Funktionale Programmierung

LVA 185.A03, VU 2.0, ECTS 3.0
WS 2018/2019

– Vorbesprechung –

(Stand: 01.10.2018)

Jens Knoop

Technische Universität Wien
Information Systems Engineering
Compilers and Languages

Funktionale Programmierung im WS 2018/19

A

Inhalt, Ziele, Motivation

Funktionale Programmierung

...komplementiert und rundet die grundlegenden Lehrveranstaltungen zu wichtigen Programmierparadigmen und -stilen ab:

- **Funktionale Programmierung**
  LVA 185.A03 Funktionale Programmierung
  VU 2.0 ECTS 3.0 WS 2018/19
- Objektorientierte Programmierung
  LVA 185.A01 Objektorientierte Programmiertechniken
  VU 2.0 ECTS 3.0 WS 2018/19
- Logikorientierte Programmierung
  LVA 185.A12 Logikprogrammierung und Constraints
  VU 4.0 ECTS 6.0 WS 2018/19

...die alle auch in zugehörigen fortgeschrittenen Lehrveranstaltungen fortgeführt und vertieft werden.

Generalthema der Lehrveranstaltung

...der **funktionale** Programmierstil, insbesondere:

- Programme als Systeme (wechselweise) rekursiver Rechenvorschriften.
- Lambda-Kalkül als rigorose semantische Fundierung funktionaler Programmiersprachen und Programmierung.
- Auswertungsordnungen für Ausdrücke und Programme, insbesondere sofortige und verzögerte Auswertung (engl. eager, lazy evaluation).
- Programmieren mit Funktionen als Argument und Resultat von Funktionen (sog. Funktionen höherer Ordnung).
- Polymorphie auf Datentypen und Funktionen.
- ...
- Umsetzung, Anwendung der Konzepte in Haskell.

Im Mittelpunkt die Frage:

"Can programming be liberated from the von Neumann style?"

John W. Backus, 1978

John W. Backus. Can Programming be Liberated from the von Neumann Style? A Functional Style and its Algebra of Programs. Communications of the ACM 21(8):613-641, 1978. (Turing Award Speech)

Lehrveranstaltungsgliederung (1)

- Teil I: Einführung
  - Warum fkt. Programmierung? Warum mit Haskell? Beispiele und Werkzeuge.
- Teil II: Grundlagen
  - Elementare Typen, Funktionen, Datentypdeklarationen.
- Teil III: Applikative Programmierung
  - Funktionen über elementaren Werten und Resultaten.
- Teil IV: Funktionale Programmierung
  - Funktionen über Funktionen, mit Funktionen als Argument und Resultat; Polymorphie auf Funktionen und Datentypen.

Lehrveranstaltungsgliederung (2)

- Teil V: Fundierung funktionaler Programmierung
  - λ-Kalkül, Auswertungsordnungen, Typprüfung, Typinferenz.
- Teil VI: Weiterführende Konzepte
  - Ein-/Ausgabe, Fehlerbehandlung, Modulkonzept, Programmierprinzipien, Programmieren mit Strömen und Funktionen höherer Ordnung.
- Teil VII: Abschluss und Ausblick
  - Abschluss, Ausblick.
- Literaturverzeichnis
- Anhänge
  - Formale Rechenmodelle
  - Andere funktionale Sprachen

Ziele der Lehrveranstaltung

Rüstzeugvermittlung zum Wissens- und Fertigkeitserwerb in Theorie und Praxis für

- die grundlegenden und tragenden Konzepte und Prinzipien funktionaler Programmierung, ihrer theoretischen Fundierung, praktischen Umsetzung und Anwendung in funktionalen Programmiersprachen und funktionaler Programmierung.
- den angemessenen und zweckmäßigen Einsatz funktionaler Programmierung und ihrer Konzepte und Prinzipien für die Lösung programmiertechnischer Aufgaben.
- die konkrete Umsetzung und Anwendung dieser Konzepte und Prinzipien in der 'state-of-the-art' Programmiersprache Haskell, darunter auch Tipps, Tricks und mehr!

...die (positive) Beantwortung der Eingangsfrage: "Can programming be liberated from the von Neumann style?"

Zum Einstieg

...drei kurze süffig zu lesende Artikel aus den Jahren:

1998: Philip Wadler. Why no one uses Functional Languages. ACM SIGPLAN Notices 33(8):23-27, 1998.

...there is a tension between building useful systems and extending the frontiers of research.

2011: Yaron Minsky. OCaml for the Masses. Communications of the ACM 54(11):53-58, 2011.

...why the next language you learn should be functional!

