

Pflichtenheft

Beste Gruppe

20. November 2016

Inhaltsverzeichnis

1	Produktübersicht	4
2	Zielbestimmung	5
2.1	Musskriterien	5
2.2	Wunschkriterien	6
2.3	Abgrenzungskriterien	6
3	Produkteinsatz	7
3.1	Anwendungsbereiche	7
3.2	Zielgruppen	7
3.3	Betriebsbedingungen	7
4	Produktumgebung	8
4.1	Software	8
4.2	Hardware	8
4.3	Orgware	8
4.4	Produkt-Schnittstellen	8
5	Funktionale Anforderungen	9
5.1	C-Code Editor für Wahlverfahren	9
5.2	Editor für formale Eigenschaften	10
6	Produktdaten	11
6.1	Code-Editor „Rigtime“	11
6.2	Editor „Properties“	11
7	Nichtfunktionale Anforderungen	12
8	Globale Testfälle	13
9	Systemmodelle	14
9.1	Szenarien	14
9.2	Anwendungsfälle	14
9.3	Objektmodelle	14
9.4	Dynamische Modelle	14
10	GUI	15

11 Phasenverantwortliche	16
11.1 Pflichtenheft	16
11.2 Entwurf	16
11.3 Implementierung	16
11.4 Qualitätssicherung	16
11.5 Abschlusspräsentation	16
12 Glossar	17

1 Produktübersicht

Das Entwicklerprogramm „Concrete Rigging of Voting Procedures“ ermöglicht es, Wahlverfahren auf formale Eigenschaften zu prüfen. So kann ein in C definiertes Wahlverfahren wie z.B. die einfache Mehrheitswahl darauf geprüft werden, ob es bestimmte Eigenschaften erfüllt. Der Benutzer kann verschiedene Parameter der Wahl zusätzlich vorgeben.

Da eine vollständige Prüfung auf Wahleigenschaften sehr lange dauern würde, kommt ein Bounded Model Check zum Einsatz. Das verwendete Werkzeug CBMC wird dabei automatisch vom Programm angesteuert.

Der Benutzer bekommt schließlich eine Antwort des Programms, in der er bei Nichterfüllung der Eigenschaft ein Gegenbeispiel angezeigt bekommt. Sollte die Prüfung jedoch erfolgreich sein, bekommt der Nutzer ein Positivbeispiel präsentiert.

2 Zielbestimmung

Das vorliegende Programm überprüft in C geschriebene Wahlverfahren mit Hilfe von CBMC auf formale Eigenschaften. Das Programm akzeptiert optionale Wahlparameter und gibt das Ergebnis der Prüfung zurück.

Die GUI ist in vier Teilen angeordnet:

1. „Rigtime“: Code-Editor für Wahlverfahren in Programmiersprache C.
2. „Properties“: Editor für Spezifikation formaler Eigenschaften in abgespeckter C-Syntax mit speziellen Macros.
3. „Params“: Eingabe von Parametern einer zu analysierenden Wahl mit Anzahl der Wähler, Kandidaten und Sitzen.
4. „Rigplete“: Ausgabe der Prüfung.

2.1 Musskriterien

- Das Programm kann auf aktuellen Versionen von Windows und Linux-Betriebssystemen betrieben werden.
- Alle Abhängigkeiten (u.a. zu CBMC) werden mit dem Programm ausgeliefert.

Die Elemente der GUI sollen folgenden Kriterien genügen:

- „Rigtime“: Code-Editor mit der Möglichkeit zum Speichern, Speichern als und Laden.
- „Properties“: Editor mit der Möglichkeit zum Speichern, Speichern als und Laden. Die Eingabe soll überprüft werden, d.h. ob die Eingabe Macros in C-Syntax darstellt. Im selben Fenster soll es eine Eingabemaske für zusätzliche symbolische Variablen geben.
- „Params“: Option für eine graphische Eingabe der Anzahl von Wählern, Kandidaten und Sitzen.
- „Rigplete“: Das Ergebnis der Wahlanalyse wird angezeigt. Falls CBMC ein Gegenbeispiel zu einer formalen Eigenschaft gefunden hat, soll das Beispiel XXX graphisch XXX angezeigt werden.

2.2 Wunschkriterien

- Das Programm kann auf aktuellen Macs betrieben werden.

Die Elemente der GUI sollen folgenden Kriterien genügen:

- „Rigtime“: Der Code-Editor soll für die Programmiersprache C bieten:
 - Syntax-Highlighting
 - Fehler-Anzeige
 - Automatisches Einrücken
 - Auto completion
 - Widerrufen, Wiederherstellen
 - Tastatur-Shortcuts
 - Warnung vor nicht unterstützten Elementen der Programmiersprache
 - Codevorlagen
- „Properties“: Der Editor soll Syntax-Highlighting und eine Fehler-Anzeige für C bieten. Codeeingaben sollen komplettiert werden können.
- „Params“: Die Analyse der Wahl soll mit Hilfe eines Buttons abgebrochen werden können.
- Wahlergebnisausgabe: Im Fenster XXX „Params“ XXX kann ein Array mit Wahlstimmen eingegeben werden. Das Ergebnis dieser Wahl wird im Fenster „Rigplete“ angezeigt.
- Zusätzlicher SAT-Solver: Das Programm bietet eine Schnittstelle, sodass auch andere SAT-Solver als CBMC angesteuert werden können.
- XXX Weitere? XXX

2.3 Abgrenzungskriterien

XXX

3 Produkteinsatz

Das Programm überprüft Wahlverfahren auf ihre formalen Eigenschaften. Es richtet sich an Kunden, die ein Interesse an der Erforschung oder Entwicklung solcher Verfahren haben. Sie benötigen ein Grundverständnis der Programmiersprache C und Logik.

