HTWK
Hochschule für Technik,
Wirtschaft und Kultur Leipzig

IM Fakultät
Informatik und Medien

Abschlussarbeit
zur Erlangung des akademischen Grades

Bachelor of Science (B.Sc.)

im Studiengang Informatik
der Fakultät Informatik und Medien

Erfüllbarkeit und Optimierung von Pressenplanungsproblemen mit Constraint-Systemen

Julian Bruder
Leipzig, den 20. August 2024

Erstgutachter: Prof. Dr. Johannes Waldmann
Zweitgutachter: Dr. Sebastian Möbius

Zusammenfassung

(Kopie aus Themebeschreibung, Anpassung notwendig)

Die Pressenplanung von Holzbalken aus Bauteilen erfordert eine fein abgestimmte Konfiguration, um die Pressen optimal auszulasten und den Verschnitt der Balken zu minimieren. Im Rahmen der Bachelorarbeit *Erfüllbarkeit und Optimierung von Pressenplanungsproblemen mit Constraint-Systemen* soll dafür eine prototypische Lösung mithilfe der Methoden der Constraint-Programmierung erarbeitet werden. Dazu sollen die Constraints der Domäne zunächst mathematisch modelliert und anschließend in einer konkreten Repräsentation des Modells implementiert werden.

Dabei soll ein Programm entwickelt werden, welches aus den textuellen Eingaben der Pressen- und Bauteilspezifikationen ein Constraint-System erzeugt, das anschließend von einem externen Constraint-Solver gelöst wird. Die Lösung des Solvers soll folglich vom Programm in eine textuelle Ausgabe umgewandelt werden. Das Programm soll kein grafisches Interface umfassen, sondern lediglich die aktuell existenten Schnittstellen der Ein- und Ausgaben implementieren. Weiter soll das Programm so strukturiert werden, dass sowohl SAT (Bitblasting) als auch SMT (QF_LRA, QF_LIA) zur Erzeugung des Constraint-Systems genutzt werden können. Als Programmiersprache für das Programm wird Haskell gewählt.

Das Ziel für die prototypische Lösung des Problems der Pressenoptimierung ist die korrekte (optimale) und *schnelle* Lösung bekannter Testfälle. Sonderfälle, wie beispielsweise Pressen mit Balken unterschiedlicher Länge, sollen vorerst nicht betrachtet werden.

Auf Wunsch von Germanedge kann der Text der Arbeit so gestaltet werden, dass in der Arbeit genannte Anwendungsfälle nur abstrakt beschrieben und Testfälle bereinigt werden. Die Arbeit soll dabei den gesamten Prozess von Anforderungsanalyse über Entwurf und Realisierung bis hin zum Test dokumentieren.

Inhaltsverzeichnis

1 Einleitung **1**

2 Problem Pressenplanung **1**
2.1 Anforderungen . . . 1
2.2 Modellierung . . . 1
2.3 Optimierung . . . 1

3 Satisfiability Modulo Theories **1**
3.1 Einführung in das Erfüllbarkeitsproblem SMT . . . 1
3.1.1 Grundlagen Prädikatenlogik . . . 1
3.1.2 Sprachstandard SMTLib2.6 . . . 1
3.1.3 SMT-Logiken und Solver . . . 1
3.2 SMT-Kodierung von Anzahl-Constraints . . . 1
3.3 Ergänzung um Optimierungsziele - Optimization Modulo Theories . . . 1

4 Implementierung in Haskell **1**
4.1 Funktionale Programmierung mit Haskell . . . 2
4.2 Hasmtlib - Haskell-Bibliothek für SMTLib . . . 2
4.2.1 Ergänzung um Anzahl-Constraints . . . 2
4.2.2 Ergänzung um Incremental Refinement . . . 2
4.2.3 Ergänzung um Observable-Sharing . . . 2
4.3 Kodierung des Pressenplanungsproblems . . . 2

5 Auswertung und Laufzeitmessung **2**
5.1 Vergleich mit realistischem Timeout . . . 2
5.2 Vergleich mit liberalem Timeout . . . 2

6 Zusammenfassung **2**

Appendix **3**
A Beispiel 1 . . . 3
B Beispiel n . . . 3

1 Einleitung

Germanedge - Unternehmen (wenige Sätze). Problem? SMT? Wo normalerweise Anwendungen SMT? Wie Anwendung hier? Beispiel mit Visualisierung und Kodierung. Alle Kapitel kurz erwähnen und in einem Satz erklären.

