universität
innsbruck

Einführung in die Theoretische Informatik

Martin Avanzini Christian Dalvit Jamie Hochrainer
Georg Moser Johannes Niederhauser Jonas Schöpf

https://tcs-informatik.uibk.ac.at

universität
innsbruck

Organisation

Zeitplan

Woche 1	8. Oktober	Woche 8	3. Dezember
Woche 2	22. Oktober	Woche 9	10. Dezember
Woche 3	29. Oktober	Woche 10	17. Dezember
Woche 4	5. November	Woche 11	14. Jänner
Woche 5	12. November	Woche 12	21. Jänner
Woche 6	19. November	Woche 13	28. Jänner
Woche 7	26. November	SL Klausur	4. Feber
		1te Klausur	11. Februar
		2te Klausur	11. März

Zeit und Ort

Vorlesung	Freitag, 10:15–12:00, HS A	Georg Moser
Tutorium	Donnerstag, 12:15-13:00, HS A	Christian Dalvit

Vorlesungsmaterial

1. Skriptum
   bei Studia (*10te überarbeitete Auflage*)


Universität Innsbruck Wintersemester 2021/22

Skriptum zur Vorlesung

Einführung in die Theoretische Informatik

Georg Moser

Wintersemester 2021/22

(c) G. Moser 10. Auflage

Online-Lehrmittel

2. Skriptum ist bei Studia verfügbar und wird ab übernächster Woche innerhalb des Universitätsnetzes verfügbar sein
3. Version mit Lösung zum Semesterende
4. Folien, Hausaufgaben, Selbsttests sind auf OLAT abrufbar
5. Folien sind (üblicherweise) vor der Vorlesung online
6. Ausgewählte Lösungen werden verfügbar gemacht, nachdem sie in den SL-Gruppen besprochen wurden

Zeit und Ort der Studienorientierungslehrveranstaltungen (SL)

Gruppe 1	Freitag, 13:15–14:00, HS C	Georg Moser
Gruppe 2	Freitag, 12:15–13:00, HSB 7	Jonas Schöpf
Gruppe 3	Freitag, 13:15–14:00, HSB 7	Jonas Schöpf
Gruppe 4	Freitag, 14:15–15:00, eLecture	Martin Avanzini
Gruppe 5	Freitag, 13:15–14:00, eLecture	Martin Avanzini
Gruppe 6	Freitag, 13:15–14:00, HSB 6	Johannes Niederhauser
Gruppe 7	Freitag, 14:15–15:00, HSB 6	Johannes Niederhauser
Gruppe 8	Freitag, 13:15–14:00, HS E	Jamie Hochrainer
Gruppe 9	Freitag, 12:15–13:00, HS E	Jamie Hochrainer

Termine

1. Die SL beginnt am 22. Oktober und endet am 28. Jänner; keine Ausnahmen für Nachmeldungen
2. SL Klausur am 4. Februar

Prüfungsmodus in Vorlesung & SL

SL

Aus formalen Gründen, besteht in der SL keine Anwesenheitspflicht. Wir empfehlen aber dringend die SL regelmäßig zu besuchen!

Klausuren

1. Die erste und zweite Vorlesungsprüfung findet im Februar, bzw. März statt; eine Anmeldung (per lfu:online) ist zwingend erforderlich.
2. Sofern möglich ist die Vorlesunsprüfung in Präsenz.
3. Die SL Klausur findet (in Präsenz) in der entsprechenden SL Gruppe statt; eine Anmeldung ist nicht erforderlich.
4. Die Klausurvorbereitungen (für VO+SL) finden im Tutorium/Vorlesung statt
5. Zur Abgrenzung von der VO KLausur wird sich die SL Klausur auf die Bereiche der LVA beschränken, die nicht in der Vorlesungsklausur behandelt werden.

