of your work; discuss the metrics how you can measure success (e.g. my new algorithm will be faster, people will be able to perform a certain task more quickly and with fewer errors, better user satisfaction, people will gain new insights, I can do something that has not been possible before but is important,

[1] Date: Jun 14, 2017
created by M. Sedlmair, Oct 13, 2014
modified by T. Möller, D. Ramsauer, P. Chlap, W. Klas, H. Hlavacs, S. Schröder, C. Löw, H. Weisgrab, P. Reichl, M. Sedlmair (Sep/Oct 2015)
modified by M. Sedlmair (Sept 2016)
modified by A.M. Faisst & T.Möller (Feb-Jun 2017)
finalized and agreed by the Faculty of Computer Science, Uni Wien, on Jun 14, 2017

...) Note: you will have to show later that you actually met the goals you claimed here!

- briefly explain the process of your work and the methods you used (e.g. collaborating with domain experts, multiple prototypes, ...)
- crisply summarize your main contributions at the end of the motivation section

2. Related Work

- Are there similar ideas to your idea? (Note: most likely, yes!)
- What should the related work section contain?
    - The related work section describes existing work that is *conceptually* similar to what you are working on. It needs to give a good overview of what others have done to overcome the same problem as you face, or conceptually similar problems.
    - Examples:
        - Assuming you are designing and implementing a new graph drawing algorithm, then you will need to identify and explain other existing graph drawing algorithms.
        - If you are building, for instance, a network visualization tool for a specific target audience, you should describe state-of-the-art in network visualization as well as in the target domain.
    - For all related works:
        - briefly explain what it is about and why it is related
        - reference it properly
        - say how your work is different and goes beyond it/is better (note: being solely different but not better is not a good argument)
    - IMPORTANT: What should related work *not* contain!
        - Related work is not the section where you talk about implementation details, such as which programming language or frameworks you have used. This belongs in the implementation section.
- Find literature
    - scientific: see below
    - non-scientific: browse the web and report general ideas that you find related to yours

3. Algorithm / Design / Major Idea

- Conceptually describe what you have done
- Explain how it works and why it is cool (i.e. helps solving the problem)
- Make use of screenshots to illustrate your implementation, your work!

3a. for more design-type papers:

- Justify design decision:
    - explain alternatives that you thought of

    - support your arguments either by literature that supports your decision
    - or by thoroughly justifying it based on the problem at hand
- Note: Don't give technical implementation details here, this comes in the next section

3b. for more algorithmic papers:

- Justify the design decisions for the algorithms:
    - explain alternatives that you thought of
    - support your arguments either by literature that supports your decision
    - or by thoroughly justifying it based on the problem at hand
- Explain how the algorithm works, provide details on the complexity of the algorithm, provide specification of the algorithm e.g. by means of pseudo code, point out how to implement the algorithm.
- Explain, outline, compare on the performance of the algorithm
- Note: Don't give technical implementation details here, this comes in the next section

3c. for more systems type papers:

- Justify system architecture decision:
    - explain alternatives that you thought of
    - support your arguments either by literature that supports your decision
    - or by thoroughly justifying it based on the problem at hand
- Explain the functionality of system components, specify interfaces, provide UML specification in order to cover the conceptual design of the system
- Explain the usage of standards, why standards are used (or maybe not used)
- Explain the demo application
- Note: Don't give technical implementation details here, this comes in the next section

4. Implementation

- explain how you engineered your tool/technique
- focus on the interesting bits here, you do not have to explain all your code
- one goal of this section is to allow someone else to reproduce your work
- describe:
    - where to find the code, how to compile it, how to deploy it
    - prerequisites for running the prototype or demo

5. (Preliminary) Evaluation and Discussion

- how did you evaluate that you reached your goals?
- did you use specific methods to do so? (e.g., a user study, a case study, usage scenario, formal mathematical proof, performance tests, (visual) comparison to state of the art, ...)

- explain methods and results of this evaluation
- discuss the results: here you should refer back to the most closely related work. Now you can say why your new thing is better (given the metrics outlined in the motivation).
- discuss lessons learned and unexpected findings
- Note that in a bachelor thesis you will often use "simple" validation methods, such as usage scenarios with example datasets, plus a solid discussion of the benefits. Note, however, that an evaluation -- even a small one -- is mandatory for a scientific thesis. This section, therefore, must not be missing.

6. Conclusions and Future Work

- give a concise summary of your main findings/insights/contributions
- explain assumptions and limitations of your approach
- outline future work
    - future work can be concrete next steps in this particular project
    - future work can be also more broadly such as streams of research that other people should focus more on

7. References

- provide a full list of references
- all references need to be cited in the text

Appendix / Supplemental Material (if any)

Evaluation / user study papers is another common form of theses we offer. Their layout differs slightly from the template provided above. For user study papers, please follow the outline provide by Saul Greenberg:

http://pages.cpsc.ucalgary.ca/~saul/hci_topics/assignments/controlled_expt/ass1_reports.html

Note: Many of the more general aspects of the template above might still be helpful (e.g. Motivation, Related Work, ...). It might therefore still be helpful to carefully read and think it through.

