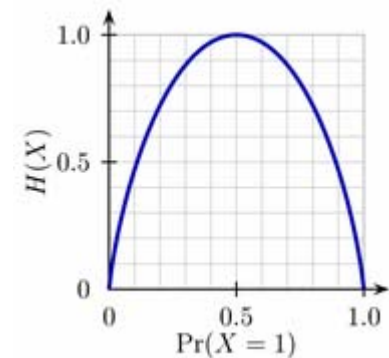


Information und Kommunikation

Hartmut Klauck
Universität Frankfurt
SS 07
8.6.



Kolmogorov Komplexität und Entropie

- Wir haben beim Informationsbegriff eine Ähnlichkeit zur Shannon Entropie gesehen
- Weiterhin gilt: Wenn eine Quelle Symbole gemäß einer Verteilung ausgibt, so dass String x Wahrscheinlichkeit $p(x)$ hat, kann die erwartete Kodierungslänge nicht unter $\sum p(x)C(x)$ sinken (weil sonst Dekodierung unmöglich ist)
- Die optimale Kodierungslänge konvergiert für Quellen von n unabhängigen Zeichen gegen $n H(p)$
- Man kann zeigen, dass auch die Kolmogorov Komplexität der erzeugten Strings gegen $n H(p)$ geht

Präfix Komplexität

- Wir wissen, dass Präfix- und normale Kolmogorov Komplexität höchstens $O(\log n)$ auseinanderliegen können
- $K(x)$ hat einige gute Eigenschaften
 - Additivität ohne log-Term
 - Monoton über Präfixe
- Kraft Ungleichung:
 - Da die erlaubten Programme einen Präfixcode bilden, erfüllen Sie die Kraftsche Ungleichung
- **Theorem 15.1**
 - $\sum_{x \in \{0,1\}^*} 2^{-K(x)} \leq 1$
- Man kann zeigen, dass dies nicht für $C()$ gilt.

Die Ω -Zahl

- Wir definieren $\Omega = \sum_{P \text{ hält auf } \varepsilon} 2^{-|P|}$
- Ω ist die Wahrscheinlichkeit einer TM zu halten, wenn P gemäß einem Bernoulli Prozess gezogen wird
- Programme sind präfixfrei kodiert, daher $\Omega \leq 1$
- Eigenschaften
 - Ω ist nicht berechenbar
 - Ω_n bezeichne die ersten n Bits von Ω ,
 $K(\Omega_n) \geq n - O(1)$

Die Ω -Zahl

- Wenn man Ω kennt, kann man beliebige Theoreme beweisen! (Z.B. Arithmetik)
- Angenommen man kennt die ersten n Bits von Ω , die erhaltene Zahl sei Ω_n
- Wir betrachten ein Theorem, das mit n Bits formal beschrieben werden kann
- Idee: man simuliert parallel (dovetailing) alle Programme, bis mindestens $H = \sum_p$ hält bereits $2^{-|p|} \geq \Omega_n$
- $\Omega - H \leq 2^{-n}$
- Damit können wir für alle Programme der Länge n das Halteproblem lösen: zum Zeitpunkt wo wir die Simulationen stoppen gibt es kein Programm der Länge n mehr das noch halten wird
- Angenommen unser Theorem ist falsch, d.h. es gibt ein Gegenbeispiel. Eine Turingmaschine kann alle Gegenbeispiele aufzählen. Diese Turingmaschine hat Länge $n + O(1)$ und hält, wenn es ein Gegenbeispiel gibt
- Wenn unser Theorem nicht falsch ist, gibt es kein Gegenbeispiel und unsere TM hält nicht.

