Writing a Computer Science Thesis

Tobias Pfandzelter

Mobile Cloud Computing Research Group
TU Berlin & Einstein Center Digital Future
Berlin, Germany
tp@mcc.tu-berlin.de

January 31, 2022

Abstract

Writing a computer science thesis is a considerable challenge for students. In this text, I give some tips and structure to write a great thesis. I will go over the research process, finding a topic, writing an exposé, and thesis structure. At the end, I include some tips on researching and writing.

1 Introduction

If you're reading this, you might be about to start a computer science (or *Information Systems Management*, *ICT Innovation*, etc.) thesis at *TU Berlin*, maybe even at the *Mobile Cloud Computing* research group. A thesis, whether it is a bachelor's or master's thesis[1], is essentially a research task that you complete yourself. This can be daunting, but you're not alone: Since completing a thesis is necessary to complete a degree at TUB, you can learn from your previous students' experiences.

In this article, I will go over some of those experiences and lessons in order to help you complete your thesis successfully. Specifically, I will address the following questions:

- What is a thesis? How is it different from a project or seminar? (Section 2)
- How do I find a topic for my thesis? (Section 3)
- How do I write an exposé? Why should I even do such a thing? (Section 4)
- How do I structure working on my thesis? (Section 5)

[1] See [10] for a discussion on why it's *master's* and not *masters*.

Additionally, I will provide tips on research (Section 6) and writing (Section 7) as well as some helpful further resources (Section 8) before concluding in Section 9. If you're not a computer science student, you might still find some information in this article useful – just keep in mind that I make some assumptions that might not apply to you.

There are two things I specifically won't go into: First, this article will not cover the regulatory aspects of writing a thesis at TUB, including time limits, how to submit the thesis, etc. These things change all the time, and it doesn't make sense for me to just repeat information that you can find in the *AllgStuPO* and your respective *StuPO* anyway. Instead, go and read the *AllgStuPO* and *StuPO* now if you haven't yet. Even if you have, it can't hurt to do so again as a refresher. Second, I will not go into detail on tooling to write your thesis. I will just say that you should use *(La)TeX* and not Word. As a computer science (or related field) student, you should not be intimidated by a command line and text editor. Our thesis template [5] should help you get started.

2 What is a Thesis?

Before we discuss *how* you write your thesis, we should first understand *what* a thesis actually is. I mentioned that a thesis is research work that you complete on your own. Obviously, that is different to *courses*, where you attend a series of lectures to learn about a topic and then write an exam at the end. It is also completely different from most *projects* that you do: In a project, you typically solve an *engineering problem*, i.e., you get a task and then work in a team (or on your own) to develop software that solves that problem. You might additionally write tests or benchmarks to evaluate your system, but you're still essentially performing engineering work.

There is some overlap between a thesis and *seminars* or *reading groups* that you may have attended. These are typically closer to actual computer science research, in that you discuss existing research. Similarly, in a thesis you tackle a *research problem*.

Research problems and engineering problems have a few things in common: Both often require you to program some software, both require you to write text, both can be near or far from real-world problems. One of the main differences between research and engineering is that in research, your text (or *manuscript*) is the focus, and it can be supported by your implementation. Engineering usually has a system or implementation as a result which may be supported by some text, like a documentation or report. This distinction is something you should keep in mind during the entire time you write your thesis. For example, if you wonder whether you should spend your limited time implementing a new feature or writing some additional text, the latter is likely to be more important.

One other thing to keep in mind is what writing a thesis is actually about: Your goal is not to advance research (although that can be a nice side effect) or to prove how smart you are or how well you can come up with new ideas

(although both help) – the goal of a thesis is to show that you can conduct research properly, that you have the "research maturity".

At the core of a research problem there is usually a *research question*. This research question can have different *interrogative words* that will dictate the direction of your thesis: If it starts with *Why* or *What*, you usually want to investigate a certain phenomenon (e.g., *Why is the gcc compiler so slow when I compile the Linux kernel?* or *What is the overhead of compiling the Linux kernel in a virtual environment?*). On the other hand, if your question starts with *How*, the focus is more on developing something new (e.g., *How can we decrease the compile time of the Linux kernel?*).

