Theoretische Informatik I

Einheit 1

Mathematische Methodik

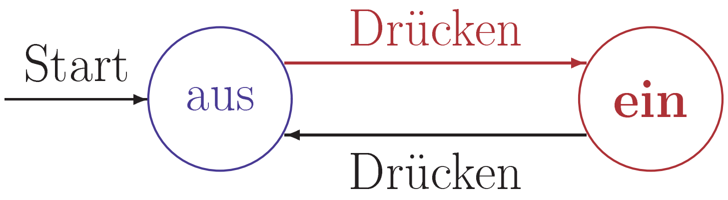
1. Formale Modelle
2. Beweistechniken
3. Wichtige Grundbegriffe

Methodik des Problemlösens

- **Klärung der Voraussetzungen**
  - Welche Begriffe sind zum Verständnis des Problems erforderlich?
  - Erstellung eines präzisen Modells: abstrahiere von überflüssigen Details
  - Formulierung des Problems im Modell: was genau ist zu tun?
- **Lösungsweg konkretisieren**
  - Welche Einzelschritte benötigt man, um das Problem zu lösen?
  - Welches Gesamtergebnis ergibt sich aus den Einzelschritten?
  - Wie beweist man die Korrektheit des Gesamtergebnisses?
- **Lösung zusammenfassen**
  - Kurz und prägnant: Argumente auf das Wesentliche beschränken
  - Umgangssprache durch mathematisch präzise Formulierungen ersetzen

Modelle für Berechnung und Sprachen

- **Automaten: Abarbeitung von Eingaben**
  - z.B. Wechselschalter: Verarbeitung von “Drück”-Eingaben



  - 2 Zustände: aus, ein – 1 Startzustand: aus
  - 1 Eingabesymbol: Drücken
  - 1 Endzustand: ein — wird erreicht bei ungerader Anzahl von Drücken

- **Grammatiken: Vorschriften für Spracherzeugung**
  - z.B.: $S \to$ Drücken | $S \to S$DrückenDrücken
  - Erzeigt nur ungerade Anzahl von Drücken-Symbolen

- **Reguläre Ausdrücke: algebraische Strukturen**
  - z.B.: (DrückenDrücken)*Drücken

Formale Beweise in der Informatik

- **Testen von Programmen ist unzureichend**
  - Nur hilfreich zur Entdeckung grober Fehler
  - Viele kleine, aber gravierende Fehler fallen durch das Testraster
    - Pentium Bug (1994), Ariane 5 (1996), Mars Polar Lander (1999), ...
- **Kritische Programme müssen "bewiesen" werden**
  - Erfolgreicher Beweis zeigt genau, wie das Programm arbeitet
  - Erfolgloser Beweisversuch deutet auf mögliche Fehler im Programm
  - Jeder Informatiker sollte die eigenen Programme beweisen
- **Jeder Informatiker muß Beweise verstehen**
  - Deduktive Beweise für sequentielle Verarbeitung
  - Induktionsbeweise für Rekursion / Schleifen
  - Widerlegungsbeweise und Gegenbeispiele für Unmöglichkeitsaussagen

Viele Informatiker wissen nicht, wie man stichhaltige Beweise führt

Behauptungen: Ausgangspunkt jeder Beweisführung

- **Wenn–Dann Aussagen:**
  - Eine Konklusion folgt aus aus einer oder mehreren Hypothesen (Annahmen)
  - z.B. "*Wenn* $x \geq 4$, *dann* $2^x \geq x^2$"
  - Auch: H *impliziert* K, *aus* H *folgt* K, K *wenn* H, $H \Rightarrow K$
  - Achtung: wenn K gilt, muß H nicht gelten (H muß nicht der Grund sein)

  Fast alle Behauptungen sind Wenn–Dann Aussagen
  - Hypothesen sind zuweilen implizit oder ergeben sich aus dem Kontext
  - z.B. "$\mathit{sin}^2\theta + \mathit{cos}^2\theta = 1$" hat implizite Hypothese "θ *ist ein Winkel*"

