

# 3 Randomisierte Protokolle

## 3.4 Approximierende Protokolle

Wir stellen in diesem Abschnitt mit *Yaos Minimax-Prinzip* ein allgemeines Prinzip zum Nachweis von unteren Schranken für randomisierte Protokolle mit zweiseitigem Fehler vor, das wir dann später auf verschiedene Weisen zu kompletten Techniken ausbauen. Die Grundidee dabei ist es, aus einem randomisierten Protokoll mit kleiner Fehlerwahrscheinlichkeit ein *deterministisches* Protokoll mit höchstens gleich großer Komplexität abzuleiten, das für einen kleinen Teil der Eingaben falsch rechnen darf.

**Definition:** Sei eine Funktion  $f: X \times Y \rightarrow Z$  gegeben und sei  $\mu$  eine Wahrscheinlichkeitsverteilung auf  $X \times Y$ . Ein deterministisches Protokoll *approximiert*  $f$  mit Fehler  $\varepsilon$  bezüglich  $\mu$  (bzw. heißt *approximierendes Protokoll für  $f$  mit Fehler  $\varepsilon$  bezüglich  $\mu$* ), wenn es höchstens auf einem  $\varepsilon$ -Anteil aller Eingaben bezüglich  $\mu$  einen von  $f$  abweichenden Wert liefert. Mit  $D_{\mu,\varepsilon}(f)$  bezeichnen wir die minimale Komplexität eines Protokolls, das  $f$  mit Fehler  $\varepsilon$  bezüglich  $\mu$  approximiert.

Wir ersetzen also die Wahrscheinlichkeitsverteilung über den Zufallsbits von Alice und Bob durch eine Wahrscheinlichkeitsverteilung über den *Eingaben*.

Wir bezeichnen mit „uniform“ die Gleichverteilung auf dem jeweils betrachteten (sich aus dem Kontext ergebenden) Definitionsbereich. Wir beschreiben zunächst die grundsätzliche Idee für den Nachweis von unteren Schranken für den Fall der Gleichverteilung. Wir betrachten ein randomisiertes Protokoll für eine Funktion  $f: X \times Y \rightarrow \{0, 1\}$  mit Fehler  $\varepsilon$  und insgesamt  $\ell$  Zufallsbits für Alice und Bob als Gleichverteilung über deterministischen Protokollen  $P_1, \dots, P_{2^\ell}$ . Wir definieren die *Fehlermatrix*  $F$  des Protokolls wie folgt: Für  $x \in X$  und  $y \in Y$  sei  $F((x, y), P_i) = 1$ , falls  $P_i(x, y) \neq f(x, y)$  und  $F((x, y), P_i) = 0$  sonst. Dann gibt es aufgrund der Fehlerschranke des Protokolls in jeder Zeile  $(x, y)$  von  $F$  höchstens einen  $\varepsilon$ -Anteil Einsen. Damit hat die Matrix auch insgesamt höchstens einen  $\varepsilon$ -Anteil von Einsen, woraus wiederum folgt, dass es mindestens eine Spalte  $P_i$  mit höchstens einem  $\varepsilon$ -Anteil von Einsen geben muss. Also ist dieses  $P_i$  ein approximierendes Protokoll für  $f$  mit Fehler  $\varepsilon$  und mit höchstens so hoher Komplexität wie das ursprüngliche Protokoll. Wir haben also gezeigt, dass

$$D_{\text{uniform},\varepsilon}(f) \leq R_\varepsilon(f).$$

Wir erhalten damit untere Schranken für randomisierte Protokolle mit zweiseitigem Fehler, wenn wir in der Lage sind, solche Schranken für approximierende Protokolle nachzuweisen. Tatsächlich lässt sich sogar die randomisierte Kommunikationskomplexität mit *öffentlichen* Zufallsbits genau durch die Komplexität approximierender Protokolle charakterisieren.

**Satz (Yaos Minimax-Prinzip):**

- (1)  $R_\varepsilon^{\text{pub}} \geq \max\{D_{\mu,\varepsilon}(f) \mid \mu \text{ Verteilung auf } X \times Y\}$ .
- (2) Für alle  $\delta > 0$  gilt:  $R_{\varepsilon+\delta}^{\text{pub}} \leq \max\{D_{\mu,\varepsilon}(f) \mid \mu \text{ Verteilung auf } X \times Y\}$ .

**Beweis:** *Teil (1):* Dieser Teil ergibt sich sofort aus den Vorüberlegungen oben, da diese auch für randomisierte Protokolle mit öffentlichem Zufall funktionieren und analog auch mit beliebigen Verteilungen anstelle der Gleichverteilung.

*Teil (2):* Sei  $c := \max_{\mu} D_{\mu, \varepsilon}(f)$ . Wir betrachten folgendes Zwei-Personen-Nullsummenspiel. Alice wählt ein deterministisches Protokoll  $P$  der Komplexität  $c$ , Bob eine Eingabe  $(x, y)$ . Alice zahlt an Bob 1, falls  $P$  auf  $(x, y)$  einen Fehler macht und sonst nichts. Aufgrund der Definition von  $c$  gibt es für jede randomisierte Strategie  $\mu$  von Bob (die eine Wahrscheinlichkeitsverteilung über den Eingaben darstellt) eine deterministische Strategie von Alice (die ein  $\varepsilon$ -Fehler-Protokoll mit Komplexität höchstens  $c$  darstellt), für die sie im Durchschnitt höchstens  $\varepsilon$  zahlen muss. Aufgrund des Minimax-Theorems der Spieltheorie gibt es dann eine randomisierte Strategie für Alice (eine Verteilung über  $c$ -Bit-Protokollen), sodass sie für jede beliebige, insbesondere deterministische Strategie von Bob (also für jede Eingabe  $(x, y)$ ) höchstens  $\varepsilon$  zahlen muss. Die randomisierte Strategie von Alice lässt sich durch Runden der Wahrscheinlichkeiten auf Binärzahlen endlicher Länge in ein randomisiertes Protokoll mit öffentlichem Zufall umwandeln, das  $f$  mit Fehler  $\varepsilon + \delta$  für beliebig kleine  $\delta > 0$  berechnet.  $\square$

Es bleibt nun die Aufgabe, untere Schranken für approximierende Protokolle mit kleinem Fehler nachzuweisen. In den nächsten zwei Abschnitten stellen wir entsprechende Techniken dafür vor.

### 3.5 Korruption

Die Korruptionstechnik greift die Idee der Rechteckmaßmethode zum Nachweis von unteren Schranken für nichtdeterministische Protokolle aus Kapitel 2 auf. Dabei ging es darum, zu zeigen, dass jedes 1-Rechteck für die betrachtete Funktion nur kleines Gewicht bezüglich einer geschickt gewählten Eingabeverteilung hat, während andererseits die 1-Eingaben der Funktion hohes Gewicht bezüglich dieser Verteilung haben. Da die 1-Eingaben bei nichtdeterministischen Protokollen exakt von 1-Rechtecken überdeckt werden müssen, ergab sich daraus direkt eine entsprechende untere Schranke für die Anzahl benötigter 1-Rechtecke.

Bei approximierenden Protokollen (mit zweiseitigem Fehler) werden die 0- bzw. 1-Eingaben im Allgemeinen nicht mehr exakt von entsprechend gefärbten Rechtecken überdeckt. Typischerweise haben wir Rechtecke, auf denen zwar größtenteils richtig gerechnet wird, die aber durch einige falsch gefärbte Eingaben „verunreinigt“ (engl. *corrupted*) sind.

**Definition:** Sei  $f: X \times Y \rightarrow \{0, 1\}$  und eine Wahrscheinlichkeitsverteilung  $\mu$  auf  $X \times Y$  gegeben. Sei  $R$  ein Rechteck in  $X \times Y$  und sei  $c \in \{0, 1\}$ . Dann heißt  $R$   $\varepsilon$ -Fehler- $c$ -Rechteck für  $f$  bezüglich  $\mu$ , falls  $\mu(R \cap f^{-1}(\bar{c})) \leq \varepsilon \cdot \mu(R)$ .

Wir zeigen im Folgenden, dass man die Idee der Rechteckmaßmethode auf diese leicht verunreinigten Rechtecke verallgemeinern kann.