3 Finding a Topic

Before we can go into how exactly a thesis is structured, we should first see how we can find a topic, i.e., how to determine your research question. To find a research question, you must start with a research area you're interested in. You can find your research area by considering what you learned about in lectures, projects, and seminars, or you do some initial investigations on what areas are currently of interest to the research community. Most research topics are given out by or created with help from your advisor. They might, e.g., have an ongoing research project that you can contribute with.

To find an advisor, consider the research groups where you have done some specialization. Check the profiles of the different members of the group and see if there is someone whose research area matches your preferences. When you write them, try to make clear what your research interests are.

4 Writing an Exposé

Before you can start working on your actual thesis, it is common practice to write an *exposé*. Think of your exposé as an outline to your thesis, covering the most important points:

1. present the research area you are working on
2. explain your research question
3. outline your approach to answer your research question

An exposé should typically only cover a A4 page and can be written in three to five paragraphs, your name, and a possible thesis title. Your exposé should not include footnotes or references as it is only meant for you, your advisor, and possibly the professor (who will be the official examiner of your thesis). The expectation is that you already have an understanding of the research area and done some initial research. However, you may not fully be aware of existing approaches to the research questions, as that will be part of the research process during your thesis.

You can derive the title from your research question, e.g., turning *Why is compiling the Linux kernel so slow?* into *A Performance Analysis of Linux Kernel Compilation*. The first paragraph of your exposé should cover your research area and provide some context for further explanations. The second paragraph introduces your research problem. Try to be as detailed as possible to avoid confusion during the work on your thesis. Be sure to also include motivation for your question: why is it important that someone takes the time to work on this? The third to fifth paragraphs should focus on what you actually plan to do in the thesis, it's basically your work plan. You can read more about what you *should* do in Section 5, but for now keep in mind the two questions that you should answer:

1. What is the approach that you plan to take to give an answer to the research question?
2. How will you evaluate your approach?

There will typically be a few iterations of the exposé before it is in an acceptable state. The goal of this process is to have you and your advisor have a common understanding and perspective on your research problem. This requires discussion about different perspectives, so don't worry if it takes a few days. However, the time between iterations is entirely up to you.

5 Structuring your Thesis

In this section, I'd like to cover two things, namely how you should structure your time and tasks when working on your thesis, and how the manuscript itself should be structured. I will cover both things together since the approach is identical – the best way to write your thesis is start at the beginning and finish at the end. Of course, that's not the entire story: during your research, you will uncover new knowledge and will have to re-write earlier parts of your manuscript, or you need to restructure some paragraphs. But the basic idea of working from start to finish remains. And what if I told you that you have already begun?

5.1 The Introduction

The very first part of any manuscript is the introduction, where you give context for your research, introduce your research question, and outline your work. Sounds familiar? That's because the introduction basically mirrors your exposé. It is supposed to give your reader an idea of what they can expect in your these, and it helps to set the scene for the text.

You should adapt the text of your exposé a bit instead of copying it directly. For example, you will now need to add references to your claims. It is also common to add a list of contributions, similarly to what I have done in Section 1.

5.2 Background

The next part of your research (and thesis manuscript) is the dreaded background. At this point, you should have done some initial research, read some papers, looked at some existing work in the field. Now, you should double down and fully understand (or at least be aware) the state-of-the-art in your research area. Follow the citation trail (e.g., by reading the papers cited by the papers you've read, reading the papers cited by that, etc.) until you get back to where you started. However, be smart with your time and don't spend too much on one obscure paper – the citation count is usually a good indicator on how a paper has aged (more on selecting the right resources in Section **??**). As an addition, make sure to keep some papers for your related work section (see Section 5.6)

In a research paper, the background section normally serves two tasks: First, it introduces the terminology and definitions you use in the rest of your paper. For example, there may be competing definitions on what a *kernel* is: the thing that runs your Linux or the thing that makes popcorn. This is your opportunity to set the record straight, ideally using some cites. Second, the background section lets you introduce the concepts you use in the rest of your paper. Every non-trivial piece of information should be mentioned here. It can be hard to decide what is considered trivial vs. non-trivial. Think about who reads your paper (e.g., your advisor) and try not to give too much information as that would make your paper boring[2]. As a rule of thumb, everything that you learned in compulsory lectures can be considered common knowledge in your field.