2 Problem Pressenplanung

Einleitung...

2.1 Anforderungen

Kurze Beschreibung der Problemdomäne. Was wird gepresst, wie, welche Schritte, ..., welche Ziele?

2.2 Modellierung

Mathematische Modelle für Erfüllbarkeit des Problems. Warum zwei Modelle? Größe des Problems (Unbekannte, Formeln)?

2.3 Optimierung

Erweiterung der Modelle um Optimierungsziele. OMT, Incremental refinement, ...

3 Satisfiability Modulo Theories

Einleitung...

3.1 Einführung in das Erfüllbarkeitsproblem SMT

Einleitung...

3.1.1 Grundlagen Prädikatenlogik

Erfüllbarkeit, Terme, Atome, Formeln, ...

3.1.2 Sprachstandard SMTLib2.6

Wie werden prädikatenlogische Probleme formuliert? Nur Kern erklären: Sorten, Variablen und Asserts

3.1.3 SMT-Logiken und Solver

Welche Logiken hier relevant? Was zeichnet diese Logiken aus? Wie performant lösen die Solver diese Logiken - Komplexität? Lösungsverfahren nur minimal erwähnen.

3.2 SMT-Kodierung von Anzahl-Constraints

3.3 Ergänzung um Optimierungsziele - Optimization Modulo Theories

4 Implementierung in Haskell

Erwähnen, dass Germanedge bereits Lösung hat. Aber: Heuristik - (optimale) Lösungen gehen möglicherweise verloren. Hier: Neuer Ansatz mit Garantie Vollständigkeit und Optimalität. Was ist Haskell und warum Haskell? Was ist Hasmtlib?

4.1 Funktionale Programmierung mit Haskell

4.2 Hasmtlib - Haskell-Bibliothek für SMTLib

Hier müssen besonders der polymorphe Formelbaum (Expr-GADT) und die dependent-like Singletons (SSMTSort) erklärt werden. Wird später gebraucht um zu verstehen, wie Pressenplanungsprobleme flexibel in verschiedenen Logiken kodiert werden (Programm polymorph in SMTSort).

4.2.1 Ergänzung um Anzahl-Constraints

4.2.2 Ergänzung um Incremental Refinement

4.2.3 Ergänzung um Observable-Sharing

Sprengt das den Rahmen? Hier müsste vorher viel erklärt werden: Repräsentation von Bäumen im Speicher in Haskell (DAG), Auszug Kategorientheorie - Morphismen (hier Paramorphismus), Vergleich zu Tseitin sinnvoll, aber bisher SAT gänzlich unerwähnt. Dann auch die Frage: Warum der Aufwand? Bringt das überhaupt was für den Anwendungsfall Pressenplanung? Wenn nein: Gehört das hier dann überhaupt rein?

4.3 Kodierung des Pressenplanungsproblems

Zwei Modelle erschweren hier möglicherweise Verständnis für den Leser. Für ein Modell entscheiden: Relationales Modell begründend wegwerfen, auf Laufzeitmessung verweisen. Ebenso Unterscheidung Incremental Refinement/OMT, OMT wegwerfen, weil Laufzeit. Geeignete Programmauszüge: Polymorphe Kodierung in Sorte (damit Logik), vereinzelt Datentypen und Constraints analog zu Modellierung.

5 Auswertung und Laufzeitmessung

Was haben wir für Testfälle? Woher kommen die Testfälle? Wie wurden die Testfälle ausgewählt? Sind sie repräsentativ für reale Probleme? Vorgehen Laufzeitmessung erwähnen - auch kurz IO? Maschinen-Specs nennen.

5.1 Vergleich mit realistischem Timeout

Time-Limit 5 Minuten, Vergleich Heuristik Germanedge vs SMT. Nur ein Modell oder beide? - Beide. Auch OMT? - Ja. Auswertung der Ergebnisse.

5.2 Vergleich mit liberalem Timeout

Wer findet bessere Lösung, wenn Timeout sehr liberal (24h?)? Hier werden Hardware-Specs ein Problem. Das muss auf Pool-PCs der HTWK gemacht werden. Darf da ein Binary von der Heuristik laufen?

6 Zusammenfassung

Appendix

A Beispiel 1

⋮

B Beispiel n