UNIKASSEL VERSITÄT | ELEKTROTECHNIK INFORMATIK

Jens Kosiol

Ausgabe: 6. Mai 2024
Abgabe: 20. Mai 2024

Graph- and Model-Driven Engineering

Übungsblatt 1

Allgemeine Hinweise

- Übungsblätter können in Gruppen von zwei Personen bearbeitet werden.
- Die Abgabe erfolgt per E-Mail an jens.kosiol@uni-kassel.de *vor Beginn der Vorlesung* am Montag.
- Aufgaben, bei denen Graphtransformationssysteme zu entwerfen sind, sollen in Groove bearbeitet werden. Es ist *immer* die entworfene Grammatik als gps-Datei mit abzugeben. Zur Erklärung ihrer Lösung können Sie entweder sog. Kommentarknoten in Groove verwenden oder Ihre Graphen und Regeln als Bilder exportieren und in eine Textdatei einbinden. Geben Sie Ihre Lösungen als *eine* zip-Datei ab, die die entworfenen Grammatiken und *maximal eine* pdf-Datei mit Erklärungen enthält. *Verwenden Sie Ihre(n) Nachnamen im Dateinamen Ihrer Abgabe.*
- Bei Fragen darf gerne der fragen-Kanal auf dem Discord-Server zur Vorlesung verwendet werden. Hier können Sie sich auch gegenseitig Hilfestellungen geben. Bloßes Abschreiben vollständiger Lösungen führt allerdings zur Bewertung der Aufgabe mit 0 Punkten.
- Für die Zulassung zur Prüfung müssen mindestens 50 % der Zettelpunkte erreicht werden; außerdem darf maximal ein Übungsblatt vollständig nicht bearbeitet sein.

UNIKASSEL VERSITÄT | ELEKTROTECHNIK INFORMATIK

Jens Kosiol

Ausgabe: 6. Mai 2024
Abgabe: 20. Mai 2024

Graph- and Model-Driven Engineering
Übungsblatt 1

1 Graphtransformationssystem (GTS) aus Code (10 Punkte)

Gegeben sei die Java-Implementierung einer Liste (Chain) aus Listing 1.[1] Entwerfen Sie, analog zum Beispiel für zirkuläre Puffer aus der Vorlesung und eines Stack aus der Übung, ein getyptes Graphtransformationssystem, das diese Datenstruktur modelliert:

a) Geben Sie einen passenden Typgraphen und Regeln, die den Methoden insert und deleteNext entsprechen, an.

b) Erstellen Sie einen sinnvollen Startgraphen für das Graphtransformationssystem, der sich aus dem Konstruktor für Chain ergibt.

c) Erstellen Sie eine Regel, die der Methode isEmpty entspricht, und erläutern Sie, wie die Anwendung der Regel als Anwendung der Methode isEmpty zu interpretieren ist.

Geben Sie außerdem einen Graphen an, der mit Hilfe Ihrer Regeln aus Ihrem Startgraph *nicht* abgeleitet werden kann.

```
public class Chain<T> {
    Cell<T> anchor;

    Chain() { this.anchor = new Cell<T>(); }

    public Boolean isEmpty() {
        return this.anchor.next == null;
    }
}

public class Cell<T> {
    T content;
    Cell<T> next;

    Cell(){};
```

[1] Inspirationsquelle für den Code: Heinz-Peter Gumm und Manfred Sommer: *Einführung in die Informatik*. Bd 1: *Programmierung, Algorithmen und Datenstrukturen* (De Gruyter Oldenbourg 2016).

Jens Kosiol

Ausgabe: 6. Mai 2024
Abgabe: 20. Mai 2024

Graph- and Model-Driven Engineering
Übungsblatt 1

```
    Cell(T content, Cell<T> next) {
        this.content = content;
        this.next = next;
    }

    public void insert(T content) {
        this.next = new Cell<T>(content, this.next);
    }

    public void deleteNext() {
        if (this.next != null) {
            this.next = this.next.next;
        }
    }
}
```

Listing 1: Java-Implementierung einer Chain.

2 Regelsatz für ein adaptiertes Behälterproblem (12 Punkte)

In dem Behälterproblem geht es grundsätzlich darum, zu entscheiden, ob es möglich ist, eine gegebene Anzahl von Objekten bestimmter Größen so auf k Behälter zu verteilen, dass keiner der Behälter überläuft; wir betrachten hier eine Variante, wo jeder Behälter seine eigene Größe hat. Formal: Gegeben sind k Behälter jeweils der Größe $b_j \in \mathbb{N}$ für $j = 1, \dots, k$ und n Objekte der Größen $a_i \in \mathbb{N}$ für $i = 1, \dots, n$. Gesucht ist eine Zuordnung $f\colon \{1, \dots, n\} \to \{1, \dots, k\}$, sodass für jedes $j \in \{1, \dots, k\}$ gilt:

$$\sum_{f(i)=j} a_i \leq b_j.$$

Entwerfen Sie in Groove einen Typgraphen, einen Startgraphen und eine Menge von Graphtransformationsregeln, die das Einfügen von Gegenständen in Behälter, das Entfernen von Gegenständen aus Behältern und das Verschieben von Gegenständen zwischen Behältern modellieren.