A background section in a thesis also serves another, indirect task: It shows the reader that you as a student have completely understood the research area you operate in. Conversely, if, during your writing process, you or your advisor notice that there is something missing from this section, you should go back into the literature and try to close that gap

5.3 The Approach

This section should be named according to its contents (e.g., *A Markov Model Approach to Linux Kernel Development*), but the basic idea is that you introduce your idea: What solves your research question? This section should be comparatively short, but in a research paper this part is often the most important one that everything revolves around. This somewhat differs in a thesis: Your grade will most likely not be determined by how "great" your idea is, as you will have discussed it in detail before with your advisor – remember that this is not an engineering report. You will be judged mainly on your execution of the thesis. Don't feel pressured to have something novel or surprising here, it's often best to stick with what you and your advisor came up with together.

The idea you present here can take many forms: it can be a study design (e.g., a benchmark that you want to perform to measure overheads), a system

[2]I will cover some further information on Dos and Don'ts of writing in Section 7

architecture (e.g., for a new database), or an algorithm (both an entirely new algorithm to solve a hitherto unsolved problem or the application of an existing algorithm to a new problem). Getting this section right can be difficult, especially as there can be some overlap with the next one. Here are some tips for mistakes that students commonly make on their first try:

Don't include implementation details When you write a systems paper, let's say your idea is to write a new kind of database, don't include your implementation of this system here. For example, for the design of your database system, it doesn't matter whether it was written in Java or C++. What matters is its overall design. Again, remember that your research question is not an engineering problem. One exception is if you require a certain technology for your approach, e.g., you investigate *gcc* and your approach doesn't extend to other compilers.

Don't include *how* you arrived at your solution To find an approach for your research question, you will often start with a simple one and iterate on that to improve it. While this is a useful technique, it doesn't translate well to your manuscript. What you will end up with is not one approach but multiple, which makes it hard for your reader to understand quite what is going on. There can be some cases where you want to compare different approaches to the same research question, but that slightly changes your research question: Instead of a *How should I solve it?*, the question then is *Which of approaches A, B, and C is the best one?*. Your approach section should then describe how you plan to execute your comparison study, while the different approaches are in your background. Focussing on a single approach and presenting that clearly makes it much easier for your readers to follow along and gives your thesis a clear message.

Don't mix evaluation and approach You will want to evaluate your approach in your thesis (and we'll get to that), but your evaluation should be in a different section. This includes the design of your evaluation, e.g., how you want to measure your approach is good.

Do include a picture A good picture can help a lot in getting your message across, even more so if your approach includes experiment designs or systems architecture. Don't overdo it, though!

5.4 Evaluation

After you have presented your approach, it's time to show *why* it's so great with an evaluation. An evaluation can take many forms, from a formal proof of an algorithm to a benchmark of an implementation. Finding the best evaluation method is something to discuss with your advisor because it depends on your

research question and what you have done so far. Here are some ideas for what you can do:

Simulation A simulation is a good way to evaluate your approach efficiently in a controlled environment. You can simply write yourself a small simulation environment (it's often not necessary to use any existing framework) and plug in a dataset or algorithm. You will likely need some existing approach or other baseline to compare your new approach to. Be sure to determine some metrics that you want to measure so you get meaningful results.

Implementation An implementation of a system can be helpful as well, although it is mostly coupled with some tests or benchmarks that show that the system does what it's supposed to do or improves an existing approach. Keep in mind that benchmarks are only meaningful when comparing to something that already exists. Include an overview of how you implemented your system but don't go into too much detail when it's not necessary.

Formal Proof You can include a formal proof for your approach if that is the best way to show its correctness.

In all cases, make sure to introduce your study design explicitly at the beginning. For simulations and benchmarks, be smart about what and how you measure: State your expectations and try to think like an adversary by coming up with scenarios that might break your approach. Also, when you notice that a result does not match your expectations, try to explain it and run additional experiments [4].