Organisatorisches

ARSNova Session

1. QR Code
2. `https://arsnova.uibk.ac.at/mobile/#id/77974190`

Sämtliche weitere organisatorische Informationen, siehe OLAT

universität
innsbruck

Theoretische Informatik

Begriffsdefinition

Die Theoretische Informatik beschäftigt sich mit der Abstraktion, Modellbildung und grundlegenden Fragestellungen, die mit der Struktur, Verarbeitung, Übertragung und Wiedergabe von Informationen in Zusammenhang stehen.

Ihre Inhalte sind Automatentheorie, Theorie der formalen Sprachen, Berechenbarkeits- und Komplexitätstheorie, aber auch Logik und formale Semantik sowie die Informations-, Algorithmen- und Datenbanktheorie.

http://de.wikipedia.org/ 2021

1. Automatentheorie
2. Theorie der formalen Sprachen
3. Berechenbarkeits- und Komplexitätstheorie
4. Logik und formale Semantik
5. Informations-, Algorithmen- und Datenbanktheorie

Handbook of Theoretical Computer Science

+

= 2293 Seiten, 4 Kilogramm

Inhaltsverzeichnis Band „Algorithms and Complexity“

Machine models and simulations, A catalog of complexity classes, Machine-independent complexity theory, Kolmogorov complexity and its applications, Algorithms for finding patterns in strings, Data structures, Computational geometry, Algorithmic motion planning in robotics, Average-case analysis of algorithms and data structures, Graph algorithms, Cryptography, Algebraic complexity theory, Algorithms in number theory, The complexity of finite functions, Communication networks, VLSI theory, Parallel algorithms for shared-memory machines, General purpose parallel architectures

Inhaltsverzeichnis Band „Formal Models and Semantics“

Finite automata, Context-free languages, Formal languages and power series, Automata on infinite objects, Graph rewriting: an algebraic and logic approach, Rewrite systems, Functional programming and lambda calculus, Type systems for programming languages, Recursive applicative program schemes, Logic programming, Denotational semantics, Semantic domains, Algebraic specification, Logics of programs, Methods and logics for proving programs, Temporal and modal logic, Elements of relational database theory, Distributed computing: models and methods, Operational and algebraic semantics of concurrent processes

Geschichte	Rechenmodelle	Digitalrechner	
	Turing Maschinen, Berechenbarkeitstheorie, Alan Turing	—	1930er
	Formale Sprachen, Automatentheorie	`Zuse Z3`, `ENIAC`	1940er
	Grammatiken, Grundlagen des Compilerbaus, Noam Chomsky	`UNIVAC`, Transistoren statt Röhren	1950er
	P vs. NP Komplexitätstheorie, Stephen Cook	Minicomputer, integrierte Schaltkreise	1960er

Inhalte der Lehrveranstaltung

Einführung in die Logik

Syntax & Semantik der Aussagenlogik, Kalkül des natürlichen Schließens, Konjunktive und Disjunktive Normalformen

Einführung in die Algebra

algebraische Strukturen, Boolesche Algebra

Einführung in die Theorie der Formalen Sprachen

Grammatiken und Formale Sprachen, Reguläre Sprachen, Kontextfreie Sprachen, Chomsky-Hierarchie, Anwendungen von formalen Sprachen

Einführung in die Berechenbarkeitstheorie und Komplexitätstheorie

Algorithmisch unlösbare Probleme, Turing Maschinen, Registermaschinen, Komplexitätstheorie

Einführung in die Programmverifikation

Prinzipien der Analyse von Programmen, Verifikation nach Hoare

Bachelor Informatik: die ersten zwei Jahre

1. Semester	Einführung in die Programmierung		Einf. Theoretische Informatik	Rechnerarchitektur
	Funktionale Programmierung		Lineare Algebra	
2. Semester	Programmier-methodik	Algorithmen und Datenstrukturen	Angewandte Mathematik	Betriebssysteme
3. Semester	Softwarearchitektur	Datenbanksysteme	Diskrete Strukturen	Rechnernetze und Internettechnik
		Daten und Wahr-scheinlichkeiten		
4. Semester	Software Engineering	Maschinelles Lernen	Logik	Einführung in das wissensch. Arbeiten
	Parallele Programmierung			

universität
innsbruck

Einführung in die Logik

Logisches Schließen im Allgemeinen

Beispiel

- We arrive at the following paradox in a globalised world: when nationalists pursue more formal sovereignty they achieve less real sovereignty of the people. They want to take back control and they end up with less control.
- That's what the UK will end up with. [...]
- Yet this paradox also has a corollary: when countries in Europe renounce formal sovereignty this leads to more real sovereignty for the people of Europe.

