Übung zur Vorlesung

Automatisierte Logik und Programmierung

Prof. Chr. Kreitz

Universität Potsdam, Theoretische Informatik — Sommersemester 2009

Blatt 3 — Abgabetermin: 26.06.2009

Aufgabe 3.1 (Anwendungsbezogene Erweiterung formaler Theorien)

Das n-Dame Problem: Gegeben ein Schachbrett mit nxn Feldern und n Dame Figuren. Eine Dame kann alle Figuren schlagen, die sich auf derselben waagerechten oder senkrechten Linie befinden sowie alle Figuren, die sich auf der von ihr ausgehenden Diagonale befinden. Gesucht sind *alle* möglichen Plazierungen der n Damen auf dem Schachbrett, so daß keine Dame eine andere schlagen kann.

Entwickeln Sie eine formale Spezifikation des n-Dame Problems. Stellen Sie eventuell notwendige Definitionen für neue Begriffe auf und beschreiben Sie einige Gesetze dieser neuen Konzepte.

Aufgabe 3.2 (Synthese von Divide & Conquer Algorithmen)

Erzeugen Sie mithilfe der formalen Synthesestrategie für Divide & Conquer Algorithmen den Quicksort-Algorithmus für das Sortieren von Listen ganzer Zahlen. Welche Lemmata benötigt die Strategie, um die Komponenten herzuleiten?

Hinweise: Wie beim Mergesort-Algorithmus der Vorlesung muß man in zwei Phasen vorgehen, da der Quicksort-Algorithmus eine nichttriviale Dekomposition besitzt. Berücksichtigen Sie auch, daß Quicksort in einem gewissem Sinne invers zu Mergesort arbeitet, also die Dekomposition invers zur Komposition von Mergesort operiert, während die Komposition verhältnismäßig einfach ist und als Ausgangspunkt genommen werden sollte.

Aufgabe 3.3 (Formalisierung von Grundbegriffen der Programmsynthese)

In der Vorlesung haben wir die Grundbegriffe der Programmsynthese semi-formal beschrieben. Für eine formale Programmsynthese innerhalb eines Beweissystems müssen diese Konzepte formalisiert werden. Formalisieren Sie die folgenden Begriffe innerhalb der CTT.

3.3–a Die Klasse aller Spezifikationen als ein Datentyp `SPECIFICATIONS`

3.3–b Die Klasse aller Programme als ein Datentyp `PROGRAMS`

3.3–c Programmkorrektheit als ein “Prädikat” p `ist korrekt` *(beachten Sie Terminierung!)*

3.3–d Erfüllbarkeit von Spezifikationen als ein “Prädikat” *spec* `ist erfüllbar`

3.3–e Die Notation für Programme
`FUNCTION` f`(x:D):R WHERE I(x) RETURNS y SUCH THAT O(x,y) =` `body(x)`

Aufgabe 3.4 (Aufwendig: Formalisierung von Graphen und Bäumen)

Graphen und Bäume sind wichtige Datenstrukturen für eine große Menge von Anwendungsproblemen. Formalisieren Sie eine Theorie endlicher Graphen und Bäume derart, daß sich die folgenden Probleme darin beschreiben lassen.

- Das Cliquen-Problem: Gegeben ein Graph $G = (V, E)$ der Größe n und eine Zahl $k \leq |V|$. Gibt es in G eine Clique der Mindestgröße k?
- Das Independent Set: Gegeben ein Graph $G = (V, E)$ der Größe n und eine Zahl $k \leq |V|$. Gibt es in G eine unabhängige Knotenmenge der Mindestgröße k?
- Das Vertex Cover Problem: Gegeben ein Graph $G = (V, E)$ der Größe n und eine Zahl $k \leq |V|$. Gibt es in G eine Knotenüberdeckung der Maximalgöße k?
- Das Travelling Salesman Problem: Gegeben ein vollständiger gewichteter Graph (G, g). Gibt es in G einen Zyklus dessen Gesamtgewicht unter B liegt?
- Das MWST Problem. Gegeben ein gewichteter Graph (G, g). Bestimme einen minimal spannenden Baum von G.

3.4–a Formalisieren Sie zunächst die wichtigsten Begriffe der Theorie endlicher Graphen und Bäume in der Typentheorie (Beispiele sind im Anhang genannt). Identifizieren Sie dabei Konzepte aus der Theorie endlicher Mengen und Listen, die für eine Formalisierung erforderlich wären.

3.4–b Formulieren Sie Lemmata, welche Zusammenhänge zwischen verschiedenen Begriffen beschreiben, z.B. den Zusammenhang zwischen Cliquen und unabhängige Knotenmengen.

