Formale Grundlagen der Informatik 3

Prof. Dr. Christoph Walther / Visar Januzaj, Nathan Wasser
Technische Universität Darmstadt — Wintersemester 2011/12

Praktische Übung 3

Version 1 vom 09.12.2011

Diese Übung wird in der **KW 02 (09.–13.01.2012)** testiert. Reichen Sie dazu Ihre mit ✓eriFun 3.4 erstellte `.vf`-Lösungsdatei bis spätestens **Sonntag, 08.01.2012 23:59 Uhr** online online im Kursportal ein. Beachten Sie bitte bei der Abgabe die dort angegebenen Hinweise. Später eintreffende Lösungen können nicht akzeptiert werden.

Es gelten die gleichen Testattermine bzw. Praktikumsgruppen G1-G90 wie in der letzten Testatwoche.

Bei Fragen und Problemen können Sie neben den Poolbetreuungszeiten und Sprechstunden der Tutoren auch das Forum zur Veranstaltung nutzen.

Der Fachbereich Informatik misst der Einhaltung der Grundregeln der wissenschaftlichen Ethik großen Wert bei. Zu diesen gehört auch die strikte Verfolgung von Plagiarismus. Mit der Abgabe einer Lösung bestätigen Sie, dass Sie der alleinige Autor des gesamten Materials sind. Bei Unklarheiten zu diesem Thema finden Sie weiterführende Informationen unter
`http://www.informatik.tu-darmstadt.de/Plagiarism`
oder sprechen Sie Ihren Betreuer an.

Vorbemerkung

Das Davis-Putnam-Verfahren ist, wie auch das aus *FGdI2* bekannte Resolutionsverfahren, eine Methode, um die Unerfüllbarkeit einer aussagenlogischen Formel zu überprüfen. Die zu überprüfende Formel ϕ muss dazu in konjunktiver Normalform vorliegen: ϕ ist also eine Konjunktion („Und-Verknüpfung“) von Klauseln, wobei jede Klausel C eine Disjunktion („Oder-Verknüpfung“) von Literalen ist.

Eine Klausel C lässt sich durch eine endliche Menge von Literalen repräsentieren. Die Formel ϕ kann als endliche Menge S von Klauseln dargestellt werden. Damit hat ϕ die Form

$$\phi = \bigwedge_{C \in S} \bigvee_{L \in C} L \ ,$$

so dass wir ϕ mit der entsprechenden Klauselmenge S identifizieren können.

Das Davis-Putnam-Verfahren liefert für eine Klauselmenge S genau dann *true*, wenn S unerfüllbar ist. Grundlage für das Verfahren ist folgender Kalkül:

Sprache: endliche Klauselmengen; formal: $\mathcal{P}_{fin}(\mathcal{P}_{fin}(LITERAL))$ für die Menge $LITERAL$ aller Literale[1]

Regeln: Die einzige Regel des Kalküls ist die *Splitting-Regel*:

$$\frac{S[L] \quad S[\sim L]}{S} \quad \text{, falls } L \in C \text{ für ein } C \in S$$

Hierbei bezeichnet $\sim L$ das Komplement eines Literals L. $S[L]$ entsteht aus S, indem alle Klauseln C mit $\sim L \in C$ entfernt werden und L aus allen verbleibenden Klauseln entfernt wird.

[1] Für eine unendliche Menge M, ist $\mathcal{P}_{fin}(M)$ die unendliche Menge der *endlichen* Teilmengen von M.

Herleitungen: Eine *Herleitung* für eine endliche Klauselmenge S ist ein Baum, dessen Wurzel mit S und dessen Knoten mit endlichen Klauselmengen markiert sind, die durch Anwendung der Splitting-Regel entstehen.[2] Ein *Beweis der Unerfüllbarkeit von* S ist eine Herleitung für S, deren Blätter S' jeweils die leere Klausel $\emptyset$ enthalten.