3.1 Anwendungsbereiche

- Universitärer Bereich
- Forschung

3.2 Zielgruppen

- Wahlforscher
- Softwareentwickler

3.3 Betriebsbedingungen

XXX welche gibt es? XXX

4 Produktumgebung

4.1 Software

- OS: Windows/Linux
- Softwareentwickler

4.2 Hardware

- PC

4.3 Orgware

XXX

4.4 Produkt-Schnittstellen

XXX bei Implementierung von Einsatz anderer SAT-Solver? XXX XXX allgemein: Ausgabe von Produktdaten? XXX

5 Funktionale Anforderungen

5.1 C-Code Editor für Wahlverfahren

/F10/ Darstellung aller für das Programmieren in C benötigten Charaktere

/F20/ Veränderung des dargestellten Textes durch Eingabe anderer Charaktere über die Tastatur wie in Notepad

/F30/ Speichern von erstellten Code in Dateiformat datei.c

/F40/ Laden und Darstellen von .c Dateien

/F50/ Automatisches Einrücken des codes in Schleifen und if-statements

/F60/ Code-Completion

- Automatisches Schließen von Klammern und Anführungszeichen
- Primitiv: Vorschlagen bereits im Code vorgekommener Wörter
- Intelligent: Durch Analysieren eines ASTs nur Vorschlagen der Wörter welche im Kontext Sinn ergeben.

/F70/ Syntax-Highlighting: Darstellung diverser Schlüsselwörter in anderen Farben als den Rest des Codes. Dies beinhaltet, ist jedoch nicht beschränkt auf:

- Typendeklaration (int, float, structs...)
- Kontrollflow-Konstrukte (if, else, while...)
- Kommentare

/F80/ Durch den User konfigurierbares Verhalten:

- Festlegen der Farben, welche beim Syntax-Highlighting verwendet werden
- Festlegen des verwendeten Fonts

/F90/ Anzeigen von Syntaktischen Fehlern im Code, welche durch einen Lexer oder Parser erkannt werden können:

- Verwendung von Schlüsselwörtern als Variablennamen
- Vergessene Semikolons am Ende von Anweisungen

/F100/ Reaktion auf typische Tastenkürzel (siehe 5.1)

s

Tabelle 5.1: Hotkeys und verbundene Operationen

Kürzel	Operation
Strg + c	Kopieren
Strg + x	Auschneiden
Strg + v	Einfügen
Strg + z	Zuletzt ausgeführte Aktion rückgängig machen
Strg + r	Zuletzt rückgängig gemachte Aktion erneut ausführen
Strg + s	Speichern
Strg + o	Öffnen
Strg + Leer	Anzeigen der Code-Completion Vorschläge

Tabelle 5.2: Makros zur Beschreibung formaler Eigenschaften

Makro	Effekt
FOR_ALL_VOTERS(E)	checkt ob E für alle Wähler gilt
FOR_ALL_CANDIDATES(E)	checkt ob E für alle Kandidaten gilt
FOR_ALL_SEATS(E)	checkt ob E für alle Sitze gilt
EXISTS_ONE_VOTER(E)	checkt ob E für zumindest einen Wähler gilt
EXISTS_ONE_CANDIDATE(E)	checkt ob E für zumindest einen Kandidaten gilt
EXISTS_ONE_SEAT(E)	checkt ob E für zumindest einen Sitz gilt
SUM_OVER_VOTER()	
SUM_OVER_CANDIDATES()	
SUM_OVER_SEATS()	

5.2 Editor für formale Eigenschaften

/F10/ Darstellung aller für das Programmieren in C benötigten Charaktere
 /F20/ Veränderung des dargestellten Textes durch Eingabe anderer Charaktere über die Tastatur wie in Notepad
 /F21/ Beschreibung formaler Eigenschaften als Vor- und Nachbedingung in abgespeckter C-Syntax
 /F30/ Bereitstellung von Makros zur Beschreibung der Eigenschaften (siehe 5.2)

/F40/ Bereitstellen symbolischer Variablen für Wähler, Kandidaten und Sitze
 /F50/ Bereitstellen von Operatoren für Implikation und Äquivalenz
 /F60/ Beliebige tiefe, lediglich von Hardware begrenzte, Schachtelung dieser Konstrukte
 /F70/ Syntax-Highlighting
 /F80/ Anzeigen von Syntaktischen Fehlern im Code
 /F90/ Code-Completion

- Auto-Vervollständigung der Makros
- Analyse des Codes und Anzeigen relevanter, bereits definierter Eigenschaften und symbolischer Variablen

6 Produktdaten

6.1 Code-Editor „Rigtime“

/D10/ Das Wahlverfahren ist als Methode „unsigned int voting(params)“ einer C-Headerdatei definiert und wird mit der Endung .h gespeichert.

6.2 Editor „Properties“

/D20/ Die formale Eigenschaft, derer das Wahlverfahren genügen muss ist als C-Datei definiert, die einmal die Methode voting(params) aus einer Headerdatei aufruft, und wird mit der Endung .c gespeichert.

7 Nichtfunktionale Anforderungen

8 Globale Testfälle

9 Systemmodelle

9.1 Szenarien

9.2 Anwendungsfälle

9.3 Objektmodelle

9.4 Dynamische Modelle

10 GUI

11 Phasenverantwortliche

11.1 Pflichtenheft

Justin Hecht

11.2 Entwurf

11.3 Implementierung

11.4 Qualitätssicherung

11.5 Abschlusspräsentation

12 Glossar

Abbildungsverzeichnis