**Satz (Korruptionstechnik):** Sei  $f: X \times Y \rightarrow \{0, 1\}$  und eine Wahrscheinlichkeitsverteilung  $\mu$  auf  $X \times Y$  von  $f$  gegeben. Sei  $\mu(f^{-1}(1)) > 0$  und  $\varepsilon/\mu(f^{-1}(1)) \leq \varepsilon' \leq 1$  und sei  $\rho$  das größte Gewicht eines  $\varepsilon'$ -Fehler-1-Rechtecks für  $f$  bezüglich  $\mu$ . Dann gilt

$$D_{\mu, \varepsilon}(f) \geq \log[(\mu(f^{-1}(1)) - \varepsilon/\varepsilon')/\rho]$$

**Beweis:** Sei  $P$  ein deterministisches Protokoll, das  $f$  mit Fehler  $\varepsilon$  bezüglich  $\mu$  approximiert. Sei  $g$  die von  $P$  berechnete Funktion. Dann stimmt  $g$  höchstens auf einem  $\varepsilon$ -Anteil aller Eingaben bezüglich  $\mu$  nicht mit  $f$  überein. O. B. d. A. ist  $\mu(g^{-1}(1)) > 0$ , da sonst  $\varepsilon = \mu(f^{-1}(1))$ ,  $\varepsilon' = 1$  und damit die untere Schranke 0 und trivialerweise erfüllt. Seien  $R_1, \dots, R_k$  (mit  $k \geq 1$ ) die 1-Rechtecke von  $P$ . Unser Ziel ist es zu zeigen, dass

$$k \geq (\mu(f^{-1}(1)) - \varepsilon/\varepsilon')/\rho$$

gilt, woraus offensichtlich die behauptete untere Schranke folgt.

Definiere

$$\varepsilon_0 := \sum_{i=1}^k \mu(R_i \cap f^{-1}(0)) \quad \text{und} \quad \varepsilon_1 := \mu(f^{-1}(1)) - \mu(f^{-1}(1) \cap g^{-1}(1)).$$

Dies sind die Gewichte der Eingaben, für die  $f$  die Ausgabe 0 berechnet, das Protokoll  $P$  jedoch fälschlicherweise 1 ausgibt bzw. umgekehrt. Daher muss aufgrund der Fehlerschranke  $\varepsilon_0 + \varepsilon_1 \leq \varepsilon$  gelten.

Für  $i = 1, \dots, k$  sei  $\rho_i := \mu(R_i \cap f^{-1}(0))/\mu(R_i)$ , d. h.  $\rho_i$  ist der relative Fehler des Protokolls auf dem Rechteck  $R_i$ . Durch Dividieren der Definition von  $\varepsilon_0$  durch  $\mu(g^{-1}(1))$  ergibt sich:

$$\sum_{i=1}^k \rho_i \cdot \frac{\mu(R_i)}{\mu(g^{-1}(1))} = \frac{\varepsilon_0}{\mu(g^{-1}(1))}.$$

Die Zahlen  $\mu(R_i)/\mu(g^{-1}(1))$ ,  $i = 1, \dots, k$ , definieren offensichtlich eine Wahrscheinlichkeitsverteilung auf  $\{1, \dots, k\}$ . Die Gleichung oben besagt, dass der Erwartungswert der  $\rho_1, \dots, \rho_k$  bezüglich dieser Verteilung  $\varepsilon_0/\mu(g^{-1}(1))$  ist. Sei  $I := \{i \mid \rho_i \leq \varepsilon'\}$ . Gemäß Markoffungleichung folgt dann, dass

$$\sum_{i \notin I} \frac{\mu(R_i)}{\mu(g^{-1}(1))} \leq \frac{\varepsilon_0}{\mu(g^{-1}(1))} \cdot \frac{1}{\varepsilon'} \Leftrightarrow \sum_{i \in I} \mu(R_i) \geq \mu(g^{-1}(1)) - \frac{\varepsilon_0}{\varepsilon'}.$$

Für alle  $i \in I$  gilt nun aber aufgrund der Definition von  $\rho$ , dass  $\mu(R_i) \leq \rho$  ist. Also folgt

$$\sum_{i \in I} \mu(R_i) \leq k \cdot \rho.$$

Zusammensetzen der letzten beiden Ungleichungen und der sich aus der Definition  $\varepsilon_1$  ergebenden Abschätzung

$$\mu(g^{-1}(1)) \geq \mu(f^{-1}(1) \cap g^{-1}(1)) = \mu(f^{-1}(1)) - \varepsilon_1$$

liefert

$$k \geq (\mu(f^{-1}(1)) - \varepsilon_1 - \varepsilon_0/\varepsilon')/\rho.$$

Unter der Nebenbedingung  $\varepsilon_0 + \varepsilon_1 \leq \varepsilon$  wird die rechte Seite als Funktion der Variablen  $\varepsilon_0$  und  $\varepsilon_1$  minimal für  $\varepsilon_0 = \varepsilon$  und  $\varepsilon_1 = 0$ , d. h.

$$k \geq (\mu(f^{-1}(1)) - \varepsilon/\varepsilon')/\rho,$$

was zu zeigen war. □

Ein typisches Beispiel für die Anwendung dieser Technik liefert die Disjunktheitsfunktion  $\text{DISJ}_n$ . Es ist nicht ganz einfach, eine passende Eingabeverteilung zu finden. Wir benötigen, dass es viele 1-Eingaben gibt, aber andererseits jedes Rechteck, das fast nur Einsen enthält, klein sein muss (bzw. äquivalent dazu jedes nicht ganz kleine Rechteck durch viele Nullen verunreinigt ist). Analog können wir natürlich auch (zweiseitiger Fehler) die 0-Eingaben betrachten. Wir beobachten, dass es bezüglich der Gleichverteilung große 0-Rechtecke gibt (es reicht, wenn  $x_i = y_i = 1$  sind und die restlichen Bits beliebig belegt werden) und andererseits die Funktion nur wenige 1-Eingaben hat, denn für zwei zufällig gleichverteilt gewählte Mengen  $x, y \subseteq \{1, \dots, n\}$  gilt

$$\Pr_{x,y}\{x \cap y = \emptyset\} = 3^n/2^{2n} \xrightarrow{n \rightarrow \infty} 0.$$

Die Gleichverteilung ist also für die Korruptionsmethode nicht geeignet.

Wir wollen die Eingabeverteilung so wählen, dass es sowohl einen konstanten Anteil 0-Eingaben als auch einen konstanten Anteil 1-Eingaben gibt. Wir müssen dann weiterhin noch eine gute obere Schranke für das Gewicht von  $\varepsilon$ -Fehler-1-Rechtecken zeigen. Wir stellen im Folgenden zwei verschiedenen Eingabeverteilungen vor, die in der Literatur untersucht wurden.

- **Babai-Frankl-Simon (1986).** Bei dieser Verteilung werden  $x, y$  unabhängig voneinander zufällig gleichverteilt unter allen Mengen der Größe  $\sqrt{n}$  gewählt (der Einfachheit halber sei  $n$  eine Quadratzahl). Dann gilt

$$\Pr_{x,y}\{x \cap y = \emptyset\} = \frac{\binom{n}{\sqrt{n}} \binom{n-\sqrt{n}}{\sqrt{n}}}{\binom{n}{\sqrt{n}}^2} \xrightarrow{n \rightarrow \infty} e^{-1},$$

wobei sich der Grenzwert durch Abschätzung der Binomialkoeffizienten mit stirlingscher Formel ergibt. Weder der Anteil der 0-Eingaben noch der der 1-Eingaben konvergiert also gegen 0. Damit ist allerdings noch nichts über die Größe von  $\varepsilon$ -Fehler-Rechtecken gesagt. Tatsächlich haben Babai, Frankl und Simon mit kombinatorischen Argumenten gezeigt, dass für eine hinreichend kleine Konstante  $\varepsilon > 0$  jedes  $\varepsilon$ -Fehler-1-Rechteck bezüglich der soeben definierten Verteilung  $\mu$  Gewicht  $2^{-\Omega(\sqrt{n})}$  hat. Damit liefert der Satz über die Korruptionstechnik die untere Schranke  $D_{\mu,\varepsilon}(\text{DISJ}_n) = \Omega(\sqrt{n})$  und zusammen mit Yaos Minimax-Prinzip haben wir auch  $R(\text{DISJ}_n) = \Omega(\sqrt{n})$ .

Das besondere an der erhaltenen Schranke für die approximierenden Protokolle ist, dass sie sogar speziell für eine Verteilung erzielt wurde, bei der Alice und Bob ihre Eingaben unabhängig voneinander wählen. Solche Verteilungen heißen *Rechteckverteilungen*. Andererseits konstruieren Babai, Frankl und Simon auch für beliebige Rechteckverteilungen approximierende Protokolle, die Komplexität  $O(\sqrt{n} \log n)$  haben. Prinzipiell ist es also nicht möglich, mit Hilfe solcher Verteilungen wesentlich bessere als wurzelige untere Schranken für  $\text{DISJ}_n$  zu zeigen.