2018: Neil Savage. Using Functions for Easier Programming. Communications of the ACM 61(5):29-30, 2018.

...when the limestone of imperative programming has worn away, the granite of functional programming will be revealed underneath.

Warum die nächste Sprache

...funktional sein sollte:

- Konrad Hinsen. The Promises of Functional Programming. Computing in Science and Engineering 11(4): 86-90, 2009.

  ...adopting a functional programming style could make your programs more robust, more compact, and more easily parallelizable.

- Konstantin Läufer, Geoge K. Thiruvathukal. The Promises of Typed, Pure, and Lazy Functional Programming: Part II. Computing in Science and Engineering 11(5): 68-75, 2009.

  ...this second installment picks up where Konrad Hinsen's article "The Promises of Functional Programming" [...] left off, covering static type inference and lazy evaluation in functional programming languages.

Wo fkt. Progr. heute angewendet wird (1)

Von Wissenschaft:

- Jerzy Karczmarczuk. Scientific Computation and Functional Programming. Computing in Science and Engineering 1(3):64-72, 1999.

  ...modern functional programming languages and lazy functional techniques are useful for describing and implementing abstract mathematical objects in quantum mechanics.

- Noah M. Daniels, Andrew Gallant, Norman Ramsey. Experience Report: Haskell in Computational Biology. In Proc. 17th ACM SIGPLAN International Conference on Functional Programming (ICFP 2012), 227-234, 2012.

  ...Haskell gives computational biologists the flexibility and rapid prototyping of a scripting language, plus the performance of native code.

Wo fkt. Progr. heute angewendet wird (2)

...über Wirtschaft:

- Curt J. Simpson. Experience Report: Haskell in the "Real World": Writing a Commercial Application in a Lazy Functional Language. In Proceedings of the 14th ACM SIGPLAN International Conference on Functional Programming (ICFP 2009), 185-190, 2009.

  ...describe[s] the initial attempt of experienced business software developers with minimal functional programming background to writ a non-trivial, business-critical application entirely in Haskell. ...discuss[es] the advantages and difficulties of Haskell in these circumstances, with a particular focus on issues that commercial developers find important but that may receive less attention from the academic community.

Wo fkt. Progr. heute angewendet wird (3)

...bis Systemprogrammierung:

- Iavor S. Dachki, Thomas Hallgren, Mark P. Jones, Rebekah Leslie, Andrew Tolmach. Writing System Software in a Functional Language: An Experience Report. In Proceedings of the 4th International Workshop on Programming Languages and Operating Systems (PLOS 2007), Article No. 1, 5 Seiten, 2007.

  ...we describe our experience developing a prototype operating system, House, in which the kernel, device drivers, and even a simple GUI, are all written in Haskell.

Wo fkt. Progr. heute angewendet wird (4)

...ausgewählte sechs weitere Vorzeigebeispiele:

- Philip Wadler. An angry half-dozen. ACM SIGPLAN Notices 33(2):25-30, 1998.

  ...You've scrutinized functional languages. You've admired the elegance of lambda calculus, checked the benchmarks from the compilers, noted the security provided by strong typing. Now you want to know if they have been used to serious purpose. Mathematical elegance is well and good, but will it run that mission-critical system?

  Here are a half-dozen examplars of [serious] functional programs...

Functional Programming is Fun!

...weil funktionale Programmierung
etwas von der Eleganz der Mathematik
in die Programmierung bringt!

Peter Pepper. *Funktionale Programmierung in OPAL, ML, Haskell und Gofer*.
Springer-V., 2. Auflage, 2003.

B

Organisation, Ablauf, Beurteilung

Anmeldung zur Lehrveranstaltung

Anmeldung

- in TISS bis spätestens Freitag, 12.10.2018 (12:00 Uhr).

Voraussetzung einer validen Anmeldung:

- Erfolgreich abgeschlossene STEOP
  (Ausnahme: Erasmus$^+$-Studierende).

Studierende mit valider Anmeldung erhalten

- ein Benutzerkonto auf der Maschine
  `g0.complang.tuwien.ac.at`
- Benutzerkennung und erstes Losungswort per elektronischer Nachricht an ihre Standardadresse
  `e<Matr.Nr>@student.tuwien.ac.at`

zur Bearbeitung und Abgabe von Übungsaufgaben.

Aufbau und Ablauf der Lehrveranstaltung

Fünf Kernbestandteile:

- Vorlesung (regelmäßig dienstags von 8-10 Uhr).
- Plenumsübung 'Haskell Live' (regelmäßig freitags von 14-15 Uhr).
- Individual-Feedback 'Haskell Private' (nach Vereinbarung und Kapazität im November und Dezember).
- Übung mit Einzelabgaben (im Regelfall wöchentliche Abgaben mit einwöchiger Nachbearbeitungsfrist für Zweitabgaben nach Verbesserung).
- Schriftliche Prüfung Mitte Januar (vorauss. Do, 17.01.2019, 16:00-18:00 Uhr).

