Automatische Erzeugung und Bewertung von Übungsaufgaben zu Algorithmen und Datenstrukturen

Johannes Waldmann, HTWK Leipzig

ABP Potsdam, 5. 10. 2017

Programmier-Aufgaben zu A&D?

- Vorlesung A&D (Algorithmen und Datenstrukt.):
  - Inhalt: Eigenschaften von (d.h., Beweise von Sätzen über) Algorithmen
  - Ziel: Entwurf, Auswahl von Algorithmen
- Vorlesung P (Programmierung):
  - Inhalt: Eigenschaften von Programmen und Programmiersprachen
  - Ziel: Implementierung von Algorithmen durch Programme
- dieser Vortrag:
  - Aufgaben zu A&D weitgehend ohne P,
  - trotzdem automatisch erzeugt und bewertet

Beispiel: AVL-Bäume

- Aufgabenstellung:
  - gegeben: Baum t_1, Baum t_2, Muster $s_?$ (Folge mit Lücken)
  - gesucht: Folge $s = [s_1, \ldots, s_k]$ von Operationen (Einfügen, Löschen), so daß $t_1 \xrightarrow{s} t_2$ und s paßt zu $s_?$
- System `autotool`
  - zeigt t_1, t_2, liest s,
  - führt s schrittweise aus, zeigt Zwischenresultate an,
  - vergleicht Endresultat mit t_2 sowie s mit $s_?$

AVL-Bäume (Demonstration)

- `https://autotool.imn.htwk-leipzig.de/cgi-bin/Trial.cgi?lecture=240`

- `https://gitlab.imn.htwk-leipzig.de/autotool/all0#leipzig-autotool`

Nachvollziehbare System-Antworten

- (. . . vergleicht Endresultat mit t_2 sowie s mit $s_?$)
- Korrektheit ist
  - nicht das Übereinstimmen mit einer geheimen Musterlösung,
  - sondern die Erfüllung der Spezifikation
- Akzeptanz und Ablehnen von Lösungsversuchen wird sofort und nachvollziehbar begründet, ohne eine korrekte Lösung zu verraten
- mehrfaches Einsenden ist möglich und erwünscht

Erzeugung von Aufgaben-Instanzen

- Aufgabensteller konfiguriert Generator (Größe der Bäume, Anzahl der gesuchten/vorgegebenen Operationen)
- `autotool` erzeugt Aufgabeninstanz je Student (verwendet deterministischen Pseudozufallsgenerator, initialisiert mit Hash der Matrikelnummer)
- verhindert Abschreiben von Lösungen (gemeinsames Bearbeiten ist erwünscht, aber es bearbeitet dann jeder eine andere Instanz)

Aufgabe zu Hashtabellen (Beispiel)

```
Replace each hole (_) in the configuration
    Config { size = 10
           , hash1 = \ x -> _ + _ * x
           , hash2 = \ x -> _           }
and in the sequence of operations
  [ Insert _ , Insert _ , Insert _
  , Insert 40 , Insert _ , Insert _ , Insert 22 ]
with a numerical expression
such that the sequence of operations transforms
    Table (listToFM [ (2,12),(9,41)])
to  Table (listToFM
      [ (0,11),(1,25),(2,12),(3,89),(4,22)
      , (6,40),(7,97),(8,64),(9,41) ])
```


Aufgabe zu Breitensuche (Beispiel)

```
Replace the holes (_) in the adjacency list of th
Adjacency_List
  [(_,[_,4,7] ),(_,[5,_]),(4,[_,_])
  ,(_,[_,_,4,5]),(_,[4,_,1,3]),(3,_),(6,[_,1,_])]
so that the breadth first search tree of G,
starting at 3, matches the pattern
  Branch 3
    [Branch _ [Branch 6 _
              ,Branch 5 [Branch 2 [],Branch _ []]
```

Beispiel-Generator

```
Config { graph_generator = Graph_Generator
   { directed = True, vertices = 7, out_degree_ra
 , handle_graph = Show 0.7, handle_bfs_tree = Sho
 , candidates = 100 }
```

autotool: Ansatz, Umfang

- Aufgabentyp = domainspezifische Sprache
  - Semantik: aufgabenspezifisch (realisiert durch spezifische Interpreter)
  - Syntax: (weitgehend) uniform
    - Haskell-Syntax für Daten-Literale für algebraische Datentypen
    - Parser, Pretty-Printer, Dokumentation: automatisch typgesteuert generiert
- 422 Aufgabentypen, 130 kLOC, seit 2002, SS17 @ HTWK: 25k Einsendg. (2.5k richtige) weitere Anwender an U Leipzig, Halle, Bonn

autotool: Implementierung

- System-Bestandteile:
  - zustandsloser Semantik-Server (erzeugt Instanzen und Lösungsvorlagen, bewertet Einsendungen)
  - Persistenz-Schicht (DB) (Aufgaben-Konfigurationen, Punkte usw.)
  - Web-Oberfläche (zur Bedienung im Browser)
- Implementierung komplett in Haskell (ausdrucksstark, sicher, effizient)
- alternativ: Persistenz und GUI in `openOlat`

autotool: das will ich auch haben!

- selbst kompilieren und betreiben (Lizenz: GPL)
- eigene openOlat-Instanz mit autOlat-Plugin bauen und betreiben, dazu Semantik-Server @ HTWK Leipzig
- andere Varianten Verhandlungssache (Nutzung von Ressourcen, Schutz personenbezogener Daten)
- Anbindung zu anderen Systemen? Gern. QTI-Standard scheint ungeeignet. Es fehlen
  - (externes) Bewerten von Einsendungen
  - (externes) Generieren von Aufgaben-Instanzen