5.5 Discussion

Not every research paper includes an explicit discussion section, but having a discussion can elevate your research tremendously. A discussion is *not* an explanation of your results – this should be part of your evaluation. Instead, a discussion critically evaluates your approach and evaluation. Most things we do in computer science are not perfect: There will be edge cases, limitations, scenarios where other approaches are better, etc. This section is your opportunity to acknowledge the weaknesses of your thesis and discuss why or why not they matter.

You might be asking yourself: "Why would I talk about things that don't work?!" Because if you don't, your examiner will. You have a unique opportunity to anticipate the comments that your examiner (or a peer reviewer) might make about your approach. This shows that you are fully aware of these limitations and edge cases, and it lets you weaken the impact of those kinds of comments. Being able to look at your own work critically is also a testament to your scientific abilities. Remember that the goal of a research project is not to sell something but to advance your field of research.

5.6 Related Work

In a related work section, you discuss other work in your area that aims to solve your research problem or related research problems. The goal of this is not to discredit these papers but rather to show how your research builds on them and where you have used them for inspiration. Keep it nice!

If you've read a few research papers, you might be wondering why I mention this section near the end. You will often find it in the front of the paper, e.g., after the background section. That can make sense for some research questions, but it will often be boring for a reader to sit through pages of related work before understanding what your idea actually is. Instead, put it at the end where the reader has a full understanding of your approach, its effects (through evaluation), and weaknesses (through your discussion).

Go about it one paper at a time. Mention the paper explicitly (e.g., "Doe et al. [10] do this and that..."), and summarize the main idea behind it. Then, explain why this is good or bad. Sometimes, multiple papers have a categorically different approach (e.g., centralized vs. decentralized) than yours. In that case, group those papers and explain the advantages and disadvantages using an example. Remember that all of these approaches do actually have advantages – otherwise, they wouldn't exist[3].

Some students find it difficult to write this section, and I will sometimes read theses that only have a few sentences here, mentioning no more than two or three papers. Often, that means that these students have not done their research properly. If you can't find anything that answers your research question specifically, broaden your search area. Maybe there is a paper that asks your research question and doesn't provide a solution. Maybe a similar research problem is also posed in a different field. Possibly, some papers you mention in your background section belong here instead. A short related work section might seem to be an indicator that your thesis is especially good because it solves a problem that no one else has looked at before and that you were now able to solve with your research superpowers – instead, it generally means that you don't quite know what you're talking about.

5.7 Conclusion

Finally, it's time to conclude your thesis. Here, give a brief overview of everything you have done. Some research papers also use this to provide an outlook on future work you plan to do – in most cases, students don't continue to work on their thesis projects, though.

5.8 Other Sections

The sections I have given are a general guideline on how to perform your research and how to write everything down. You can vary the overall structure of your thesis a bit if required, but you will hardly find any published work that

[3] Unless you iterate on them, in which case you should also explicitly mention the similarities

dramatically deviates from this common structure. Nevertheless, you will likely need some additional components in your manuscript:

Abstract An abstract (and the German equivalent "Zusammenfassung") is a short summary of your work. A reader who does not yet know whether they want to read your paper should be able to gather the most important information about it from reading just the abstract.

Bibliography The bibliography (or references) list all the references of your paper. There are different reference styles you can use, but you should probably stick to what your template dictates.

Table of Contents Unlike research papers, theses usually also provide a table of contents that make it possible to quickly jump to the section you want.

List of Figures, Tables, Abbreviations Some templates include a list of figures, tables, or abbreviations that are generated automatically. While it's great to have one more page in your PDF, use those lists only when necessary, i.e., when you manage lots of tables or figures.

Acknowledgements If you want, you can include an acknowledgements section. In a research paper, this is usually used to indicate conflicts of interest, funding, or people who have helped you. If you can only think of your advisor to put here, leave this out and thank them in person instead. As your thesis will not be published as-is, there is no reason to include it anyway.

6 Research Tips

Having the right approach to research is one of the most important parts of writing a good thesis. The reader should get the feeling that you know what you are talking about, have a full understanding of your research area, and can critically review research papers and articles before you reach a conclusion. I can't teach you how to conduct your research here (this is a skill you should have honed in your studies), but I want to give you some tips and guidelines.