Zur schriftlichen Prüfung

- Dauer: 90 Minuten.
- Prüfungsstoff: Vorlesungsstoff, Übungsstoff, folgende zusammengehörende wissenschaftliche Artikel, die Sie sich selbstständig im Lauf der Vorlesungszeit erschließen:
  1. Konrad Hinsen. The Promises of Functional Programming. Computing in Science and Engineering 11(4): 86-90, 2009.
  2. Konstantin Läufer, Geoge K. Thiruvathukal. The Promises of Typed, Pure, and Lazy Functional Programming: Part II. Computing in Science and Engineering 11(5): 68-75, 2009.

  (Zugänglich aus TUW-Netz in IEEE Digital Library)
- Hilfsmittel: Keine.
- Anmeldung: Erforderlich; erfolgt in TISS.

Alle weiteren Informationen zu Anmelde- und Prüfungszeiten, Räumen, etc. der schriftlichen Prüfung: Siehe TISS!

Zur praktischen Übung

- Ausgabe von Aufgaben: Im Regelfall jeden Mittwoch ein neues Aufgabenblatt, abrufbar auf der Webseite der LVA); insgesamt vorauss. 8 Aufgabenblätter, beginnend am Mittwoch, den 17.10.2018.
- Abgabe von Lösungen: Eine Woche nach Ausgabe bis 15:00 Uhr; automatische Absammlung aus Home-Verzeichnissen (Top-level! Nicht in Unterverzeichnissen!) und Überprüfung mittels Hugs.
- Zweitabgabe von Lösungen bzw. verbesserten Lösungen: Eine bzw. zwei Wochen nach Erstabgabe abhängig von der Verfügbarkeit der Auswertungsergebnisse in ihrem Home-Verzeichnis (siehe Aufgabenblatt 1 für Details).
- Punktzahl pro Aufgabenblatt: 100.
- Erreichte Punkte pro Aufgabenblatt gemäß der Formel:
  `(Punkte Erstabgabe + Punkte Zweitabgabe) / 2`

Zur praktischen Übung (fgs.)

Wichtig:

- Zweitabgaben nach Verbesserung können die erreichte Punktzahl für ein Aufgabenblatt positiv und (bei nur vermeintlicher Verbesserung) negativ beeinflussen (siehe ‘goldene’ Berechnungsformel).
- Besonders wichtig: Auch wenn Sie schon beim ersten Mal 100 Punkte erreicht haben, müssen Sie für die Zweitabgabe eine Lösung zum Absammeln vorhalten (z.B. die Lösung ihrer Erstabgabe!).

Arbeit mit TUW- und eigenen Rechnern

- Server für die praktischen Programmierübungen: `g0.complang.tuwien.ac.at`
- Arbeit mit TUW-Rechnern: Verfügbar im Labor Argentinierstraße 8, Erdgeschoss im Innenhof.
- Arbeit mit anderen, eigenen Rechnern: Z.B. zu Hause ist möglich.
- Abgaben von Übungsaufgaben: Ausschließlich auf `g0.complang.tuwien.ac.at`.
- Nötige Software: Hugs (frei verfügbar).
- **Wichtig:** Abgaben werden auf der g0 ausschließlich unter Hugs getestet. Überzeugen Sie sich deshalb stets von der gewünschten Funktionalität Ihrer Programmierlösungen auf der g0 unter Hugs!

Gesamtbeurteilung, Zeugnisausstellung

- Positiver Abschluss: Nur, wenn praktische Übung (mindestens 50% der insgesamt möglichen Punkte) und schriftliche Prüfung (mindestens 'ausreichend') je für sich positiv abgeschlossen sind.
- Gesamtbeurteilung: Je zur Hälfte gewichtet die Beurteilung der praktischen Übungen und des Ergebnisses der schriftlichen Prüfung.
- Zeugnisausstellung: Jeweils frühestmöglich, insbesondere für jeden schriftlichen Prüfungsantritt; spätestens nach Ablauf des letzten Termins für die schriftliche Prüfung am Ende der Vorlesungszeit im SS 2019.