Length

A frequent question is, e.g. how many pages, references, etc. the final, written part of a Bachelor thesis needs. While the answer is 'it depends', here are some guidelines:

- approx. 30 pages (using the template provided) is a good aim

- if you have any spacious material, such as long code snippets, large and/or many screenshots, that you want to include but that are not core relevant, you can include them as supplemental materials
- references: As a rule of thumb, we are expecting 20+ references. At least 15 of those should be from scientific literature, that is research papers or scientific books.

Important note: We are not at all grading your thesis by bean counting! Rather we look at it in a more holistic way, judge the contributions, ideas, design, implementation, evaluation, discussion and presentation. That said, if you for instance "only" have 20 pages, but these really rock and have everything relevant in it, you of course still would get a good grade. Note, however, the challenge usually is the other way round: Instead of filling pages you might find it challenging to fit everything into the page limit.

Template

Please use the Latex or Word template from the course's Moodle page.

Submission

Please submit the following via Moodle (before the deadline, see administrative guidelines):

- PDF file of the Bachelor thesis
- (if applicable) running prototype (for instance, on the web)
  If you have developed software this is mandatory. Note, we do not have the time to run complicated instructions to get your prototype to run. That is, preferably your tool should run without any extra software, etc. However, we do know that in some cases that might not be feasible. If so, please clarify it with us upfront (i.e. before you send us the first version of your prototype). In this case, include a readme.txt in which all steps are explained properly.
- (if applicable) documented code
  If you have developed software this is mandatory.
- (if applicable) additional screenshots or a video
- (if applicable) study protocols and data

ALL files that you submit must start with your last name, e.g. "Mustermann_thesis.pdf".

(2) Tips on how to conduct and report your research

Finding Scientific Literature

Core relevant literature will depend on your thesis topic and should be clarified with your supervisor. However, some good sources for literature searches are

- Our library: http://bibliothek.univie.ac.at

- IEEE Xplore: http://ieeexplore.ieee.org/Xplore/home.jsp
- ACM digital library: http://dl.acm.org/
- Springer Link: http://link.springer.com/
- Elsevier: http://www.elsevier.com/
- IFIP Open Digital Library: http://dl.ifip.org/

Note: you need to be on campus or logged in via VPN to access full text material

Alternatively you might find relevant work via:

- Google scholar: https://scholar.google.at
- Citeseer: http://citeseerx.ist.psu.edu
- or simply google

Note 1: It is a good idea to engage in "snowball" exploration, i.e. following up on interesting references that you find in core-relevant papers.

Note 2: Do your literature search before starting the project and take notes. Otherwise, you will have forgotten your thoughts when it comes to writing up your thesis.

Note 3 (important): While most of the literature will go into the related work section of your thesis, also other sections of your thesis should be backed up with literature, e.g.:

- Motivation: Who dealt with similar problems, e.g. name the most important ones and briefly explain them (and then explain them in more detail in the Related Work section)
- Design: Cite papers that justify your design decisions (e.g. because they have conducted a comparative study, or took a similar decision that turned out to work well)
- Implementation: Cite frameworks, etc. that you used, e.g. D3, pbrt, paraview, etc.
- Evaluation/Discussion: Revisit related work and compare and discuss them given the new insights you have gained through your work.
- On a very general level, just note that basically every claim you make needs some backing from scientific literature.

Writing your thesis

Writing up your thesis is the final step of your BSc project, either after you have done all the practical work, or at least a huge portion of it.

Your thesis can be in English or German. English is preferred, as almost all of the relevant literature is in English and as it gives you the opportunity to work on your English writing skills. English language and grammar will **not** be graded (unless your thesis is totally incomprehensible, which however can similarly happen in German). But make sure you use spell-check and grammar-check to improve your writings, and don’t forget to ask a colleague or friend to have a look at your thesis before submitting.

Writing is hard and science is often harder to communicate. It usually requires lots of practice and many revisions. There are lots of approaches to writing, many tricks and suggestions. What we list here is a diverse set of sources that is meant to help you communicate your thoughts, approaches, and results in the best possible way. This list of suggestions is not meant to be complete, but rather a good starting point, collected from different communities.

- A standard reference is "The Science of Scientific Writing" by George D. Gopen and Judith A. Swan. While it addresses scientific writing in general, many principles are applicable to computer science as well. http://engineering.missouri.edu/civil/files/science-of-writing.pdf
- While very active in the visualization community, Tamara Munzner has written several fairly helpful and very detailed suggestions on how to improve your writing, from general approaches (see http://www.cs.ubc.ca/~tmm/policy.txt) to very detail-oriented common pitfalls (see also http://www.cs.ubc.ca/~tmm/writing.txt). Following these guidelines, will also help to improve your English writing skills. A concise writeup is her paper on writing pitfalls (see http://www.cs.ubc.ca/labs/imager/tr/2008/pitfalls/). While the 'Type pitfalls' as well as 'Visual encoding pitfalls' might be specific to visualization research, we believe that the remaining pitfalls are applicable to a large set of computer science writing rookies.
- Some German literature on writing guidelines that comes highly recommended include:
  - Karmasin, Ribnig: „Die Gestaltung wissenschaftlicher Arbeiten: Ein Leitfaden für Seminararbeiten, Bachelor-, Master-, Magister- und Diplomarbeiten sowie Dissertationen"
  - Krajewski: „Lesen Schreiben Denken: Zur wissenschaftlichen Abschlussarbeit in 7 Schritten" (available in the library)