- **Razborov (1991).** Es wurde seit den ersten Arbeiten über Kommunikationskomplexität vermutet, dass  $\text{DISJ}_n$  lineare randomisierte Komplexität erfordert, aber erst 1991 gelang unabhängig voneinander Kalyanasundaram und Schnitger bzw. Razborov der Nachweis. Razborovs Beweis ist eine direkte Anwendung der Korruptionsmethode mit einer expliziten Eingabeverteilung (dies kann zwangsläufig keine Rechteckverteilung mehr sein). Um diese zu definieren, sei  $n = 4\ell - 1$  für ein  $\ell \in \mathbb{N}$ . Wir wählen zunächst zufällig gleichverteilt disjunkte Teilmengen  $T_1, T_2, \{i\} \subseteq \{1, \dots, n\}$ , wobei  $|T_1| = |T_2| = 2\ell - 1$  sei. Wir wählen dann  $x \subseteq T_1 \cup \{i\}$  und  $y \subseteq T_2 \cup \{i\}$  zufällig gleichverteilt mit  $|x| = |y| = \ell$ . Es ist

$$\mu(\text{DISJ}_n^{-1}(0)) = \frac{\binom{2\ell-1}{\ell-1}^2}{\binom{2\ell}{\ell}^2} = \left(\frac{\ell}{2\ell}\right)^2 = \frac{1}{4}$$

und damit auch  $\mu(\text{DISJ}_n^{-1}(1)) = 3/4$ . Razborov hat gezeigt (siehe auch Abschnitt 4.6 im Buch von Kushilevitz und Nisan), dass für eine hinreichend kleine Konstante  $\varepsilon > 0$  jedes  $\varepsilon$ -Fehler-1-Rechteck für  $\text{DISJ}_n$  bezüglich  $\mu$  Gewicht höchstens  $2^{-\Omega(n)}$  hat. Daraus folgt  $D_{\mu, \varepsilon}(\text{DISJ}_n) = \Omega(n)$  und  $R(\text{DISJ}_n) = \Omega(n)$ .

Wir sparen uns an dieser Stelle die entsprechenden Beweise der Rechteckmaß-Schranken, da wir im Kapitel 5 noch einmal ausführlich mit einer moderneren Methode (basierend auf Informationskomplexität) nachweisen werden, dass die Disjunktheitsfunktion lineare randomisierte Kommunikationskomplexität erfordert.



# 4 Informationstheoretische Grundlagen

## 4.1 Entropie

Die (*Shannon-*)*Entropie* misst intuitiv die Unsicherheit über den Ausgang eines Zufallsexperiments, bevor wir dessen Ergebnis kennen bzw. etwas formaler die Unsicherheit über den Wert einer Zufallsvariablen. Dual dazu ist die Sichtweise als die Menge an *Information*, die wir gewinnen, wenn wir den Ausgang des Zufallsexperiments erfahren.

**Definition 4.1:** Sei  $X$  eine Zufallsvariable mit endlichem Wertebereich  $R$  und für  $x \in R$  sei  $p(x) := \Pr\{X = x\}$ . Dann ist die *Entropie von  $X$* ,  $H(X)$ , definiert gemäß

$$H(X) := - \sum_{x \in R} p(x) \log p(x),$$

wobei  $\log$  den Logarithmus zur Basis 2 bezeichnet. Alternativ schreibt man bei gegebener Wahrscheinlichkeitsverteilung  $p: R \rightarrow [0, 1]$  auch  $H(p)$  für die Entropie von  $p$ .

Wir erlauben in dieser Definition auch den Fall  $p(x) = 0$ , indem wir  $0 \cdot \log 0 := 0$  definieren (dies kann man dadurch rechtfertigen, dass  $\lim_{x \rightarrow 0} x \log x = 0$  ist). Folgende Eigenschaften ergeben sich direkt aus der Definition:

**Fakt 4.2:**

- (1) Sei  $U$  die Gleichverteilung auf  $R$ . Dann gilt  $H(U) = \log |R|$ .
- (2) Für beliebige Zufallsvariablen  $X$  gilt  $H(X) \geq 0$ .

Wir werden später auch zeigen, dass  $H(X) \leq \log |R|$  gilt mit Gleichheit genau dann, wenn  $X$  die Gleichverteilung auf  $R$  ist.

Die Definition der Entropie kann man axiomatisch begründen (siehe Cover und Thomas, Chapter 2, Exercise 4). Praktisch interessanter ist die Tatsache, dass man formal nachweisen kann, dass für die erwartete Länge  $\ell(X)$  einer Codierung der Zufallsvariablen  $X$  durch Bitstrings gilt:

$$H(X) \leq \ell(X) < H(X) + 1.$$

Dies ist der Inhalt des *Noiseless-Coding-Theorems* von Shannon. Anders ausgedrückt: Wenn wir einen Strom von unabhängigen, wie  $X$  verteilten Datensymbolen in binär codierter Form über einen fehlerfreien Kanal so übertragen wollen, dass diese aus den Codewörtern eindeutig rückgewinnbar sind, dann sind dazu mindestens  $H(X)$  Bits notwendig und maximal  $H(X) + 1$  Bits reichen aus. Die Entropie gibt also insbesondere eine absolute theoretische untere Schranke für mögliche Datenkompression vor. Für Details dazu und zu vielen weiteren Anwendungen des Entropiebegriffs, die wir hier nicht diskutieren können, siehe das Buch von Cover und Thomas.

**Definition 4.3:** Die *binäre Entropie*  $H_2(p)$  ist die Entropie einer 0-1-wertigen Zufallsvariablen  $X$  mit  $\Pr\{X = 1\} = p$  (und damit  $\Pr\{X = 0\} = 1 - p$ ), also

$$H_2(p) = -p \log p - (1 - p) \log(1 - p).$$

Offensichtlich ist  $H(0) = H(1) = 0$  und  $H(1/2) = 1$ . Der Funktionsgraph für den gesamten Definitionsbereich sieht so aus:

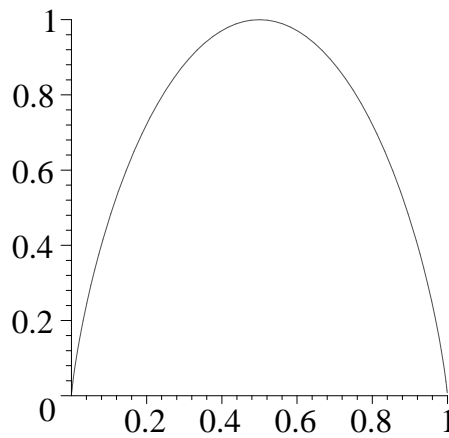


Abbildung 4.1: Binäre Entropiefunktion.

**Definition 4.4:** Seien  $X, Y$  Zufallsvariablen mit endlichen Wertebereichen  $R, S$ . Dann definieren wir

(1) die *gemeinsame Entropie von  $X$  und  $Y$*  als

$$H(X, Y) := - \sum_{x \in R, y \in S} \Pr\{X = x, Y = y\} \log \Pr\{X = x, Y = y\}.$$

(2) die *bedingte Entropie von  $Y$  gegeben  $X$*  als

$$H(Y | X) := \sum_{x \in R} \Pr\{X = x\} \cdot H(Y | X = x).$$

Der Ausdruck  $H(Y | X = x)$  bezeichnet dabei nichts weiter als die Entropie der Zufallsvariablen  $Y$  unter der Bedingung  $X = x$ , d. h. der Wahrscheinlichkeitsverteilung  $p$  mit

$$p(y) = \Pr\{Y = y | X = x\} = \frac{\Pr\{Y = y, X = x\}}{\Pr\{X = x\}}.$$

Wir halten noch folgende Umformulierung der Definition fest.

$$\begin{aligned} H(Y | X) &= \sum_{x \in R} \Pr\{X = x\} \sum_{y \in S} \Pr\{Y = y | X = x\} \log \Pr\{Y = y | X = x\} \\ &= \sum_{x \in R, y \in S} \Pr\{X = x, Y = y\} \log \Pr\{Y = y | X = x\}. \end{aligned}$$

Als eine einfache Anwendung der Definition zeigen wir:

**Satz 4.5:** Sei  $X$  eine Zufallsvariable mit endlichem Wertebereich  $R$ . Sei  $f: R \rightarrow S$  eine beliebige Funktion. Dann gilt  $H(f(X) | X) = 0$ .

**Beweis:** Anwenden der Definition liefert

$$H(f(X) | X) = \sum_{x \in R} \Pr\{X = x\} \cdot H(f(X) | X = x).$$

Die Zufallsvariable  $f(X)$  nimmt unter Bedingung  $X = x$  den festen Wert  $f(x)$  an, d. h.  $H(f(X) | X = x) = 0$  für alle  $x$ . Daraus folgt die Behauptung.  $\square$

Die Begriffe der gemeinsamen und bedingten Entropie hängen durch folgende wichtige und auch intuitiv einleuchtende Formel zusammen:

**Satz 4.6 (Kettenregel für die Entropie):**  $H(X, Y) = H(X) + H(Y | X) = H(Y) + H(X | Y)$ .

**Beweis:** Es reicht, die erste Gleichheit zu zeigen:

$$\begin{aligned} H(X, Y) &= - \sum_{x,y} \Pr\{X = x, Y = y\} \log \Pr\{X = x, Y = y\} \\ &= - \sum_{x,y} \Pr\{X = x, Y = y\} \log(\Pr\{X = x\} \Pr\{Y = y | X = x\}) \\ &= - \sum_{x,y} \Pr\{X = x, Y = y\} \log \Pr\{X = x\} \\ &\quad - \sum_{x,y} \Pr\{X = x, Y = y\} \log \Pr\{Y = y | X = x\} \\ &= - \sum_x \Pr\{X = x\} \log \Pr\{X = x\} - \sum_{x,y} \Pr\{X = x, Y = y\} \log \Pr\{Y = y | X = x\} \\ &= H(X) - H(Y | X). \end{aligned}$$

$\square$

Wir bemerken noch, dass sich die behandelten Begriffe auf nahe liegende Weise auf mehr als zwei Zufallsvariablen fortsetzen, z. B. ist  $H(X_1, \dots, X_n)$  die Entropie der gemeinsamen Verteilung der Zufallsvariablen  $X_1, \dots, X_n$ . Auch bedingte Entropien mit mehreren Bedingungen, z. B.  $H(X | Y_1, \dots, Y_n)$ , sind durch die obige Definition mit erfasst: Wir interpretieren  $(Y_1, \dots, Y_n)$  als einzelne Zufallsvariable mit der gemeinsamen Verteilung von  $Y_1, \dots, Y_n$ .