- **Genau dann, wenn Aussagen**
  - Zwei Aussagen A und B sind äquivalent ($A \Leftrightarrow B$, $A \equiv B$, A *iff* B (engl.))
  - z.B. "$x^2 = 1$ *genau dann, wenn* $x = 1$"
  - Gleichwertig mit $A \Rightarrow B$ und $B \Rightarrow B$

Beispiel eines deduktiven Beweises

Wenn x die Summe der Quadrate von vier positiven ganzen Zahlen ist, dann gilt $2^x \geq x^2$

- **Informaler Beweis**
  - Es sei x die Summe der Quadrate von vier positiven ganzen Zahlen
  - Das Quadrat jeder positiven ganzen Zahl ist mindestens 1
  - Aus der Annahme folgt damit, daß $x \geq 4$ sein muß
  - Wir benutzen den Satz "*Wenn* $x \geq 4$*, dann* $2^x \geq x^2$" HMU Satz 1.3, Folie 14
    und schließen daraus, daß $2^x \geq x^2$ gilt

- **Formaler Beweis**

	Aussage	Begründung
1.	$x = a^2 + b^2 + c^2 + d^2$	Gegeben
2.	$a \geq 1, b \geq 1, c \geq 1, d \geq 1$	Gegeben
3.	$a^2 \geq 1, b^2 \geq 1, c^2 \geq 1, d^2 \geq 1$	(2) und Gesetze der Arithmetik
4.	$x \geq 4$	(1), (3) und Gesetze der Arithmetik
5.	$2^x \geq x^2$	(4) und HMU Satz 1.3, Folie 14

Deduktive Beweisführung

Logische Beweisschritte von Annahme zur Konklusion

- **Beweis $\hat{=}$ Folge von Zwischenaussagen**
  - Beginne mit (Menge der) Annahmen
  - Jede Zwischenaussage folgt schlüssig aus (allen) vorhergehenden Aussagen
  - Konklusion ergibt sich als letzter Beweisschritt

- **Zulässige Argumente in Beweisschritten**
  - Logischer Schluß: Sind A und $A \Rightarrow B$ bekannt, kann B gefolgert werden
  - Bekannte mathematische Grundgesetze (z.B. Arithmetik)
  - Bereits bewiesene Sätze
  - Auflösung von Definitionen
  - Extensionalität von Mengen: $M{=}M'$ genau dann wenn $M{\subseteq}M' \wedge M{\subseteq}M'$
    $M{\subseteq}M'$ genau dann wenn $(\forall x)\, x \in M \Rightarrow x \in M'$
  - Gleichheit von Zahlen: $x{=}y$ genau dann wenn weder $x{<}y$ noch $x{>}y$

Beispiel für Auflösung von Definitionen

Wenn S endliche Teilmenge einer Menge U ist und das Komplement von S (bezüglich U) endlich ist, dann ist U endlich

- **Definitionen**

  S endlich $\equiv$ Es gibt eine ganze Zahl n mit $||S|| = n$

  T Komplement von S $\equiv$ $T{\cup}S = U$ und $T{\cap}S = \emptyset$

- **Beweis**

	Aussage	Begründung
1.	S endlich	Gegeben
2.	T Komplement von S	Gegeben
3.	T endlich	Gegeben
4.	$\|\|S\|\| = n$ für ein $n \in \mathbb{N}$	Auflösen der Definition in (1)
5.	$\|\|T\|\| = m$ für ein $m \in \mathbb{N}$	Auflösen der Definition in (3)
6.	$T{\cup}S = U$	Auflösen der Definition in (2)
7.	$T{\cap}S = \emptyset$	Auflösen der Definition in (2)
8.	$\|\|U\|\| = m + n$ für $n, m \in \mathbb{N}$	(4),(5),(6), (7) und Gesetze der Kardinalität
9.	U endlich	Einsetzen der Definition in (8)

Beweis einer Mengenäquivalenz

Für beliebige Mengen R und S gilt $R \cup S = S \cup R$

- **Definitionen**

  $x \in R \cup S \quad \equiv \quad x \in R$ oder $x \in S$

- **Zu zeigen:**
  - $R \cup S = S \cup R$ also
  - $R \cup S \subseteq S \cup R$ und $S \cup R \subseteq R \cup S$ also
  - *Wenn* $x \in R \cup S$ *dann* $x \in S \cup R$ und *wenn* $x \in S \cup R$ *dann* $x \in R \cup S$

- **Beweis der ersten Implikation**

	Aussage	Begründung
1.	$x \in R \cup S$	Gegeben
2.	$x \in R$ oder $x \in S$	Auflösen der Definition in (1)
3.	$x \in S$ oder $x \in R$	Logische Umstellung von (2)
4.	$x \in S \cup R$	Einsetzen der Definition in (3)

Beweis der zweiten Implikation genauso

Wie genau/formal muss ein Beweis sein?