Die Idee des Verfahrens ist, die Unerfüllbarkeit von S zu zeigen, indem gezeigt wird, dass S weder erfüllbar ist, wenn das Literal L mit *falsch* belegt wird, noch wenn L mit *wahr* belegt wird. Wenn L mit *falsch* belegt wird, werden alle Klauseln C mit $\sim L \in C$ erfüllt und können aus S gestrichen werden. Aus den verbleibenden Klauseln kann L entfernt werden, weil *falsch* das neutrale Element der Disjunktion ist. Das so entstandene Teilproblem ist in der Splitting-Regel mit $S[L]$ bezeichnet. Analog bezeichnet $S[\sim L]$ das Teilproblem von S, in dem L mit *wahr* belegt wird. Eine Klauselmenge S' mit $\emptyset \in S'$ ist trivialerweise unerfüllbar, da hierdurch die Formel $\phi' = \mathit{false}$ repräsentiert wird.

Anstatt mit *Mengen* (von Literalen oder Klauseln) zu arbeiten, verwenden wir *Listen*. Während die Repräsentation einer Klausel als Menge bereits die Idempotenz der Disjunktion berücksichtigt ($L \vee L \Leftrightarrow L$), kann eine Liste ein Literal mehrfach enthalten. Mittels der Prozedur $\backslash\backslash$ werden (z. B. bei der Implementierung der Splitting-Regel) *sämtliche* Vorkommen eines Literals aus einer Klausel entfernt.

Die Prozeduren $\models$ und $\models\!\models$ definieren die Semantik von Klauseln bzw. Klauselmengen. $\sigma \models C$ liefert genau dann *true*, wenn die Belegung σ die Klausel C erfüllt. $\sigma \models\!\models S$ liefert genau dann *true*, wenn die Belegung σ die Klauselmenge S erfüllt.

Die Prozedur `function` $\mathit{DavisPutnam}(S : \mathit{list}[\mathit{list}[\mathit{LITERAL}]]) : \mathit{bool}$ implementiert das Davis-Putnam-Verfahren. Nachzuweisen sind die Korrektheit und die Vollständigkeit des implementierten Verfahrens.

Praktische Übung 3.1 (Vorbereitung)

Bevor Sie mit der eigentlichen Bearbeitung beginnen, sollten Sie sich einen Überblick über die benötigten Prozeduren schaffen. Vermerken Sie im Comment Fenster[3] der folgenden Prozeduren was diese berechnen:

$\in$, $\in\in$, $\backslash\backslash$, $\sim$, $\models$, $\nabla\models$, $\models\!\models$, $\models\!\models$, split

Praktische Übung 3.2 (Allgemeine Hilfslemmata)

Sie sollten ein paar hilfreiche Eigenschaften von $\sim$ und $\nabla\models$ als Lemmata formulieren und beweisen.

(a) $\sim (\sim (L)) =$????

(b) Wenn $L = \sim (K)$, dann $K =$????

(c) Wie verhält sich L zu $\sim (L)$?

(d) Wie verhält sich $\nabla\models(\sigma, L)$ zu $\nabla\models(\sigma, \sim (L))$?

(e) Was gilt für $\sigma\models L$ und $\sigma\models \sim (L)$ wenn $\nabla\models(\sigma, L)$ gilt?

Praktische Übung 3.3 (Korrektheit des Davis-Putnam-Verfahrens)

Das Davis-Putnam-Verfahren ist genau dann *korrekt*, wenn bei Rückgabewert *true* tatsächlich keine Variablenbelegung σ, die alle in S vorkommenden Variablen belegt, die Klauselmenge S erfüllt:

$$\forall S : \mathit{list}[\mathit{list}[\mathit{LITERAL}]].\ \mathit{DavisPutnam}(S) \rightarrow \big(\forall \sigma : \mathit{list}[\mathit{LITERAL}].\ \neg\, \sigma \models\!\models S\big) \qquad (1)$$

[2] Wie beim Sequenzenkalkül der Aussagenlogik erstellt man solche Herleitungsbäume „von der Wurzel nach oben".