Schließlich bleiben die bereits hergeleiteten Gleichungen und Ungleichungen für Entropien gültig, wenn auf beiden Seiten überall jeweils die gleiche Bedingung hinzugefügt wird. Dies folgt direkt durch Anwenden der Definition der bedingten Entropie (Durchschnittsbildung auf beiden Seiten). Z. B. erhalten wir so eine bedingte Version der Kettenregel,

$$H(X, Y | Z) = H(X | Z) + H(Y | X, Z) = H(Y | Z) + H(X | Y, Z).$$

## 4.2 Relative Entropie

Die relative Entropie ist eines von vielen Maßen, mit denen Wahrscheinlichkeitsverteilungen verglichen werden können. Wir führen den Begriff hier ein, da sich damit einige wichtige Beziehungen zwischen den Entropien von zwei (oder mehr) Zufallsvariablen auf elegante Weise herleiten lassen.

**Definition 4.7:** Seien  $p$  und  $q$  Wahrscheinlichkeitsverteilungen auf dem endlichen Wertebereich  $R$ . Dann ist die *relative Entropie* zwischen  $p$  und  $q$  (auch *Kullback-Leibler-Divergenz* von  $p$  und  $q$  genannt) definiert als

$$\text{KL}(p, q) := \sum_{x \in R} p(x) \log \frac{p(x)}{q(x)}.$$

Wir benutzen dabei die Konventionen „ $0 \cdot \log \frac{0}{q(x)} = 0$ “ und „ $p(x) \cdot \log \frac{p(x)}{0} = \infty$ “. Wir werden zeigen, dass immer  $\text{KL}(p, q) \geq 0$  gilt mit Gleichheit genau dann, wenn  $p = q$ . Intuitiv ist es hilfreich, die relative Entropie als Maß für die Verschiedenheit zwischen den Verteilungen  $p$  und  $q$  aufzufassen. Allerdings ist die relative Entropie keine Metrik, da sie weder symmetrisch in  $p$  und  $q$  ist, noch die Dreiecksungleichung erfüllt (wie man an einfachen Beispielen sehen kann).

**Beispiel:** Betrachte zwei Verteilungen  $p$  und  $q$  auf  $\{0, 1\}$  mit  $p(0) = p(1) = 1/2$  und  $q(0) = 1/4, q(1) = 3/4$ . Dann gilt

$$\begin{aligned} \text{KL}(p, q) &= \frac{1}{2} \left( \log \frac{1/2}{1/4} + \log \frac{1/2}{3/4} \right) = \frac{1}{2} \left( 1 + \log \frac{2}{3} \right) \approx 0,208 \quad \text{und} \\ \text{KL}(q, p) &= \frac{1}{4} \log \frac{1/4}{1/2} + \frac{3}{4} \log \frac{3/4}{1/2} = -\frac{1}{4} + \frac{3}{4} \log \frac{3}{2} \approx 0,189. \end{aligned}$$

**Satz 4.8 (Nichtnegativität der relativen Entropie):** Für beliebige Wahrscheinlichkeitsverteilungen  $p, q$  auf dem endlichen Wertebereich  $R$  gilt  $\text{KL}(p, q) \geq 0$  mit Gleichheit genau dann, wenn  $p = q$ .

**Beweis:** Wir benutzen die aus der Analysis bekannte Tatsache, dass für alle  $x > 0$  gilt:  $\ln x \leq x - 1$ , wobei Gleichheit genau für  $x = 1$  eintritt. Außerdem ist  $\log x = \frac{\ln x}{\ln 2}$ . O. B. d. A. nehmen wir an, dass  $p(x) > 0$  für alle  $x \in R$ . Damit erhalten wir

$$\begin{aligned} \text{KL}(p, q) &= \sum_{x \in R} p(x) \left( -\log \frac{q(x)}{p(x)} \right) = \frac{1}{\ln 2} \sum_{x \in R} p(x) \left( -\ln \frac{q(x)}{p(x)} \right) \\ &\geq \frac{1}{\ln 2} \sum_{x \in R} p(x) \left( 1 - \frac{q(x)}{p(x)} \right) = \frac{1}{\ln 2} \sum_{x \in R} (p(x) - q(x)) = \frac{1}{\ln 2} (1 - 1) = 0. \end{aligned}$$

Damit ist die Nichtnegativität gezeigt. Gleichheit tritt nur dann ein, wenn bei den Abschätzungen für alle  $x \in R$  Gleichheit vorliegt, d. h. wenn für alle  $x \in R$  gilt, dass  $q(x)/p(x) = 1$ .  $\square$

**Folgerung 4.9:** Sei  $X$  eine Zufallsvariable mit Wertebereich  $R$ . Dann gilt  $H(X) \leq \log |R|$ .

**Beweis:** Sei  $p$  die Verteilung von  $X$  und  $u$  die Gleichverteilung auf  $R$ . Dann ist

$$\text{KL}(p, u) = \sum_{x \in R} p(x) \log \frac{p(x)}{1/|R|} = \log |R| - H(X).$$

Damit folgt die Behauptung aus Satz 4.8. □

### 4.3 Wechselseitige Information

Unser Ziel ist es hier, die Information (bzw. Reduktion der Entropie) zu messen, die eine Zufallsvariable über eine andere liefert.

**Definition 4.10:** Für zwei Zufallsvariablen  $X, Y$  mit endlichem Wertebereich definieren wir die *wechselseitige Information* zwischen  $X$  und  $Y$ ,  $I(X : Y)$ , durch  $I(X : Y) := H(X) + H(Y) - H(X, Y)$ .

Offensichtlich ist das neu definierte Maß symmetrisch in  $X$  und  $Y$ , d. h.  $X$  liefert über  $Y$  genausoviel Information wie  $Y$  über  $X$ . Mit der Kettenregel für die Entropie ergibt sich aus der Definition die folgende, leichter intuitiv interpretierbare Charakterisierung der wechselseitigen Information:

**Fakt 4.11:**  $I(X : Y) = H(X) - H(X | Y) = H(Y) - H(Y | X)$ .

Die bedingte Entropie  $H(X | Y)$  misst, wieviel Unsicherheit über  $X$  verbleibt, wenn wir  $Y$  kennen lernen. Das Maß  $I(X : Y)$  gibt damit die Reduktion der Unsicherheit über  $X$  an, die stattfindet, wenn wir  $Y$  kennen lernen. Analoges gilt mit vertauschten Rollen von  $X$  und  $Y$ .

Weiterhin können wir die wechselseitige Information auch mit Hilfe der relativen Entropie charakterisieren:

**Satz 4.12:** Für zwei Zufallsvariablen  $X$  und  $Y$  seien  $p_X, p_Y$  die zugehörigen Verteilungen, es sei  $p$  deren gemeinsame Verteilung (d. h.  $p(x, y) = \Pr\{X = x, Y = y\}$  für alle  $x, y$ ) und es sei  $p_X \otimes p_Y$  die Produktverteilung von  $p_X$  und  $p_Y$  ( $(p_X \otimes p_Y)(x, y) = \Pr\{X = x\} \cdot \Pr\{Y = y\}$  für alle  $x, y$ ). Dann gilt  $I(X : Y) = \text{KL}(p, p_X \otimes p_Y)$ .

Die wechselseitige Information misst also den Unterschied zwischen der tatsächlich vorliegenden gemeinsamen Verteilung von  $X$  und  $Y$  und der Verteilung, die im Falle von Unabhängigkeit vorliegen würde.

**Beweis:**

$$\begin{aligned} \text{KL}(p, p_X \otimes p_Y) &= \sum_{x,y} p(x, y) \log \frac{p(x, y)}{p_X(x)p_Y(y)} \\ &= -\sum_{x,y} p(x, y) \log p_X(x) - \sum_{x,y} p(x, y) \log p_Y(y) + \sum_{x,y} p(x, y) \log p(x, y) \\ &= H(X) + H(Y) - H(X, Y) = I(X : Y). \end{aligned}$$

□

Satz 4.12 liefert zusammen mit der Nichtnegativität der relativen Entropie (Satz 4.8) sofort:

**Satz 4.13 (Nichtnegativität der wechselseitigen Information):**  $I(X, Y) \geq 0$  mit Gleichheit genau dann, wenn  $X$  und  $Y$  unabhängig sind.