First, I want to emphasize the result of the distinction between engineering and research problems on how you handle your research. In an engineering documentation or other project report, you will feel inclined to *defend* what you have done, e.g., you might try to find a study that shows the downsides of a decentralized approach to defend that you have come up with a centralized approach. That makes sense when you want to sell something, but an important pillar of science is *transparency*. To clearly show what your idea can and can't do, give a holistic overview of state-of-the-art research, including competing ideas. In most cases, it is best to finish your research before you begin developing your idea. Of course, it might be necessary to go looking into certain aspects a

bit deeper once you have some experience developing your idea; then, however, don't be afraid to rethink aspects of your work when you learn more.

An important aspect of doing research is finding and evaluating publications – not all papers are created equally! Before you read a text, try to answer a few questions:

Is this a peer-reviewed text? Officially published texts, such as conference papers and journal articles, will receive a "peer-review", where other researchers in the field give feedback on the text and decide whether it meets certain scientific standards. To find out whether a text has been peer-reviewed, see if it is published in a proper journal or conference proceedings. Preprints of papers and articles are sometimes uploaded to servers such as arXiv [1] before they go through peer-review to have them available earlier – when you encounter a preprint, check who uploaded it and whether a published version exists. Some texts, such as industry white papers, blog posts, websites, and software documentation, do not go through peer-review. Use these texts only as additional references when absolutely required and try to find peer-reviewed alternatives.

Is the venue reputable? The peer-review process is usually part of publication in venue, i.e., a conference, magazine, or journal. Not all venues are equally reputable: Some conferences accept only a small fraction of submitted papers and have high standards, while other journals might publish anything as long as the authors pay enough. If you are not familiar with a venue, look at the institutions behind it: If it is held or sponsored by *ACM*, *IEEE*, or other well-known organizations, there is a good chance that certain standards are met. For sake of completeness, *Wiley* and *Elsevier* are trustworthy as well. On the other side of the spectrum, *predatory publishers* publish everything they can get their hands on if the authors pay their high fees. Note that some journals and conferences don't work with a publisher directly, making it a bit harder to determine their quality levels. In this case, look at *who* publishes with them (more on determining trustworthiness of authors below). Another thing you can look at are conference rankings, but this can be a bit hit-and-miss.

Are the authors trustworthy? Finally, see if you find the authors trustworthy. Are they affiliated with an institution you know and trust? Find out if the authors have previously researched in this area. Usually, the last listed author is the most senior researcher, e.g., the professor – find out whether they are notable or not. You can also check out the references in the manuscript: Do the authors only cite their own work or do they themselves follow good scientific practices? Finally, you might think that a paper by an industry group such as Microsoft Research will be biased, e.g., hiding information to make the company look better. However, you can trust the peer-review process and the authors' integrity (especially at such a reputable institution).

If you answered, "Yes" to all three questions, does that mean you can just take everything in the text at face-value? Well, no. There can always be mis-

takes in a paper, or things that the authors cannot know when they write a text (e.g., how a technology will develop over the years). Conversely, even if you have answered "No" to one of the questions above, you might still have a good paper in front of you. Do keep an open mind when you read any text you encounter.

7 Writing Tips

Finally, here is a non-exhaustive list of writing tips for your thesis. I highly recommend that you adapt the writing style of the papers you read (with the notable exception of [8]).

Use a spell-checker There is no argument against this. Spell-checkers exist, they are free to use, they're available on any platform. Trust them to do their job correctly, they know the English language better than you do. I recommend [13] for LaTeX and VS Code, but you may choose a different one based on your preferences.

Use a spell-checker I'm not sure if you've heard me the first time. It's literally free! It takes a minute to set up! There is no justification for having your advisor exposed to poor language, it can only hurt you! Some advisors may consider this an insult and a waste of their time.

Be direct in your wording Don't use passive wording but be direct instead. Use present tense (or past tense if you have to, but be consistent[4]) everywhere and describe your actions clearly.

Be professional (i.e., don't write like I do) Use professional language, this is an academic environment, after all. Do not use short forms ("do not" instead of "don't") or other colloquial language, such as "like". Additionally, don't address the reader directly, as I do here. There is usually no need for this.

Use the Oxford comma Use the Oxford comma, i.e., a comma after "and" where applicable [2]. This makes your text much more readable as it removes ambiguities.

