Guidelines for writing the IT4BI Master Thesis

Master Thesis Documentation

Follow the instructions below to write your IT4BI master thesis document. As you will notice, writing the document is not a final step to be performed, but you need to systematically trace your progress. The suggested issues to be considered from the very beginning are as follows:

- At the beginning, decide with your advisor(s) / supervisor(s) a way to track and record your progress. You already have some experience on how to document from previous courses. Some things to decide at this point:
    - The on-line repository to share with your advisor(s) / supervisor(s),
    - Structure of the repository,
    - When and how to update and maintain the repository.

- The main elements you need to record your progress on during the whole semester are:
    - Produce comprehensive summaries about the state of the art you searched for. It will be absolutely impossible to do it later, as you will need to go through all the papers again. Thus:
        - Systematically gather the conclusions from each paper,
        - Come up with final conclusions embracing all the papers you read,
        - Iterate in a systematic way.
    - After reading some of the seminal papers in the domain of your thesis and familiarizing yourself with the field of study, stop and think about the main goal of your thesis. It is time to try to write down some motivation for the first time. Writing is a great way of organizing your thoughts and will help you pinpointing weaknesses in your motivation:
        - This should include a clear research question,
        - A clear contextualized motivation supporting the need for such question, and
        - Preferably quantitative and possibly qualitative criteria to measure the quality of your solution (*should be faster than…*, *more accurate than…*, *allow to more easily express*…).
    - Track the overall idea of your solution to the research question. This includes:
        - A conceptualization of the solution provided,
        - A methodology to be followed to implement such solution.
    - As a next step, you should formalize the artifact designed. This includes:
        - Formalisms (such as pseudo-coude algorithms[i]),
        - Conceptual models such as, for instance, ER diagrams, dimensional fact models, UML diagrams, workflow models.
        - Theoretical proofs.
    - Keep track of your "engineering" solution. Clearly motivate design choices. This will include:
        - Tools selected (why did you select a specific tool; what are the alternatives),
        - Architecture to be deployed (are there alternative architectures?),
        - Implemented algorithms (complexity of the algorithms).

- Write down the rationale behind the battery of experiments you will validate your artifact with. This should be aligned with the research question and should back up your claims. Thus:
    - Design a battery of experiments and show their representativeness,
    - Discuss the results and draw general conclusions (avoid statements such as: *algorithm A is faster than B on datasets D1, D2, D3, and D4 and slower on D5, D6, and D7*. Better: *In our experiments we observe that Algorithm A outperforms algorithm B on datasets of high cardinality. This difference in performance is explained by algorithm A having a lower memory footprint than algorithm B, as established in Chapter X. One exception, however is …*)
    - If necessary, support your conclusions with additional experiments (wrong: *A possible explanation for the superior performance of Algorithm A is that due to the heuristic it is using it may skip large parts of the search space*. Much stronger: *One hypothesis for the superior performance of Algorithm A is that due to the heuristic H it is using it may skip large parts of the search space. To test this hypothesis, in Figure Y we have plotted the size of the search space explored by algorithm A with H (blue line) and without H (red line). As can be seen in this figure, …*)

During the semester, you should not be concerned about giving an overall view to the elements you track. But be sure that the important and relevant factors are there. For the rules to be followed in the dissertation to be submitted as result of your IT4BI MSc thesis please contact the local coordinator at your specialization university.

---

[i] „Pseudo-code“ must be understood as code „for human consumption“, so should be intuitive:

For example, **No**:

```
m=-1000000
for j=0 to len(S)-1:
    v=f(S[j])
    if v<m then m=v
end for
```

Yes:

```
m = min{ f(s) | s\in S }
```

Maybe:

```
let m be the minimal f(s) for all s in S
```

Pseudo-code is restricted in length, not involve low-level considerations such as pointers, memory allocation (unless of course the handling of low-level memory considerations is the whole point of the algorithm) and use variable names that are connected to their intended use.