Teil II

Kommunikationskomplexität

Kommunikationskomplexität

- Wir beginnen mit einem einfachen Spezialfall
- Einweg Kommunikation
- Alice erhält eine Eingabe $x \in \{0,1\}^n$
- Bob erhält eine Eingabe $y \in \{0,1\}^m$
- $f: \{0,1\}^n \times \{0,1\}^m \rightarrow \{0,1\}$ sei eine Funktion
- Ein deterministisches Einweg-Protokoll ist ein Paar von Funktionen
 - $s: \{0,1\}^n \rightarrow \{0,1\}^c$
 - $r: \{0,1\}^m \times \{0,1\}^c \rightarrow \{0,1\}$
- Das Protokoll akzeptiert x, y , wenn $r(y, s(x)) = 1$
- s beschreibt die Nachricht von Alice, r Bobs Entscheidung über die Ausgabe
- c ist die Komplexität des Protokolls P
- Ein Protokoll berechnet f , wenn es für alle x, y die Ausgabe $f(x, y)$ hat

Kommunikationskomplexität

- **Definition 15.2**

- Es sei $D^1(f)$ das Minimum der Komplexitäten aller deterministischen Einweg-Protokolle, die f berechnen

- **Definition 15.3**

- Die Kommunikationsmatrix von $f: \{0,1\}^n \times \{0,1\}^m \rightarrow \{0,1\}$ sei die $2^n \times 2^m$ Matrix M_f mit $M_f(x,y) = f(x,y)$

Ein Beispiel

- Das Gleichheitsproblem
 - $Eq(x,y)=1 \Leftrightarrow x=y$ ($m=n$ in diesem Fall)
- Die Matrix von Eq enthält Einsen auf der Diagonalen und Nullen sonst

Einweg Kommunikation

- **Theorem 15.4**
 - $D^1(f)$ ist gleich $\lceil \log \text{row}(M_f) \rceil$, wenn $\text{row}(M)$ die Anzahl der verschiedenen Zeilen in M bezeichnet.
- **Beweis:**
 - Wir zeigen, dass mindestens $\text{row}(M_f)$ viele verschiedene Nachrichten verwendet werden müssen
 - Angenommen zwei verschiedene Zeilen der Kommunikationsmatrix haben dieselbe Nachricht, d.h. $s(x)=s(x')$. Dann gibt es ein y mit $f(x,y) \neq f(x',y)$. Das Protokoll gibt $r(y,s(x))=r(y,s(x'))$ aus, ist also fehlerhaft.
 - Das es in allen korrekten Protokollen $\text{row}(M_f)$ Nachrichten gibt, gilt für alle korrekten Protokolle $c \geq \lceil \log \text{row}(M_f) \rceil$, und damit das Theorem.

Einweg Kommunikation

- Beispiele:
 - $f(x,y) = \sum x_i + \sum y_i \bmod 2$
 - $D^1(f) = 1$
 - $D^1(\text{Eq}) = n$, da es 2^n Zeilen in M_{Eq} gibt
 - $\text{Maj}(x,y) = 1$, wenn $\sum x_i + \sum y_i \geq n$
wobei $|x| = |y| = n$
 - $D^1(\text{Maj}) \leq \log n + 1$
 - $D^1(\text{Maj}) \geq \log n$

Einweg Kommunikation

- $\text{Index}(x,y): |x|=n, |y|=\log n$
- $\text{Index}(x,y)=x_y$
- Matrix:

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ \vdots & & \\ 1 & 1 & 1 \end{pmatrix}$$

- $D^1(\text{Index})=n$

Anwendung 1

- $L \subseteq \{0,1\}^n$ sei eine Sprache
- L ist endlich, daher regulär, d.h. es existiert ein endlicher Automat, der L entscheidet
- Wir sind interessiert an der Größe endlicher Automaten für L , d.h. der Anzahl Zustände
- **Theorem 15.5**
 - Sei $f_i: \{0,1\}^i \times \{0,1\}^{n-i}$ die Funktion mit $f(x,y)=1 \Leftrightarrow xy \in L$
 - $D_i = D^1(f_i)$
 - Ein endlicher Automat für $L \subseteq \{0,1\}^n$ hat mindestens 2^{D_i} Zustände für alle i .