Das folgende Venn-Diagramm veranschaulicht die Beziehung zwischen den wichtigsten bisher behandelten Maßen (wobei man dies nur als Merkhilfe benutzen und nicht wörtlich nehmen sollte, da die auftauchenden Objekte gar keine Mengen sind).

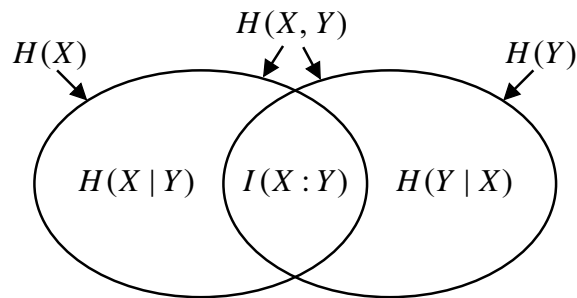


Abbildung 4.2: Beziehungen zwischen den grundlegenden Entropiemaßen.

Im Rest des Abschnittes leiten wir einige weitere grundlegende Eigenschaften der jetzt bekannten informationstheoretischen Begriffe her, die wir in den Anwendungen brauchen.

**Folgerung 4.14 (Subadditivität der Entropie):**  $H(X, Y) \leq H(X) + H(Y)$  mit Gleichheit genau dann, wenn  $X$  und  $Y$  unabhängig sind.

**Beweis:** Folgt direkt aus der Definition der wechselseitigen Information und deren Nichtnegativität (Satz 4.13).  $\square$

Die Subadditivität der Entropie gilt offensichtlich (Induktion) auch für eine beliebige endliche Anzahl von Zufallsvariablen, d. h. für beliebige  $X_1, \dots, X_n$  ist

$$H(X_1, \dots, X_n) \leq H(X_1) + \dots + H(X_n)$$

Außerdem gelten analoge Aussagen, wenn Bedingungen hinzugefügt werden, z. B.

$$H(X_1, \dots, X_n | Y) \leq H(X_1 | Y) + \dots + H(X_n | Y)$$

Dies folgt direkt aus der vorhergehenden Gleichung und der Definition der bedingten Entropie.

**Folgerung 4.15 (Bedingungen reduzieren die Entropie):**  $H(Y | X) \leq H(Y)$ .

**Beweis:** Die Kettenregel für die Entropie,  $H(X, Y) = H(X) + H(Y | X)$ , und die Subadditivität der Entropie liefern zusammen die Behauptung.  $\square$

**Satz 4.16 (Superadditivität der wechselseitigen Information):** Seien  $X_1, \dots, X_n, Y$  Zufallsvariablen mit endlichem Wertebereich, wobei  $X_1, \dots, X_n$  voneinander unabhängig seien. Sei  $X := (X_1, \dots, X_n)$ . Dann gilt

$$I(X : Y) \geq \sum_{i=1}^n I(X_i : Y).$$

**Beweis:**

$$\begin{aligned} I(X : Y) &= H(X_1, \dots, X_n) - H(X_1, \dots, X_n | Y) \\ &= \sum_i H(X_i) - H(X_1, \dots, X_n | Y) \quad (\text{Unabhängigkeit der } X_1, \dots, X_n) \\ &\geq \sum_i H(X_i) - \sum_i H(X_i | Y) \quad (\text{Subadditivität der Entropie}) \\ &= \sum_i I(X_i : Y). \end{aligned}$$

□

Die Voraussetzung, dass  $X_1, \dots, X_n$  unabhängig sind, ist im obigen Satz tatsächlich notwendig (wie man sich anhand von Beispielen klar machen kann), im Allgemeinen ist die wechselseitige Information weder sub- noch superadditiv.

Wechselseitige Information zwischen Zufallsvariablen, wenn eine weitere bekannt ist, definieren wir auf die nahe liegende Weise analog zur bedingten Entropie.

**Definition 4.17 (Bedingte wechselseitige Information):** Seien  $X, Y, Z$  Zufallsvariablen mit endlichem Wertebereich und sei  $R$  der Wertebereich von  $Z$ . Die *wechselseitige Information zwischen  $X$  und  $Y$  gegeben  $Z$*  ist definiert als

$$I(X : Y | Z) := \sum_{z \in R} \Pr\{Z = z\} I(X : Y | Z = z).$$

Dabei benutzen wir die übliche abkürzende Schreibweise

$$I(X : Y | Z = z) := I((X | Z = z) : (Y | Z = z)).$$

**Fakt 4.18:**  $I(X : Y | Z) = H(X | Z) - H(X | Y, Z) = H(Y | Z) - H(Y | X, Z)$ .

## 4.4 Fanos Ungleichung

Wir betrachten folgendes Szenario. Wir kennen eine Zufallsvariable  $Y$ , aus der wir den Wert einer uns unbekanntem Zufallsvariable  $X$  vorhersagen wollen. Sei  $f$  die Funktion, die der von uns verwendete (deterministische) Algorithmus berechnet. Dann ist  $\varepsilon := \Pr\{f(Y) \neq X\}$  der Vorhersagefehler. Man kann intuitiv vermuten, dass der Vorhersagefehler zwangsläufig groß sein muss, wenn die verbleibende Unsicherheit über  $X$  gegeben  $Y$ , d. h.  $H(X | Y)$ , groß ist. Fanos Ungleichung präzisiert diese Intuition.

**Satz 4.19 (Fanos Ungleichung):** Seien  $X$  und  $Y$  Zufallsvariablen über den Wertebereichen  $R$  bzw.  $S$  und sei  $f: R \rightarrow S$  eine beliebige Funktion. Sei  $\varepsilon := \Pr\{f(Y) \neq X\}$ . Dann gilt  $H_2(\varepsilon) + \varepsilon \cdot \log(|R| - 1) \geq H(X | Y)$ . Speziell für boolesche  $X$  ( $|R| = 2$ ) ergibt sich  $H_2(\varepsilon) \geq H(X | Y)$ .

**Beweis:** Wir definieren die Indikatorzufallsvariable  $E := [f(Y) \neq X]$ , d.h.  $E = 1$ , falls  $f(Y) \neq X$  und  $E = 0$  sonst. Indem wir die Kettenregel für die Entropie auf beide mögliche Weisen anwenden, erhalten wir:

$$H(X, E | Y) = H(X | Y) + H(E | X, Y) = H(E | Y) + H(X | E, Y).$$

Wir untersuchen nacheinander die letzten drei Terme in der obigen Zeile. Da  $E$  eine Funktion von  $X$  und  $Y$  ist, folgt  $H(E | X, Y) = 0$  mit Satz 4.5. Weiter ist  $H(E | Y) \leq H(E)$  (Bedingungen verringern die Entropie). Per Definition gilt  $H(E) = H_2(\varepsilon)$ . Schließlich erhalten wir für den letzten Term:

$$H(X | E, Y) = H(X | E = 0, Y) \cdot \Pr\{E = 0\} + H(X | E = 1, Y) \cdot \Pr\{E = 1\}.$$

Es ist  $H(X | E = 0, Y) = 0$ , da unter der Bedingung  $E = 0$  gilt, dass  $X = f(Y)$ . Unter der Bedingung  $E = 1$  gilt  $X \in R - \{f(Y)\}$  und damit  $H(X | E = 1, Y) \leq \log(|R| - 1)$ . Zusammensetzen unserer Einzelergebnisse liefert die Behauptung.  $\square$

### Zusammenfassung

Seien  $X, X_1, \dots, X_n, Y$  Zufallsvariablen mit endlichem Wertebereich. Seien  $R_X, R_Y$  die Wertebereiche von  $X$  bzw.  $Y$ .

- (1)  $H(X) = - \sum_{x \in R_X} \Pr\{X = x\} \log \Pr\{X = x\}$ .
- (2)  $0 \leq H(X) \leq \log |R_X|$ , Gleichheit genau dann, wenn  $X$  uniforme Verteilung auf  $R_X$ .
- (3)  $H(f(X) | X) = 0$ .
- (4)  $H(X | Y) = \sum_{y \in R_Y} \Pr\{Y = y\} H(X | Y = y)$ .
- (5)  $H(X, Y) = H(X) + H(Y | X) = H(Y) + H(X | Y)$  (Kettenregel für die Entropie).
- (6)  $I(X : Y) = H(X) + H(Y) - H(X, Y) = H(X) - H(X | Y) = H(Y) - H(Y | X) \geq 0$ .
- (7)  $H(X, Y) \leq H(X) + H(Y)$  (Subadditivität der Entropie).
- (8)  $H(X | Y) \leq H(X)$  (Bedingungen reduzieren die Entropie).
- (9)  $X_1, \dots, X_n$  unabhängig:  $I(X_1, \dots, X_n : Y) \geq \sum_i I(X_i : Y)$   
(Superadditivität der wechselseitigen Information).
- (10)  $H_2(\Pr\{f(Y) \neq X\}) \geq H(X | Y)$  (Fanos Ungleichung für boolesche  $X$ ).

# 5 Fortgeschrittene Themen

## 5.1 Runden

Sei  $\text{rows}(f)$  die Anzahl verschiedener Zeilen in der Kommunikationsmatrix für  $f$ .

