University of Magdeburg

School of Computer Science

Bachelor Thesis

[The Title of the Thesis]

Author:

[Forename] [Surname]

[Month 13, 2014]

Advisors:

Prof. *[Name]*

Department of *[...]*

***[Surname]*, *[Forename]*:**
[The Title of the Thesis]
Bachelor Thesis, University of Magdeburg, *[2014]*.

Abstract

[...]

Contents

List of Figures

List of Tables

List of Code Listings

1. Introduction

[...]

Goal of this Thesis

[...]

Structure of the Thesis

[...]

2. Background

[...]

3. Example Chapter

This chapter gives you some examples how to include graphics, create tables, or include code listings. But first, we start with a short description how you can efficiently cite in LaTeX. The following footnote shows you how to reference URLs and where this document is available online.[1]

3.1 Citation

There are several types of literature. The most citations are workshop and conference papers. Please use the inproceedings-tag for those citations (e.g., [KAK09]). You should have short-hands for workshop and conference names to be sure the naming is consistent and uniform (see our BibTeX files how to do that).

Slightly different are articles published in journals (e.g., [KG06]). Make sure you that the volume and number-tags are present and that no inproceeding is tagged as article or vice versa.

You might want to take a look at the example BibTeX file to find out how to cite books [CE00], technical reports [KCH+90], websites [CDT09], PhD theses, or master theses [Beu03, Ros09].

3.2 Formulas

There are different types of mathematical environments to set formulas. The equation $E = m \cdot c^2$ is an inline formula. But you can also have formulas at a separate line (see Equation 3.1).

$$P = \big(\mathcal{A} \Rightarrow (\mathcal{B} \Leftrightarrow \mathcal{C}) \wedge (\mathcal{B} \Leftrightarrow \mathcal{D})\big) \wedge (\mathcal{B} \Rightarrow \mathcal{A}) \wedge (\mathcal{C} \Rightarrow \mathcal{A}) \wedge (\mathcal{D} \Rightarrow \mathcal{A}) \tag{3.1}$$

[1] http://www.ovgu.de/tthuem

If you need multiple lines that are aligned to each other, you might want to use the following code.

$$
\begin{aligned}
& \text{GraphLibrary} \\
\wedge\ & (\text{GraphLibrary} \Rightarrow \text{Edges}) \wedge (\text{Edges} \vee \text{Algorithms} \Rightarrow \text{GraphLibrary}) \\
\wedge\ & (\text{Edges} \Leftrightarrow \text{Directed} \vee \text{Undirected}) \wedge (\neg\text{Directed} \vee \neg\text{Undirected}) \\
\wedge\ & (\text{Algorithms} \Leftrightarrow \text{Number} \vee \text{Cycle}) \\
\wedge\ & (\text{Cycle} \Rightarrow \text{Directed}).
\end{aligned}
$$

3.3 Graphics

In Figure 3.1, we give a small example how to insert and reference a figure.

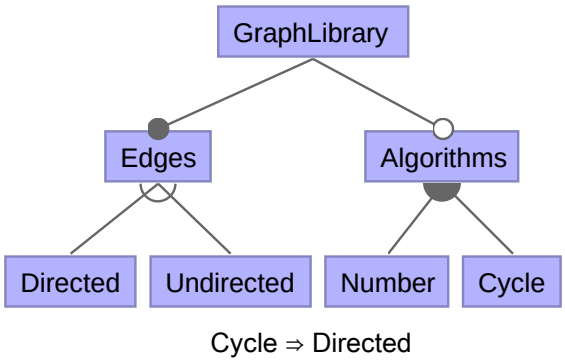

Figure 3.1: A feature model representing a graph product line

3.4 Tables

Table 3.1 shows the result of a simple tabular environment.

Group Type	Propositional Formula
And	$(P \Rightarrow C_{k_1} \wedge \ldots \wedge C_{k_m}) \wedge (C_1 \vee \ldots \vee C_n \Rightarrow P)$
Or	$P \Leftrightarrow C_1 \vee \ldots \vee C_n$
Alternative	$(P \Leftrightarrow C_1 \vee \ldots \vee C_n) \wedge \text{atmost1}(C_1, \ldots, C_n)$

Table 3.1: Mapping a feature model to a propositional formula

3.5 Code Listings

In Listing 3.1 on the next page, we give an example of a source code listing.

```
1  class A extends Object {
2      A() { super(); }
3  }
4  class B extends Object {
5      B() { super(); }
6  }
7  class Pair extends Object {
8      Object fst;
9      Object snd;
10     Pair(Object fst, Object snd) {
11         super(); this.fst=fst; this.snd=snd;
12     }
13     Pair setfst(Object newfst) {
14         return new Pair(newfst, this.snd);
15     }
16 }
```

Listing 3.1: Java source code

4. Evaluation

[...]

5. Related Work

[...]

6. Conclusion

[...]

7. Future Work

[...]

A. Appendix

[...]

Bibliography

[Beu03] Danilo Beuche. *Composition and Construction of Embedded Software Families*. PhD thesis, University of Magdeburg, Germany, 2003. (cited on Page 5)

[CDT09] The Coq Development Team. The Coq Proof Assistant. Website, September 2009. Available online at http://coq.inria.fr/; visited on November 9th, 2009. (cited on Page 5)

[CE00] Krzysztof Czarnecki and Ulrich W. Eisenecker. *Generative Programming: Methods, Tools, and Applications*. ACM Press/Addison-Wesley, 2000. (cited on Page 5)

[KAK09] Christian Kästner, Sven Apel, and Martin Kuhlemann. A Model of Refactoring Physically and Virtually Separated Features. In *Proceedings of the International Conference on Generative Programming and Component Engineering (GPCE)*, pages 157–166. ACM, October 2009. (cited on Page 5)

[KCH+90] Kyo C. Kang, Sholom G. Cohen, James A. Hess, William E. Novak, and A. Spencer Peterson. Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, 1990. (cited on Page 5)

[KG06] Cory J. Kapser and Michael W. Godfrey. Supporting the Analysis of Clones in Software Systems: A Case Study. *Journal of Software Maintenance and Evolution*, 18(2):61–82, 2006. (cited on Page 5)

[Ros09] Malte Rosenthal. Alternative Features in Colored Featherweight Java. Diplomarbeit, University of Passau, Germany, July 2009. (cited on Page 5)

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Magdeburg, den *[. . .]*