Anwendung 1

- Beweis:
 - Angenommen es gibt einen Automaten für L mit K Zuständen
 - Ein Protokoll für f_i kann einfach den Automaten simulieren.
 - Alice simuliert den Automaten auf x
 - Alice kommuniziert den erreichten Zustand mit $\log K$ Bits
 - Bob führt die Simulation auf y fort
 - Damit gilt $D^1(f_i) \leq \log K$

Anwendung 1

- Ein Beispiel:
 - $L = \{xy : x, y \in \{0,1\}^n, x \text{ ist } y \text{ rückwärts geschrieben}\}$
 - Menge der Palindrome (grader Länge)
 - Setze $i=n$, dann ist f_i äquivalent zu Eq
 - $D^1(f_i) = n$
 - Jeder endliche Automat für L braucht mindestens 2^n Zustände
 - $O(2^n)$ Zustände reichen auch aus

Anwendung 2

- Formal ist eine Boolesche Formel für eine Funktion $f:\{0,1\}^n\rightarrow\{0,1\}$ gegeben durch einen Baum
 - Die Blätter sind mit Eingaben x_i markiert
 - Alle Knoten haben Eingangsgrad 1 oder 2 und Ausgangsgrad 1 (bis auf die Wurzel)
 - Jeder innere Knoten mit Eingangsgrad i ist mit einer Funktion $g:\{0,1\}^i\rightarrow\{0,1\}$ markiert
 - In einer Berechnung werden die Knoten von den Blättern angefangen in topologischer Reihenfolge ausgewertet
 - Die Auswertung der Wurzel ergibt den Funktionswert

Anwendung 2

- Eine Formel berechnet eine Funktion, wenn sie auf allen Eingaben das korrekte Ergebnis liefert
- Die Länge einer Formel ist die Anzahl der Blätter im Formelbaum.
- **Definition 15.6**
 - Die Formelkomplexität $L(f)$ einer Funktion f ist die minimale Länge einer Formel, welche f berechnet

Anwendung 2

- Formeln sind ein Spezialfall von Schaltkreisen (bei denen der Ausgangsgrad beliebig ist).
- Shannon hat gezeigt, dass fast alle Booleschen Funktionen $f: \{0,1\}^n \rightarrow \{0,1\}$ Schaltkreise die Größe $\Theta(2^n/n)$ haben
- Für Formeln ist für fast alle Funktionen die Komplexität $\Theta(2^n)$
- Problem: Es sind keine expliziten Funktionen bekannt, für die Schaltkreise groß sein müssen (z.B. SAT)
- Beste explizite untere Schranken sind $3n$ (für Paritätsfunktion, wenn Gatterfunktionen nur UND ODER NICHT sind)
- Wir zeigen eine quadratische untere Schranke für Formeln

Anwendung 2

- Gegeben sei also eine Funktion $f:\{0,1\}^n \rightarrow \{0,1\}$
- Wir wollen eine untere Schranke für die Länge von Formeln angeben, die beliebige Gatterfunktionen haben
- S_1, \dots, S_k sei eine beliebige Partition der n Variablen in k Blöcke
- Wir betrachten k Kommunikationsspiele:
 - Alice erhält alle Eingaben außer denen in S_i
 - Bob erhält die Eingaben in S_i
 - f_i sei die Funktion, die sich bei dieser Aufteilung der Variablen wie f verhält
 - $D_i = D^1(f_i)$

Anwendung 2

- Ein Beispiel
 - Die Funktion $ISA(U,X,Y)$
 - „indirect storage access“
 - Eingaben sind in 3 Blöcke
 - $|U| = \log n - \log \log n$
 - $|X| = |Y| = n$
 - X habe $n/\log n$ Blöcke der Länge $\log n$
 - U (als Zahl) indiziert einen Block in X , der Block indiziert ein Bit in Y
 - $ISA(U,X,Y) = Y_{X_U}$