**Satz 5.1:**  $D^{A \rightarrow B}(f) = \lceil \log \text{rows}(f) \rceil + 1$ ,

**Beweis:** Die obere Schranke ist offensichtlich. Für die untere Schranke beobachten wir, dass in einem deterministischen Einrundenprotokoll für  $f$  Alice für verschiedene Zeilen in der Kommunikationsmatrix verschiedene Botschaften verschicken muss, da sonst Bob falsche Ausgaben produziert. Damit benötigt Alice insgesamt mindestens  $\text{rows}(f)$  verschiedene Botschaften. Für diese Botschaften werden  $\lceil \log \text{rows}(f) \rceil$  Bits benötigt und 1 Bit für die Ausgabe von Bob.  $\square$

Die *Indexfunktion*  $\text{IND}_n: \{0, 1\}^n \times \{1, \dots, n\} \rightarrow \{0, 1\}$  sei für  $x \in \{0, 1\}^n$ ,  $y \in \{1, \dots, n\}$  definiert durch  $\text{IND}_n(x, y) := x_y$ .

**Satz 5.2:** Für beliebige  $0 \leq \varepsilon \leq 1/2$  gilt:  $D_\varepsilon^{A \rightarrow B}(\text{IND}_n) \geq (1 - H_2(\varepsilon))n$ . Insbesondere folgt damit auch  $R^{A \rightarrow B}(\text{IND}_n) = \Omega(n)$ .

Wir führen den Beweis mit Hilfe von Informationstheorie.

**Lemma 5.3:** Betrachte ein deterministisches Einrundenprotokoll mit Komplexität  $c$ . Für eine Eingabe  $x$  sei  $a(x)$  die von Alice verschickte Botschaft. Sei  $X$  eine Zufallsvariable, die eine zufällige Eingabe für Alice beschreibt. Dann gilt  $c \geq H(a(X))$ .

**Beweis:** Sei  $m$  die Anzahl verschiedener Botschaften, die Alice in dem gegebenen Protokoll verschicken kann. Es ist dann  $c \geq \log m$ . Andererseits liefert Folgerung 4.9, dass  $\log m \geq H(a(X))$ .  $\square$

**Beweis von Satz 5.2:** Wir betrachten als Verteilung für die approximativen Protokolle die Produktverteilung aus den Gleichverteilungen über Alices' und Bobs Eingaben. Seien  $X = (X_1, \dots, X_n) \in \{0, 1\}^n$  und  $Y \in \{1, \dots, n\}$  zufällig gleichverteilt. Für  $x \in \{0, 1\}^n$  sei  $a(x)$  die Botschaft, die Alice für die Eingabe  $x$  an Bob schickt. Die approximative Berechnung der Ausgabe durch Bob ist genau die Situation, die Fanos Ungleichung beschreibt: Er muss aus dem Paar aus Zufallsvariablen  $(a(X), Y)$  die unbekanntene Zufallsvariable  $\text{IND}(X, Y)$  vorherhersagen. Sein Vorhersagefehler  $\varepsilon$  ist gerade die Fehlerwahrscheinlichkeit des Protokolls. Fanos Ungleichung liefert also:

$$H_2(\varepsilon) \geq H(\text{IND}(X, Y) | a(X), Y).$$

Wir zeigen zunächst, dass

$$H(\text{IND}(X, Y) | a(X), Y) = \sum_{y \in \{1, \dots, n\}} \Pr\{Y = y\} H(\text{IND}(X, y) | a(X)). \quad (*)$$

Einerseits gilt (Fakt 4.18):

$$I(\text{IND}(X, Y) : a(X) | Y) = H(\text{IND}(X, Y) | Y) - H(\text{IND}(X, Y) | a(X), Y),$$

andererseits haben wir aufgrund der Definition der bedingten wechselseitigen Information

$$\begin{aligned} I(\text{IND}(X, Y) : a(X) | Y) &= \sum_y \Pr\{Y = y\} I(\text{IND}(X, Y) : a(X) | Y = y) \\ &= \sum_y \Pr\{Y = y\} I(\text{IND}(X, y) : a(X)) \\ &= \sum_y \Pr\{Y = y\} (H(\text{IND}(X, y)) - H(\text{IND}(X, y) | a(X))) \\ &= H(\text{IND}(X, Y) | Y) - \sum_y \Pr\{Y = y\} H(\text{IND}(X, y) | a(X)). \end{aligned}$$

Zusammensetzen der beiden Ergebnisse liefert (\*). Da  $\Pr\{Y = y\} = 1/n$  für alle  $y$  ist, erhalten wir mit Hilfe von (\*)

$$H_2(\varepsilon) \geq \frac{1}{n} \sum_y H(\text{IND}(X, y) | a(X)) \Leftrightarrow nH_2(\varepsilon) \geq \sum_y H(\text{IND}(X, y) | a(X)).$$

Wir schätzen die rechte Seite der rechten Ungleichung weiter ab:

$$\sum_y H(\text{IND}(X, y) | a(X)) \geq H((X_1, \dots, X_n) | a(X)) \geq H(X) - H(a(X)).$$

Die erste Ungleichung folgt aus der Subadditivität der Entropie und der Definition  $\text{IND}(x, y) = x_y$ . Die zweite Ungleichung erhalten wir durch Anwendung der Kettenregel in beiden möglichen Formen. Nun ist  $H(X) = n$ , wodurch wir insgesamt

$$H(a(X)) \geq n(1 - H_2(\varepsilon))$$

gezeigt haben. Die behauptete untere Schranke folgt damit aus Lemma 5.3.  $\square$

Im Folgenden präsentieren wir eine ausführliche Beschreibung des Rundenhierarchieergebnisses aus dem Buch von Kushilevitz und Nisan (Abschnitt 4.2 dort)

**Definition 5.4:** Sei  $D$  eine Verteilung auf  $X$  und  $S \subseteq X^m$ . Nenne  $i \in \{1, \dots, m\}$  *frei in  $S$  bez.  $D$* , falls es  $x_1, \dots, x_{i-1} \in X$  und eine Menge  $G \subseteq X$  gibt, sodass:

- (1)  $D(G) \geq 0,9$ .
- (2) Für jedes  $g \in G$  gibt es  $x_{i+1}, \dots, x_m \in X$ , sodass  $(x_1, \dots, x_{i-1}, g, x_{i+1}, \dots, x_m) \in S$ .

**Lemma 5.5:** Sei  $D$  eine Verteilung auf  $X$  und sei  $S \subseteq X^m$  so, dass kein  $i \in \{1, \dots, m\}$  frei in  $S$  ist bez.  $D$ . Dann gilt  $D^m(S) < 0,9^m$ .

Sei  $f: X \times Y \rightarrow \{0, 1\}$  eine beliebige Funktion. Definiere  $f^{*m}: X^m \times Y \times \{1, \dots, m\} \rightarrow \{0, 1\}$  für  $\vec{x} = (x_1, \dots, x_m) \in X^m$ ,  $y \in Y$  und  $i \in \{1, \dots, m\}$  durch  $f^{*m}(\vec{x}, y, i) := f(x_i, y)$ .

**Satz 5.6 (Rundenelimination):**

$$R^k(f^{*m}) \geq \min\left\{\frac{m}{100 \log m}, \frac{R^{k-1,B}(f)}{10 \log m}\right\}.$$

**Beweis:** Sei  $P$  ein randomisiertes  $k$ -Runden-Protokoll für  $f^{*m}$  mit Komplexität  $R^k(f^{*m})$  und Fehler höchstens  $1/4$ . Wir wenden Probability-Amplification mit  $10 \log m$  Wiederholungen an, um den Fehler auf höchstens  $1/(40m)$  zu senken (geht mit passender Chernoff-Schranke, hier ohne Rechnungsdetails). Sei  $P'$  das resultierende Protokoll mit Komplexität  $c \leq 10 \log m \cdot R^k(f^{*m})$ . O. B. d. A. ist  $c \leq m/10$ , ansonsten ( $c > m/10$ ) sind wir fertig, da dann  $R^k(f) > m/(100 \log m)$  folgt.

Sei  $D_f$  eine beliebige Verteilung auf den Eingaben von  $f$ , also  $X \times Y$ . Wir konstruieren zu dieser Verteilung eine neue Verteilung  $D^*$  auf den Eingaben von  $f^{*m}$ , also  $X^m \times Y \times \{1, \dots, m\}$  wie folgt:

- Wähle  $(x_1, y_1), \dots, (x_m, y_m) \in X \times Y$  zufällig gemäß  $D_f$ .
- Wähle  $i \in \{1, \dots, m\}$  zufällig gleichverteilt.
- Setze  $\vec{x} := (x_1, \dots, x_m)$  und  $y := y_i$ .

Die Eingabe für  $f^{*m}$  gemäß  $D^*$  ist dann  $(\vec{x}, y, i)$ .