Termine für die schriftliche Prüfung

- Haupttermin: Vorauss. am Do, 17.01.2019, Anmeldung in TISS erforderlich!).
- Nebentermine: Insgesamt 3 Termine zu Beginn, in der Mitte und gegen Ende der Vorlesungszeit im SS 2019, vorauss. am Fr, 01.03.2019, Fr, 26.04.2019 und Fr, 21.06.2019, Anmeldung in TISS jeweils erforderlich!).
- Über diese Termine hinaus, keine weiteren Termine für die schriftliche Prüfung.
- Die genauen Termine, Rauminformationen und mögliche Änderungen werden in TISS bekanntgegeben; merken Sie sich die Termine bitte vor und planen Sie entsprechend!
- Wichtig: Im Zweifel gelten die in TISS angegebenen Termine.

Lehrbücher, wiss. Arbeiten, Haskell im Netz

...gezielte Hinweise auf

- Lehrbücher, wissenschaftliche Arbeiten
  - Siehe Vorlesungsunterlagen, detaillierte Leseempfehlungen am Ende jedes Kapitels.
- Haskell im Netz
  - Haskell-Homepage: `www.haskell.org/`
  - Haskell-Wiki: `wiki.haskell.org/Haskell/`
  - Haskell-Tutorial: `www.haskell.org/tutorial/`
  - Hugs-Interpretierer: `www.haskell.org/hugs`
- Haskell auf youtube
  - Interview mit John Hughes über 'Funktionale Programmierung und Haskell'.
    `https://www.youtube.com/watch?v=LnX3B9oaKzw`

Wichtige wiss. Zeitschriften und Konferenzen

...zur Publikation von Forschungsergebnissen im Umfeld funktionaler Programmierung und von Haskell sind besonders:

- Zeitschrift:
  - The Journal of Functional Programming. Matthias Felleisen, Jeremy Gibbons (Hrsg.), Cambridge, UK, seit 1991.
    `https://www.cambridge.org/jfp`
- Konferenz- und Symposiumsreihen:
  - ACM SIGPLAN International Conference Series on Functional Programming (ICFP), jährlich seit 1996.
    `http://www.sigplan.org/Conferences/ICFP`
  - ACM SIGPLAN International Haskell Symposium Series, jährlich seit 2008 (2002-2007 als ACM SIGPLAN Haskell Workshop Series).
    `https://wwww.haskell.org/haskell-symposium`

Bei Fragen, Problemen

...insbesondere:

- Webseite der LVA:
  `www.complang.tuwien.ac.at/knoop/fp185A03.html`
- Plenumsübung ‘Haskell Live’.
- Individual-Feedback ‘Haskell Private’.
- Tutorensprechstunde, -betreuung im Labor:
  Informationen zu den genauen Zeiten finden Sie (in Kürze) auf der Webseite der Lehrveranstaltung.

Mitveranstalter, Tutoren

- Mitveranstalter:
  1. Ass.Prof. Dipl.-Ing. Dr. techn. Ulrich Neumerkel
- Tutoren:
  1. Georg Faustmann
  2. Jakob Kremsner
  3. Bruno Tiefengraber

Nächste Termine

...für Vorlesung, Haskell Live, Haskell Private, Übung:

- **Vorlesung:**
  - Di, 02.10.2018 (08:15-09:45 Uhr), Informatikhörsaal.
  - Fr, 05.10.2018 (10:15-11:45 Uhr), EI3 Sahulka-Hörsaal.
  - Di, 09.10.2018 (08:15-09:45 Uhr), Informatikhörsaal.
  - Di, 16.10.2018 (08:15-09:45 Uhr), Informatikhörsaal.
- **Haskell Live:**
  - Fr, 12.10.2018 (14:15-15:00 Uhr), Informatikhörsaal.
  - Fr, 19.10.2018 (14:15-15:00 Uhr), Informatikhörsaal.
  - Fr, 26.10.2018 (Nationalfeiertag), Fr, 02.11.2018 (Allerseelen): Kein Haskell Live
  - Fr, 09.11.2018 (14:15-15:00 Uhr), Informatikhörsaal.
- **Haskell Private:** Im November und Dezember.
- **Praktische Übung:**
  - Erstes Aufgabenblatt: Mi, 17.10.2018, LVA-Webseite.
  - Erste Abgabe: Mi, 24.10.2018 (15:00 Uhr), auf der g0.

Für alle weiteren Termine: Siehe LVA-Webseite!

Zu guter Letzt

...wir, die FP-Teammitglieder, wünschen Ihnen viel Erfolg für diese Lehrveranstaltung und dass Sie auch langfristig von ihr profitieren!

Die Vorlesung lebt mit Ihnen! Ihre Rückmeldungen, Anregungen, Verbesserungsvorschläge sind willkommen! Natürlich auch, wenn Ihnen etwas gut gefallen hat!