3.4–c Formalisieren Sie die obengenannten Probleme als “Spezifikationstheoreme”, mit denen im Sinne des Prinzips “Beweise als Programme” gezeigt würde, daß für jede Problemstellung eine Lösung konstruierbar ist. Wie müssen dabei Entscheidungsprobleme spezifiziert werden?

Diese Aufgabe ließe sich leicht zu einem Projekt/Studienarbeit erweitern

Beispiele graphentheoretischer Definitionen

- Ein (ungerichteter) *Graph* ist ein Paar $G = (V, E)$, wobei V endliche Menge und $E \subseteq \{ \{v, v'\} \mid v, v' \in V \wedge v \neq v' \}$.
  Ein Graph ist darstellbar als Liste $v_1, ..., v_n, \{v_{i_1}, v'_{i_1}\}, ..., \{v_{i_m}, v'_{i_m}\}$.
- Ein *gerichteter Graph* ist ein Paar $G = (V, E)$, wobei V endliche Menge und $E \subseteq V \times V$.
- Ein *gewichteter Graph* ist ein Graph $G = (V, E)$ mit einer Gewichtungsfunktion $g : E \to \mathbb{N}$.
- Ein Graph $H = (V_H, E_H)$ ist genau dann *Subgraph* des Graphen $G = (V, E)$ ($H \sqsubseteq G$), wenn alle Ecken und Kanten von H auch Ecken bzw. Kanten in G sind:
  $$(V_H, E_H) \sqsubseteq (V, E) :\Leftrightarrow V_H \subseteq V \wedge E_H \subseteq E$$
- $H = (V_H, E_H)$ ist *isomorph* zu $G = (V, E)$ (kurz: $H \cong G$), wenn die Graphen durch Umbenennung (bijektive Abbildung $h : V_2 \to V$) ineinander überführt werden können:
  $(V_H, E_H) \cong (V, E) :\Leftrightarrow \exists h : V \to V_H.(h \text{ bijektiv} \wedge E_H = \{\{h(u), h(v)\} \mid \{u, v\} \in E\})$
- Die *Größe* $|G|$ eines Graphen $G = (V, E)$ ist die Anzahl $|E|$ seiner Kanten.
- Der *Komplementärgraph* des Graphen $G = (V, E)$ ist der Graph $G^c = (V, E^c)$ mit $E^c = \{ \{v, v'\} \mid v, v' \in V \} - E$.
- Eine *Clique* der Größe k im Graphen $G = (V, E)$ ist eine vollständig verbundene Knotenmenge $V' \subseteq V$ mit $|V| = k$.
  Dabei heißt V' *vollständig verbunden*, wenn gilt: $\forall v, v' \in V'. v \neq v' \Rightarrow \{v, v'\} \in E$
- Eine *unabhängige Knotenmenge* der Größe k im Graphen $G = (V, E)$ ist eine Knotenmenge $V' \subseteq V$ mit $|V| = k$ mit der Eigenschaft $\forall v, v' \in V'. v \neq v' \Rightarrow \{v, v'\} \notin E$
- Eine *Knotenüberdeckung* (Vertex cover) des Graphen $G = (V, E)$ ist eine Knotenmenge $V' \subseteq V$ mit der Eigenschaft $\forall \{v, v'\} \in E. v \in V' \vee v' \in V'$
- Ein *Zyklus* (*Kreis*) in einem Graphen $G = (V, E)$ ist eine Menge $V_z = \{v_1, .., v_n\} \subseteq V$ mit der Eigenschaft $\forall i < n. \{v_i, v_{i+1}\} \in E \wedge \{v_n, v_1\} \in E$)
- Ein *Hamiltonscher Kreis* im Graphen $G = (V, E)$ ist ein Kreis, der nur aus Kanten aus E besteht und jeden Knoten genau einmal berührt.
- Ein Graph $G = (V, E)$ heißt zusammenhängend, wenn jeder Knoten in V von jedem anderen Knoten über Kanten aus E erreichbar ist.
- Ein *Baum* ist ein zyklenfreier zusammenhängender Graph.
- Ein *spannender Baum* in einem Graphen $G = (V, E)$ ist ein Subgraph $G_B = (V, E_B)$ von G, der ein Baum ist.
- Ein *minimal spannender Baum* in einem gewichteten Graphen (G, g) ist ein spannender Baum von G mit minimalem Gesamtgewicht.

Detailliertere Formulierungen mancher Konzepte findet man z.B. in

- S. O. Krumke, H. Noltemeier: *Graphentheoretische Konzepte und Algorithmen*, Teubner 2005.
- C. Meinel, M. Mundhenk: *Mathematische Grundlagen der Informatik*, Teubner 2002.
- K. Denecke: *Algebra und Diskrete Mathematik für Informatiker*, Teubner 2003.