Gemäß dem ersten Teil von Yaos Minimax-Prinzip erhalten wir aus  $P'$  ein approximierendes  $k$ -Runden-Protokoll für  $f^{*m}$  mit Fehler höchstens  $1/40m$  bez.  $D^*$  und höchstens genauso hoher Komplexität, also höchstens Komplexität  $c$ .

Im Folgenden sei  $D$  die Randverteilung auf  $X$ , die die Verteilung  $D_f$  induziert, also  $D(x) := \sum_{y \in Y} D_f(x, y)$  für  $x \in X$ . Wir beobachten, dass für eine zufällige Eingabe  $(\vec{X}, Y, I) \in X^m \times Y \times \{1, \dots, m\}$  gemäß der Verteilung  $D^*$  die Komponente  $\vec{X}$  gerade gemäß  $D^m$  verteilt ist.

*Behauptung.* Es gibt eine Menge  $S \subseteq X^m$  mit:

- (1) Für alle  $\vec{x} \in S$  sendet Alice dieselbe Botschaft  $a$  in  $P^*$ .
- (2)  $D^m(S) \geq 0,9^m$ .
- (3) Für alle  $\vec{x} \in S$  gilt:  $\Pr\{P^*(\vec{X}, Y, I) \neq f^{*m}(\vec{X}, Y, I) \mid \vec{X} = \vec{x}\} \leq 1/20m$ , wobei  $(\vec{X}, Y, I) \in X^m \times Y \times \{1, \dots, m\}$  eine zufällige Eingabe gemäß  $D^*$  bezeichne.

**Beweis der Behauptung:**

- Sei  $T$  die Menge aller  $\vec{x} \in X^m$ , sodass

$$\Pr\{P^*(\vec{X}, Y, I) \neq f^{*m}(\vec{X}, Y, I) \mid \vec{X} = \vec{x}\} \leq 1/20m.$$

Es gilt (Fehlerschranke von  $P^*$  und Satz von der totalen Wahrscheinlichkeit):

$$\begin{aligned} \frac{1}{40m} &\geq \Pr\{P^*(\vec{X}, Y, I) \neq f^{*m}(\vec{X}, Y, I)\} \\ &= \sum_{\vec{x} \in X^m} \Pr\{\vec{X} = \vec{x}\} \cdot \Pr\{P^*(\vec{X}, Y, I) \neq f^{*m}(\vec{X}, Y, I) \mid \vec{X} = \vec{x}\}. \end{aligned}$$

Aus der Markoffungleichung folgt damit, dass für mindestens die Hälfte aller  $\vec{x} \in X^m$  gemäß der durch  $D^*$  induzierten Verteilung  $D^m$  die Fehlerwahrscheinlichkeit höchstens doppelt so hoch ist wie die Schranke für den Durchschnitt, also höchstens  $1/20m$ . Damit gilt  $D^m(T) \geq 1/2$ .

- Alice sendet gemäß Konstruktion höchstens  $c \leq m/10$  Bits in  $P^*$ , also gibt es höchstens  $2^{m/10}$  verschiedene Botschaften. Mindestens eine davon, die wir  $a$  nennen, wird damit bei einem  $(1/2^{m/10})$ -Anteil aller Eingaben aus  $T$  gesendet. Sei  $S \subseteq T$  die Menge dieser Eingaben.
- Es gilt

$$D^m(S) \geq \frac{D^m(T)}{2^{m/10}} \geq 2^{-m/10-1} \geq 0,9^m.$$

Damit erfüllt  $S$  alle drei Eigenschaften. □

Sei  $S$  die Menge aus der obigen Zwischenbehauptung und  $i$  ein gemäß Lemma 5.5 existierender freier Index in  $S$  bez.  $D^m$ . Seien  $x_1, \dots, x_{i-1} \in X$  und  $G \subseteq X$  gemäß Definition der freien Indizes. Wir konstruieren nun aus  $P^*$  ein approximierendes  $(k-1)$ -Runden-Protokoll  $P''$  für  $f$  bez. der Verteilung  $D_f$  wie folgt.

**Protokoll  $P''$ :** Eingabe  $(x, y) \in X \times Y$ .

Die Spieler konstruieren Eingaben für  $P^*$  wie folgt:

- Alice:
  - 1. Fall,  $x \in G$ : Alice wählt  $\vec{x}(x) := (x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_m) \in S$ , wobei sich  $x_{i+1}, \dots, x_m \in X$  aus der Definition der freien Indizes ergeben.
  - 2. Fall,  $x \notin G$ : Alice wählt  $\vec{x}(x) \in X^m$  beliebig.
- Bob wählt  $(y, i)$  als Eingabe.

Die Spieler simulieren dann  $P^*$  auf der Eingabe  $(\vec{x}(x), y, i)$ . Dabei überspringen sie die erste Runde, in der sie davon ausgehen, dass Alice die Botschaft  $a$  gesendet hat und simulieren nur Runden  $2, \dots, k$ .

Es ist offensichtlich, dass  $P''$  höchstens die Komplexität von  $P^*$ , also höchstens  $c$  hat. Es bleibt nur noch die Fehlerwahrscheinlichkeit als approximierendes Protokoll für  $f$  bez.  $D_f$  abzuschätzen.

Sei  $(X, Y)$  eine zufällige Eingabe für  $f$  gemäß  $D_f$ . Sei  $(\vec{x}(X), Y, I)$  die von  $P''$  für solch eine Eingabe konstruierte, ebenfalls zufällige Eingabe für  $f^{*m}$ . Wir schätzen den Anteil der von  $P''$

bez.  $D_f$  falsch berechneten Eingaben ab:

$$\begin{aligned} \Pr\{P''(X, Y) \neq f(X, Y)\} &= \Pr\{P^*(\vec{x}(X), Y, I) \neq f^{*m}(\vec{x}(X), Y)\} \\ &\leq \Pr\{X \notin G\} + \sum_{x \in G} \Pr\{X = x\} \cdot \Pr\{P^*(\vec{x}(X), Y, I) \neq f^{*m}(\vec{x}(X), Y, I) \mid X = x\}. \end{aligned}$$

Da  $i$  frei in  $S$  bez.  $D$  ist und  $X$  gemäß  $D$  verteilt, folgt  $\Pr\{X \notin G\} \leq 0,1$ . Es bleibt die Wahrscheinlichkeit in der Summe abzuschätzen. Es ist

$$\begin{aligned} \Pr\{P^*(\vec{x}(x), Y, I) \neq f^{*m}(\vec{x}(x), Y, I) \mid X = x\} \\ &\leq m \cdot \Pr\{P^*(\vec{x}(X), Y, I) \neq f^{*m}(\vec{x}(X), Y, I) \mid X = x, I = i\} \\ &= m \cdot \Pr\{P^*(\vec{x}(X), Y, i) \neq f^{*m}(\vec{x}(X), Y, i) \mid X = x\}. \end{aligned}$$

Wir benutzen dabei, dass allgemein für Ereignisse  $A, B, C$ , wo  $B$  und  $C$  unabhängig sind, die Abschätzung

$$\begin{aligned} \Pr(A \mid B \cap C) &= \frac{\Pr(A \cap B \cap C)}{\Pr(B \cap C)} = \frac{\Pr(A \cap B \cap C)}{\Pr(B) \Pr(C)} \leq \frac{1}{\Pr(C)} \cdot \frac{\Pr(A \cap B)}{\Pr(B)} \\ &= \frac{1}{\Pr(C)} \cdot \Pr(A \mid B). \end{aligned}$$

gilt. Nun haben wir aber für  $x \in G$ , dass  $\vec{x}(x) \in S$  und damit gemäß Wahl von  $S$ :

$$\Pr\{P^*(\vec{x}(X), Y, i) \neq f^{*m}(\vec{x}(X), Y, i) \mid X = x\} \leq \frac{1}{20m},$$

also

$$\Pr\{P^*(\vec{x}(x), Y, I) \neq f^{*m}(\vec{x}(x), Y, I) \mid X = x\} \leq m \cdot \frac{1}{20m} = \frac{1}{20}.$$

Insgesamt ist damit die Fehlerwahrscheinlichkeit von  $P''$  durch  $0,1 + 1/20 = 0,15$  beschränkt.

Wir haben damit für jede Verteilung  $D_f$  auf  $X \times Y$  ein approximierendes Protokoll  $P''$  für  $f$  mit Fehler höchstens  $0,15$  bez.  $D_f$  der Komplexität höchstens  $c \leq 10 \log m \cdot R^k(f^{*m})$ . Das Protokoll hat  $k - 1$  Runden, wobei Bob beginnt. Eine Anwendung des zweiten Teils von Yao's Minimax-Prinzip liefert daraus ein randomisiertes  $(k - 1)$ -Runden-Protokoll für  $f$  mit geringfügig größerem, aber noch durch  $1/4$  beschränkten Fehler. Damit gilt in dem hier betrachteten Fall  $R^k(f^{*m}) \geq R^{k-1, B}(f)/(10 \log m)$ .  $\square$

## 5.3 Informationskomplexität

Wir haben bereits gesehen, wie wir mit Fanos Ungleichung untere Schranken für Einrundenprotokolle via Informationstheorie beweisen können. Dabei ist die Idee, dass Alices' Botschaften für eine Berechnung der betrachteten Funktion mit kleinem Fehler viel Information über ihre Eingabe verraten müssen, was hohe Kommunikationskomplexität impliziert. In diesem Abschnitt verallgemeinern wir diese Idee zu einer sehr mächtigen Technik, die auch für beliebige

Kommunikationsprotokolle ohne Rundenbeschränkung einsetzbar ist. Mit Informationskomplexität bezeichnen wir die Menge an Information, die ein Protokoll zwangsläufig über die Eingaben beider Spieler verraten muss, wenn es die betrachtete Funktion mit kleinem Fehler berechnet.