Ein Beweis ist ein Argument, das den Leser überzeugt

- Präzise genug. um Details rekonstruieren zu können
- Knapp genug, um übersichtlich und merkbar zu sein
- Zwischenschritte müssen mit "üblichen" Vorkenntnissen erklärbar sein
- Also nicht notwendig formal oder mit allen Details
- Gedankensprünge sind erlaubt, wenn Sie die Materie gut genug verstehen, daß Sie nichts mehr falsch machen können
  … es reicht nicht, daß Sie es einmal richtig gemacht haben

- Tip: ausführliche Lösungen entwickeln, bis Sie genug Erfahrung haben.
  Bei Präsentation für andere zentrale Gedanken aus Lösung extrahieren
- Test: verstehen Ihre Kommilitonen Ihre Lösung und warum sie funktioniert?

Widerlegungsbeweise I

Zeige, daß eine Aussage A nicht gilt

- **Beweis durch Widerspruch**
  - A gilt nicht, wenn aus der Annahme von A ein Widerspruch folgt
  - z.B. *Wenn S endliche Teilmenge einer unendlichen Menge U ist, dann ist das Komplement von S (bezüglich U) unendlich*
  - Beweis

	Aussage	Begründung
1.	S endlich	Gegeben
2.	T Komplement von S	Gegeben
3.	U unendlich	Gegeben
4.	T endlich	Annahme
5.	U endlich	(1), (4) mit Satz auf Folie 7
6.	Widerspruch	(3),(5)
7.	T unendlich	Annahme (4) muß falsch sein

Widerlegungsbeweise II

- **Beweis durch Gegenbeispiel**
  - A ist nicht allgemeingültig, wenn es ein einziges Gegenbeispiel gibt
  - z.B. *Wenn x eine Primzahl ist, dann ist x ungerade* ist falsch
  - Beweis: 2 ist eine gerade Zahl, die eine Primzahl ist

- **Beweis durch Kontraposition**
  - Statt *wenn H dann K* zeige *wenn nicht K dann nicht H*
  - Behauptungen sind aussagenlogisch äquivalent

- **Spezielle Anwendung: Indirekte Beweisführung**
  - Zeige, daß aus *H und nicht K* ein Widerspruch folgt

    Aussagenlogisch äquivalent zu *wenn H dann K*
  - z.B. *Wenn für eine natürliche Zahl $x^2>1$ ist, dann ist $x\geq 2$*
  - Beweis: *Sei $x^2>1$. Wenn $x\geq 2$ nicht gilt, dann ist $x=1$ oder $x=0$.*

    Wegen $1^2=1$ und $0^2=0$ ist $x^2>1$ in beiden Fällen falsch.

    Also muß $x\geq 2$ sein

Gegenbeispielkonstruktion für unendliche Objekte

- **Terminierung von Programmen ist "unentscheidbar"**
  Es gibt kein Programm, das testen kann, ob eine beliebiges Programm bei einer bestimmten Eingabe überhaupt anhält

- **Beweis stützt sich auf wenige Grundannahmen**
  1. Programme und ihre Daten sind als Zahlen codierbar
  2. Computer sind universelle Maschinen
     · Bei Eingabe von Programm und Daten berechnen sie das Ergebnis
     · Schreibweise: $p_i(j) \hat{=}$ Anwendung des i-ten Programms auf die Zahl j
  3. Man kann Programme beliebig zu neuen Programmen zusammensetzen
     ... und die Nummer des neuen Programms berechnen

Terminierung von Programmen ist unentscheidbar

- **Annahme: es gibt ein Programm zum Test auf Terminierung**
  – `Term`(i,j)=1 falls $p_i(j)$ hält (sonst 0)