Anwendung 2

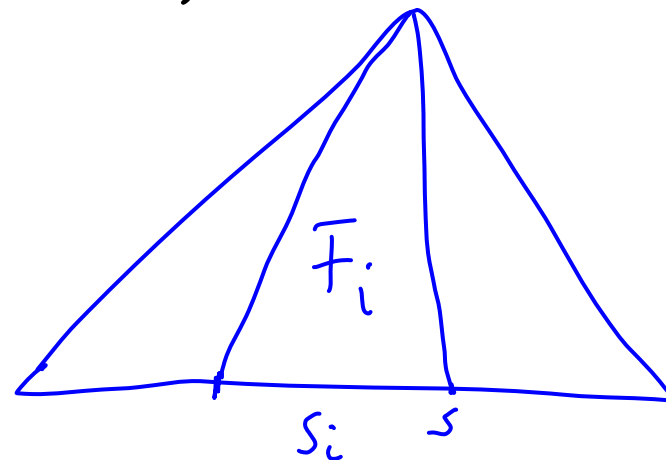
- Wir betrachten $n/\log n$ Kommunikationsspiele für ISA
- Spiel i :
 - Alice erhält U, Y , alle Blöcke von X außer Block i
 - D.h. $n + n - \log n + \log n - \log \log n$ Bits
 - Bob erhält Block i von X
 - Wir setzen $U=i$
- Alle Spiele sind Instanzen der Index Funktion!
- D.h. $D_i = n$ für alle i

Anwendung 2

- Theorem 15.7 [Neciporuk]
 - Für alle Funktionen f , alle Partitionen S_1, \dots, S_k der Eingaben gilt:
 - $L(f) \geq \sum D_i / 4$
- Beispiel: ISA: $n / \log n$ Spiele mit $D_i = n$, d.h. $L(\text{ISA}) \geq n^2 / (4 \log n)$
- Bemerkung: das ist die bestmögliche Schranke mit dieser Methode
- Keine größeren Schranken bekannt wenn beliebige Gatterfunktionen erlaubt sind

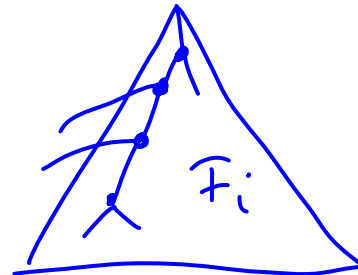
Anwendung 2

- Beweis:
- Sei F eine Formel für f mit Länge L .
- Wir zeigen, wie F effizient durch k Einwegprotokolle simuliert werden kann
- In Spiel i kennt Alice alle Eingaben außer denen in S_i
- F_i sei die Teilformel, deren Blätter in S_i liegen (Blätter und alle Pfade bis zur Wurzel)



Anwendung 2

- V_i sei die Menge der Knoten in F_i , die alle Vorgänger in F_i haben
- P_i sei die Menge aller Pfade, welche in V_i starten, und in V_i enden (oder an der Wurzel), aber dazwischen keine Knoten aus V_i enthalten



Anwendung 2

- Das oberste Gatter nicht in V_i für einen Pfad in P_i berechnet abhängig von Alices Eingabe entweder $0, 1, g, \neg g$ für die Funktion g , die vom untersten Gatter (dem in V_i) des Pfades berechnet wird
- Daher reicht es aus, für jeden Pfad 2 Bits zu kommunizieren
- Dann kann Bob die Formel bis zur Wurzel auswerten

Anwendung 2

- Die Anzahl der Pfade in P_i ist höchstens $2|V_i|+1$
- Damit ist die Kommunikation in Spiel i höchstens $4|V_i|+2$
- Die Anzahl der Blätter mit Variablen in S_i ist $|V_i|+1$
- Daher gilt: $D_i = D^1(f_i) \leq 4|V_i|+2 \leq 4|L_i|$
für die Anzahl L_i der Blätter mit Variablen in S_i
- $\sum L_i = L$, und damit $L \geq \sum D_i / 4$