Jens Kosiol

Ausgabe: 6. Mai 2024
Abgabe: 20. Mai 2024

Graph- and Model-Driven Engineering
Übungsblatt 1

Die folgenden Bedingungen sollen hierbei erfüllt sein:

- Der Startgraph soll aus drei Behältern mit den Volumen 50 (2×) und 75 bestehen. Des weiteren sind 10 Objekte mit den Gewichten 8 (2×), 12 (2×), 17 (2×), 22, 25 und 27 (2×) gegeben.
- Die Regeln sollen so gestaltet sein, dass ein Gegenstand nie zwei Behältern gleichzeitig zugeordnet sein kann und Behälter nicht überladen werden können.

Wichtig: Entwerfen Sie einen Typgraphen und aktivieren Sie diesen. Beachten Sie, dass Sie keinen Algorithmus zur Lösung des Behälterproblems entwerfen müssen, sondern lediglich Regeln zum Hinzufügen, Entfernen und Verschieben von Gegenständen, also Regeln, die innerhalb eines solchen Algorithmus eingesetzt werden könnten.

3 Graphtransformationssysteme mit Vererbung (8 Punkte)

Abbildung 1 zeigt den Typgraphen, Startgraphen und Transformationsregeln eines getypten Graphtransformationssystems mit Vererbung.

a) Spezifizieren Sie *formal* das getypte Graphtransformationssystem, welches aus dem Typgraph *TG*, dem Startgraph *SG* und der Regel *FensterZu* besteht. Geben Sie also textuell alle Tupel, Mengen und Morphismen an, welche das Graphtransformationssystem definieren. Nutzen Sie ggfs. Indizes, um gleichnamige Elemente unterschiedlicher Graphen zu unterscheiden (bspw. $Kitchen_{TG}$, $Kitchen_{SG}$, $Kitchen_{R}$ usw.).

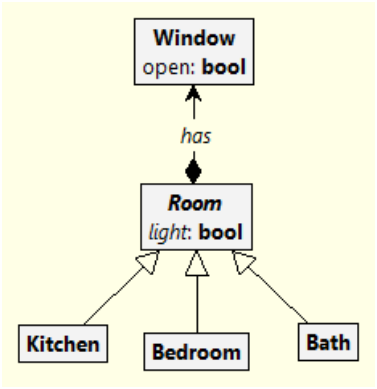
b) Klopfen Sie den Typgraphen in Bezug auf die darin enthaltene Vererbungshierarchie flach. Entfernen Sie also jegliche Vererbung und passen Sie Objekte und Kanten so an, dass keine Informationen verloren gehen. Überlegen und begründen Sie zudem, wieviele Regeln benötigt werden, um das Verhalten der in Abbildung 1 dargestellten Regeln für den angepassten Typgraphen zu realisieren. Legen Sie Ihrer Lösung den angepassten Typgraphen als Zeichnung oder Grafik bei.

Jens Kosiol

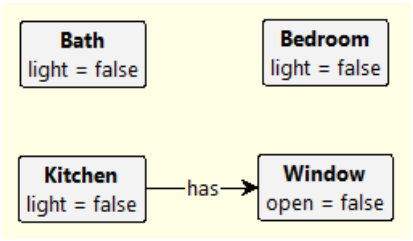
Ausgabe: 6. Mai 2024
Abgabe: 20. Mai 2024

Graph- and Model-Driven Engineering

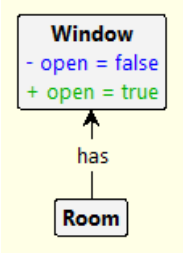
Übungsblatt 1



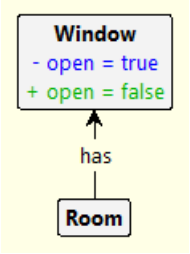
(a) TG



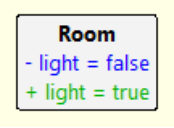
(b) SG



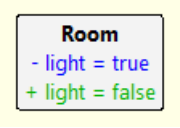
(c) FensterAuf



(d) FensterZu



(e) LichtAn



(f) LichtAus

Abbildung 1: Typgraph, Startgraph und Regeln eines GTS.