	0	**1**	**2**	**3**	**4**	…
p_0	$\perp$	$\times$	$\times$	$\perp$	$\times$	…
p_1	$\perp$	$\times$	$\times$	$\times$	$\times$	…
p_2	$\times$	$\times$	$\times$	$\times$	$\times$	…
p_3	$\perp$	$\times$	$\perp$	$\times$	$\perp$	…
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	…

$\times \hat{=}$ Terminierung, $\perp \hat{=}$ hält nicht

- **Konstruiere ein neues Programm**
  `Unsinn` **wie folgt**:

$$\texttt{Unsinn}(i) = \begin{cases} 0 & \text{wenn } \texttt{Term}(i,i)=0 \\ \perp & \text{sonst} \end{cases}$$

- `Unsinn` **ist ein Programm**
  Also muß `Unsinn` **eine Nummer k haben**

- **Was macht p_k=`Unsinn` auf seiner eigenen Nummer?**
  – Wenn $p_k(k)$ hält, dann `Term`(k,k)=1, also hält `Unsinn`(k) nicht an ???
  – Wenn $p_k(k)$ nicht hält, dann `Term`(k,k)=0, also hält `Unsinn`(k) an ???

- **Dies ist ein Widerspruch,**
  Also kann es den Test auf Terminierung nicht geben

Induktive Beweise I

Beweise eine Aussage A für alle natürlichen Zahlen

- **Standardinduktion**
  - Gilt A für i und folgt A für $n{+}1$, wenn A für n gilt dann gilt A für alle $n{\geq}i$
  - z.B. *Wenn* $x{\geq}4$, *dann* $2^x{\geq}x^2$

    Induktionsanfang $x{=}4$: Es ist $2^x = 16 \geq 16 = x^2$

    Induktionsschritt: Es gelte $2^n{\geq}n^2$ für ein beliebiges $n{\geq}4$

    Dann ist $2^{n+1} = 2{*}2^n \geq 2n^2$ aufgrund der Induktionsannahme

    und $(n{+}1)^2 = n^2{+}2n{+}1 = n(n{+}2{+}1/n) \leq n(n{+}n) = 2n^2$ wegen $n{\geq}4$

    also gilt $2^{n+1} \geq (n{+}1)^2$

- **Vollständige Induktion**
  - Folgt A für n, wenn A für alle $j{<}n$ mit $j{\geq}i$ gilt dann gilt A für alle $n{\geq}i$
  - Mächtiger, da man nicht den unmittelbaren Vorgänger benutzen muß

Strukturelle Induktion

Zeige A für alle Elemente einer rekursiven Datenstruktur

Gilt A für das Basiselement und folgt A für ein zusammengesetztes Element, wenn A für seine Unterelemente gilt, dann gilt A für alle Elemente

– z.B. *Die Summe einer Liste L von positiven ganzen Zahlen ist mindestens so groß wie ihre Länge*

Induktionsanfang L ist leer: Die Summe und die Länge von L ist 0

Induktionsschritt: Es gelte $sum(L) \geq |L|$

Betrachte die Liste $L \circ x$, die durch Anhängen von x and L entsteht

Dann gilt $sum(L \circ x) = sum(L) + x \geq sum(L) + 1 \geq |L| + 1 = |L|$

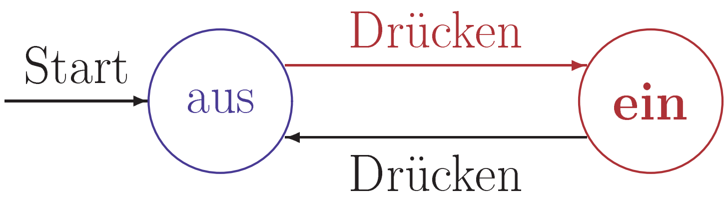
Häufig eingesetzt für Analyse von

- **Baumstrukturen** (Suchen, Sortieren, ...)
- **Syntaktische Strukturen** (Formeln, Programmiersprachen, ...)