Use "We" instead of "I" This is hard to wrap your head around for the first few chapters you write, but its common practice to use the first-person plural instead of singular when referring to the author(s), even when it's just a single person (like you, writing your thesis). There is no reason for a reader to care whether one or more people wrote a text, so why confront them with that information in every paragraph? Besides, no research is ever the result of the

[4] Please, please be consistent.

work of only a single person – even when they're not listed as an author, your advisor or peers helped you in your thesis.

Avoid weasel words Weasel words such as "much", "very", or "significantly" don't help in your texts. Instead of "We were able to improve performance significantly.", write "We improved read latency by 87%.". This would make for a terrible novel, but is much better for a thesis.

Paraphrase, don't copy text When you cite a text, don't copy parts of the text, such as a definition. Instead, paraphrase the key message of that text and integrate it into your own text.

Please use a spell-checker Please. Pretty please.

8 Further Reading

I can't cover everything in this article, so I do want to draw your attention to a few further resources. In a blog post [9], Philipp Leitner gives some valuable tips for writing a software engineering thesis. Simon Peyton Jones of Microsoft Research shares some valuable insights into writing technical papers [7] and giving research talks [6]. You can find additional writing tips in [3, 11, 12].

9 Conclusion

Writing a computer science thesis is hard, especially when you do it the first time. Having some research experience from seminars can help, but there are a lot of firsts for each student. I hope that this text gives you some structure for your upcoming thesis and that the tips and hints help you along the way.

References

[1] *arXiv.org e-Print Archive*. Accessed: 2022-1-30. arXiv.org. URL: `https://arxiv.org/`.

[2] A. Edwards. *What Is the Oxford Comma (or Serial Comma)?* URL: `https://www.grammarly.com/blog/what-is-the-oxford-comma-and-why-do-people-care-so-much-about-it/`.

[3] M. Ernst. *How to write a technical paper or a research paper*. Accessed: 2022-1-30. URL: `https://homes.cs.washington.edu/~mernst/advice/write-technical-paper.html`.

[4] M. Handley. *Seems all I do is ask "why does the graph do that?" 1st year PhD: "Does what?" 2nd year PhD: "I wondered that too" 3rd year PhD: "Maybe it's X?" Final year PhD: "I knew you'd ask that, so I ran this new experiment, and here are the extra graphs that explain it!"* Accessed: 2022-1-27. URL: https://twitter.com/markjhandley/status/1323303318832226304.

[5] J. Hasenburg and T. Pfandzelter. *TU Thesis Template – GitHub*. Accessed: 2022-1-27. URL: https://www.github.com/pfandzelter/tu-thesis-template.

[6] S. P. Jones. *How to give a great research talk*. Accessed: 2022-1-30. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/07/How-to-give-a-great-research-talk.pdf.

[7] S. P. Jones. *How to write a great research paper*. Accessed: 2022-1-30. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/07/How-to-write-a-great-research-paper.pdf.

[8] L. Lamport. "The part-time parliament". In: *ACM Transactions on Computer Systems* 16.2 (May 1998), pp. 133–169.

[9] P. Leitner. *Some Frequent Writing Tips I Give Software Engineering Thesis Students*. https://philippleitner.medium.com/some-frequent-writing-tips-i-give-software-engineering-thesis-students-da2acab30381. July 2021. URL: https://www.github.com/pfandzelter/tu-thesis-template.

[10] M. Maddox. *Masters Degree or Master's Degree?* Accessed: 2022-1-27. URL: http://www.dailywritingtips.com/masters-degree-or-masters-degree/.

[11] D. Patterson. *Dave Patterson's Writing Advice*. Accessed: 2022-1-30. URL: https://people.eecs.berkeley.edu/~pattrsn/talks/writingtips.html.

[12] H. Schulzrinne. *Common Bugs in Writing*. Accessed: 2022-1-30. URL: http://www.cs.columbia.edu/~hgs/etc/writing-bugs.html.

[13] J. Valentin. *LTeX – LanguageTool grammar/spell checking*. Accessed: 2022-1-30. URL: https://marketplace.visualstudio.com/items?itemName=valentjn.vscode-ltex.