[3] Element selektieren und im Menü Program->Comment wählen.

Der innere Allquantor kann äquivalenzerhaltend nach vorne gezogen werden:

$$\forall S : \mathit{list}[\mathit{list}[\mathit{LITERAL}]], \sigma : \mathit{list}[\mathit{LITERAL}].\ \mathit{DavisPutnam}(S) \rightarrow \neg\, \sigma \models S \tag{2}$$

Dies ist die Aussage des Lemmas *DavisPutnam is sound*. Beweisen Sie dieses Lemma. Hierzu wurde schon Vorarbeit geleistet, um den Beweisbaum durch Fallunterscheidung zu fünf einfachere Teilbäume zu reduzieren.

Praktische Übung 3.4 (Vollständigkeit des Davis-Putnam-Verfahrens)

Das Davis-Putnam-Verfahren ist genau dann *vollständig*, wenn es für jede unerfüllbare Klauselmenge S den Rückgabewert *true* liefert:

$$\forall S : \mathit{list}[\mathit{list}[\mathit{LITERAL}]].\ \big(\forall \sigma : \mathit{list}[\mathit{LITERAL}].\ \neg\, \sigma \models S\big) \rightarrow \mathit{DavisPutnam}(S) \tag{3}$$

Diese Formel ist zu folgender Formel äquivalent (durch Kontraposition):

$$\forall S : \mathit{list}[\mathit{list}[\mathit{LITERAL}]].\ \neg\, \mathit{DavisPutnam}(S) \rightarrow \big(\exists \sigma : \mathit{list}[\mathit{LITERAL}].\ \sigma \models S\big) \tag{4}$$

Wenn das Davis-Putnam-Verfahren also *false* liefert, muss es eine Variablenbelegung σ für die in S vorkommenden Variablen geben, die S erfüllt.

Ein konstruktiver Beweis dieser Aussage erfordert, dass wir für jede Klauselmenge S mit $\neg\, \mathit{DavisPutnam}(S)$ eine S erfüllende Variablenbelegung σ angeben. Formal lässt sich dies durch Angabe einer Funktion $f : \mathit{list}[\mathit{list}[\mathit{LITERAL}]] \rightarrow \mathit{list}[\mathit{LITERAL}]$ bewerkstelligen, die zu jedem S mit $\neg\, \mathit{DavisPutnam}(S)$ ein geeignetes $\sigma := f(S)$ berechnet. Um (4) zu beweisen, geben wir eine solche Funktion f an und beweisen

$$\forall S : \mathit{list}[\mathit{list}[\mathit{LITERAL}]].\ \neg\, \mathit{DavisPutnam}(S) \rightarrow f(S) \models S\ . \tag{5}$$

Damit ist der Existenzquantor aus (4) eliminiert, so dass sich (5) als Lemma in der Sprache $\mathcal{L}$ formulieren lässt. Man nennt eine Funktion f, die zur Elimination eines Existenzquantors eingeführt wird, *Skolemfunktion*. Die Skolemfunktion f in dieser Aufgabe berechnet ein Modell (d. h. eine erfüllende Variablenbelegung) für eine Klauselmenge S. Wir nennen die entsprechende Prozedur ρ. Sie ist in der Datei `Praxis3.vf` bereits implementiert und geht analog zum Davis-Putnam-Verfahren vor.

Sehen Sie sich die Prozedur ρ genau an, um ihre Funktionsweise zu verstehen. Beweisen Sie dann das Lemma *DavisPutnam is complete*.

Hier ein paar Hinweise:

- Lassen Sie die globalen Hypothesen und Induktionshypothesen nicht aus den Augen.
- Beweisen Sie zuerst den letzten Induktionsschritt.
- Beim ersten Induktionsschritt sollten Sie durch Fallunterscheidung kleinere Beweisebäume erstellen und diese dann bearbeiten.