Paul De Grauwe, 2017[1]

[1] Siehe `https://blogs.lse.ac.uk/brexit/2017/10/06/the-catalan-crisis-and-brexit-stem-from-the-same-kind-of-nationalism/`.

Results of Brexit

Beispiel

Yet this paradox also has a corollary: when countries in Europe renounce formal sovereignty this leads to more real sovereignty for the people of Europe.

Bemerkung

- In der Informatik und im Allgemeinen in den Natur- und Ingenieurswissenschaften muss ein „Korollar“ logisch folgen.
- Dazu haben wir in der Logik eine eigene formale Sprache für Folgerungen und allgemeiner gültige Schlüsse eingeführt.

Beispiel (Fortsetzung)

In dieser Sprache, stellt sich das sogenannte „Korollar“, wie folgt dar (und ist logisch falsch):

$$\text{„more formal sovereignty“} \to \text{„less real sovereignty“} \models$$
$$\models \neg\text{„more formal sovereignty“} \to \neg\text{„less real sovereignty“}$$

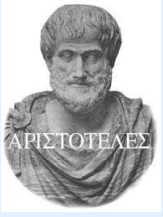
universität
innsbruck

Grundlagen der Logik

Aristoteles sagte

Topik I 1, 100a25-27

Eine Deduktion (syllogismos) ist also ein Argument, in welchem sich, wenn etwas gesetzt wurde, etwas anderes als das Gesetzte mit Notwendigkeit durch das Gesetzte ergibt



Beispiel

Sokrates ist ein Mensch	}	Prämisse ①
Alle Menschen sind sterblich	}	Prämisse ②
Somit ist Sokrates sterblich	}	Konklusion

Definition

- Schlussfiguren dieser Art heißen Syllogismen
- Syllogismen wurden bereits im antiken Griechenland untersucht, Grundlage der modernen Logik

Fakt

Nicht die Wahrheit der Prämissen, oder der Konklusion, sondern die Wahrheit der Schlussfigur ist entscheidend

Beispiele für Arten von Syllogismen

Alle Griechen sind Menschen Alle Menschen sind sterblich Somit sind alle Griechen sterblich	AAA - modus barbara
Alle Professoren sind ernst Einige Dozenten sind nicht ernst Somit sind einige Dozenten keine Professoren	AOO - modus baroco

Typisierung der Relationen

A	Alle *S* sind *P*	E	Keine *S* sind *P*
I	Einige *S* sind *P*	O	Einige *S* sind nicht *P*

Modus Ponens

Beispiel

Wenn das Kind schreit, hat es Hunger

Das Kind schreit

Also, hat das Kind Hunger

Fakt

Korrektheit dieser Schlussfigur ist unabhängig von den konkreten Aussagen

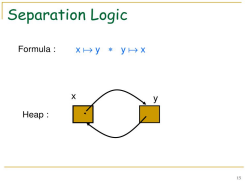
Definition (*Modus Ponens*)

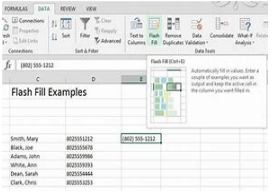
Wenn *A*, dann *B*

A gilt

Also, gilt *B*

Logik in der Informatik





Die Bedeutung der Logik in der Informatik ist um einiges größer als die Bedeutung der Logik in der Mathematik (oder Philosophie, Linguistik, . . .)