⋮

Gegenseitige Induktion

Zeige mehrere zusammengehörige Aussagen simultan



Zeige: Automat ist ein Wechselschalter

– $S_1(n)$: Ist n gerade, so ist der Automat nach n-fachem Drücken ausgeschaltet

– $S_2(n)$: Ist n ungerade, so ist der Automat nach n-fachem Drücken eingeschaltet

Induktionsanfang $n{=}0$: n ist gerade also gilt $S_2(0)$
der Automat ist ausgeschaltet, also gilt $S_1(0)$

Induktionsschritt: Es gelte $S_1(n)$ und $S_2(n)$. Betrachte $n{+}1$

– Falls $n{+}1$ ungerade, dann gilt $S_1(n{+}1)$ und n ist gerade.
Wegen $S_1(n)$ war der Automat "aus" und wechselt auf "ein". Es gilt $S_2(n)$

– Falls $n{+}1$ gerade, dann gilt $S_2(n{+}1)$ und n ist ungerade.
Wegen $S_2(n)$ war der Automat "ein" und wechselt auf "aus". Es gilt $S_1(n)$

Mathematisches Vokabular I: Worte und Sprachen

- **Alphabet Σ**: endliche Menge von Symbolen,
  z.B. $\Sigma = \{0, 1\}$, $\Sigma = \{0, .., 9\}$, $\Sigma = \{A, .., Z, a, .., z, \ , ?, !, ..\}$
- **Worte**: endliche Folge w von Symbolen eines Alphabets
  Auch Zeichenreihen oder Strings genannt
- $\boldsymbol{\epsilon}$: Leeres Wort (ohne jedes Symbol)
- $\boldsymbol{w\ v}$: Konkatenation (Aneinanderhängung) der Worte w und v
- $\boldsymbol{u^i}$: i-fache Konkatenation des Wortes (oder Symbols) u
- $\boldsymbol{|w|}$: Länge des Wortes w (Anzahl der Symbole)
- $\boldsymbol{v \sqsubseteq w}$: v Präfix von w, wenn $w = v\,u$ für ein Wort u
- $\boldsymbol{\Sigma^k}$: Menge der Worte der Länge k mit Symbolen aus Σ
- $\boldsymbol{\Sigma^*}$: Menge aller Worte über Σ
- $\boldsymbol{\Sigma^+}$: Menge aller nichtleeren Worte über Σ
- **Sprache L**: Beliebige Menge von Worten über einem Alphabet Σ
  Üblicherweise in abstrakter Mengennotation gegeben
  z.B. $\{w \in \{0, 1\}^* \mid |w| \text{ ist gerade}\}$ $\quad \{0^n 1^n \mid n \in \mathbb{N}\}$
- **Problem P**: Menge von Worten über einem Alphabet Σ
  Das “Problem” ist, Zugehörigkeit zur Menge P zu testen

Mathematisches Vokabular II: Funktionen

- **Funktion $f : S \to S'$**: Abbildung zwischen den Grundmengen S und S'
  nicht unbedingt auf allen Elementen von S definiert
- **Domain von f**: $domain(f) = \{x \in S \mid f(x) \text{ definiert}\}$ (Definitionsbereich)
- **Range von f**: $range(f) = \{y \in S' \mid \exists x \in S\ f(x) = y\}$ (Wertebereich)
- **f total**: $domain(f) = S$ (andernfalls ist **f partiell**)
- **f injektiv**: $x \neq y \Rightarrow f(x) \neq f(y)$
- **f surjektiv**: $range(f) = S'$
- **f bijektiv**: f injektiv und surjektiv
- **Umkehrfunktion $f^{-1}: S' \to S$**: $f^{-1}(y) = x \Leftrightarrow f(x) = y$ (f injektiv!)
- **Urbild $f^{-1}(L)$**: Die Menge $\{x \in S \mid f(x) \in L\}$
- **Charakteristische Funktion χ_L von $L \subseteq S$**: $\chi_L(w) = \begin{cases} 1 & \text{falls } w \in L, \\ 0 & \text{sonst} \end{cases}$
- **Partiell-charakteristische Funktion ψ_L**: $\psi_L(w) = \begin{cases} 1 & \text{falls } w \in L, \\ \perp & \text{sonst} \end{cases}$

Mehr Vokabular wird bei Bedarf vorgestellt