Um dies zu formalisieren, bezeichne für ein deterministisches Kommunikationsprotokoll  $P$  und dafür geeignete Eingaben  $x, y$  für die beiden Spieler mit  $P(x, y)$  das im Protokollbaum für  $x, y$  erreichte Blatt bzw. anschaulicher und äquivalent dazu die Mitschrift aller ausgetauschten Botschaften (üblicherweise ist es kein Problem, wenn wir Protokoll und Blätter mit demselben Buchstaben bezeichnen). Für randomisierte Protokolle ist  $P(x, y)$  eine Zufallsvariable. Unter der *Informationskomplexität von  $P$*  für zufällige Eingaben  $X, Y$  verstehen wir nun einfach den Wert  $I(P(X, Y) : X, Y)$ , d. h. intuitiv die Information, die die Mitschrift des Protokolls über die Eingaben preisgibt. Tatsächlich brauchen wir für die spätere Anwendung noch die Möglichkeit, die Verteilung der Eingaben durch eine zusätzliche Bedingung zu beeinflussen. Dafür verwenden wir folgende allgemeinere Version der Informationskomplexität.

**Definition 5.7:** Sei  $P$  ein randomisiertes Protokoll und sei  $Z$  eine Zufallsvariable, die eine zufällige Eingabe für  $P$  (für beide Spieler) beschreibt. Sei  $D$  eine beliebige Zufallsvariable. Dann bezeichnen wir den Wert  $I(P(Z) : Z | D)$  als die *bedingte Informationskomplexität von  $P$  für Eingabe  $Z$  unter der Bedingung  $D$* . Für  $0 \leq \varepsilon < 1/2$  ist die  $\varepsilon$ -Fehler-*Informationskomplexität von  $f$  für Eingabe  $Z$  unter der Bedingung  $D$* ,  $IC_\varepsilon(f ; Z | D)$ , definiert als das Minimum der entsprechenden Informationskomplexitäten von randomisierten Protokollen, die  $f$  mit Fehler  $\varepsilon$  berechnen.

Den Zusammenhang zur Kommunikationskomplexität liefert folgender Satz.

**Satz 5.8:**  $R_\varepsilon(f) \geq IC_\varepsilon(f ; Z | D)$ .

**Beweis:** Sei  $P$  ein randomisiertes Protokoll mit Informationskomplexität  $IC_\varepsilon(f ; Z | D)$ , das  $f$  mit Fehler  $\varepsilon$  berechnet. Sei  $m$  die Anzahl verschiedener Mitschriften von  $P$ . Dann gilt:

$$\begin{aligned} IC_\varepsilon(f ; Z | D) &= I(P(Z) : Z | D) = H(P(Z) | D) - H(P(Z) | Z, D) \\ &\leq H(P(Z) | D) \leq H(P(Z)) \leq \log m. \end{aligned}$$

Da  $\log m$  eine untere Schranke für die Kommunikationskomplexität von  $P$  ist, folgt die Behauptung.  $\square$

Wir setzen im Folgenden Informationskomplexität ein, um untere Schranken für Direkte-Summen-Probleme nachzuweisen. Eine zentrale Erkenntnis dabei ist es, dass die bedingte Informationskomplexität für das Gesamtproblem nach unten beschränkt ist durch die Summe der bedingten Informationskomplexitäten der einzelnen Teilprobleme, was als *Direkte-Summen-Eigenschaft der bedingten Informationskomplexität* bezeichnet wird. Damit brauchen wir „nur noch“ nachzuweisen, dass die Teilprobleme hohe Informationskomplexität haben, um eine gute untere Schranke für das Gesamtproblem zu erhalten. Als konkrete Anwendung zeigen wir im nächsten Abschnitt eine lineare untere Schranke für die randomisierte Kommunikationskomplexität der Disjunktion.

Für beliebige Hilfsfunktionen  $g: \{0, 1\}^n \rightarrow \{0, 1\}$  und  $h: A \times B \rightarrow \{0, 1\}$ ,  $A, B$  endliche Mengen, sei das betrachtete Direkte-Summen-Problem  $f: A^n \times B^n \rightarrow \{0, 1\}$  definiert durch

$$f((x_1, \dots, x_n), (y_1, \dots, y_n)) := g(h(x_1, y_1), \dots, h(x_n, y_n)).$$

**Beispiel:** Das Komplement der Disjunktheitsfunktion ergibt sich durch Wahl von  $g(z_1, \dots, z_n) := z_1 \vee \dots \vee z_n$  und  $h(x, y) := \text{AND}(x, y) = x \wedge y$ . Dann ist

$$\begin{aligned} g(h(x_1, y_1), \dots, h(x_n, y_n)) &= (x_1 \wedge y_1) \vee \dots \vee (x_n \wedge y_n) \\ &= \overline{\text{DISJ}_n((x_1, \dots, x_n), (y_1, \dots, y_n))}. \end{aligned}$$

Es sei  $\vec{Z} = (Z_1, \dots, Z_n)$  ein zufälliger Eingabevektor für  $f$ , wobei  $Z_i = (X_i, Y_i)$  eine Eingabe für die  $i$ -te Kopie von  $h$  sei. Dabei sollen  $Z_1, \dots, Z_n$  voneinander unabhängig sein. Für unsere Anwendung auf das Disjunktheitsproblem ist es entscheidend, dass auch solche  $Z_i$  zugelassen sind, wo die einzelnen  $X_i, Y_i$  *nicht* unabhängig sein müssen, d. h. wir haben keine Rechteckverteilung. Allerdings setzen wir voraus, dass das Hinzufügen einer Bedingung  $D_i = d$  diese Zufallsvariablen unabhängig macht, d. h. für beliebige  $d$  sind  $(X_i | D_i = d)$  und  $(Y_i | D_i = d)$  unabhängig. Sei im Folgenden  $\vec{D} := (D_1, \dots, D_n)$ . Um einen Namen zu haben, nennen wir einen Zufallsvektor  $\vec{Z}$  mit den beschriebenen Eigenschaften *zerlegbar bezüglich  $\vec{D}$* .

**Beispiel:** Wir betrachten für die Anwendung auf das Komplement der Disjunktheitsfunktion folgende zufällige Eingabe  $Z = (X, Y) \in \{0, 1\}^2$  für AND. Es sei  $D \in \{1, 2\}$  zufällig gleichverteilt. Falls  $D = 1$ , sei  $(X, Y) = (0, U)$ , wobei  $U$  ein zufällig gleichverteiltes Bit ist. Falls  $D = 2$ , sei umgekehrt  $(X, Y) = (U, 0)$ . Formal:  $\Pr\{X = 0 | D = 1\} = \Pr\{Y = 0 | D = 2\} = 1$  und  $\Pr\{Y = 0 | D = 1\} = \Pr\{X = 0 | D = 2\} = 1/2$ . Die beiden Komponenten von  $Z$  sind trivialerweise unabhängig unter den Bedingungen  $D = 1$  bzw.  $D = 2$ .

Weiterhin nehmen wir an, dass die zufälligen Eingabevektoren  $\vec{Z}$  *kollabierend* für das betrachtete Direkte-Summen-Problem  $f$  sind in folgendem Sinn: Sei irgendein  $z \in A \times B$  gegeben. Sei  $\vec{Z}_{-i}$  der Vektor  $\vec{Z}$  ohne die  $i$ -te Komponente und sei  $(\vec{Z}_{-i}, z)$  der Vektor, der aus  $\vec{Z}$  durch Ersetzen der  $i$ -ten Komponente durch  $z$  entsteht. Dann soll  $f(\vec{Z}_{-i}, z) = h(z)$  gelten, d. h. die Gesamtfunktion kollabiert zur Auswertung einer einzelnen Kopie der Teilfunktion  $h$  auf der vorgegebenen Eingabe  $z$ .

**Beispiel:** Wir betrachten wieder das Komplement der Disjunktheitsfunktion. Sei  $\vec{Z} = (Z_1, \dots, Z_n)$ , wobei jedes  $Z_i = (X_i, Y_i)$  eine unabhängige Kopie der oben konstruierten, zufälligen Eingabe für AND sei. Da  $Z_i \in \{(0, 0), (0, 1), (1, 0)\}$ , ist  $h(Z_i) = \text{AND}(X_i, Y_i) = 0$  für alle  $i$ . Damit kann aber durch Einsetzung von beliebigem  $z = (x, y)$  in eine der Komponenten immer noch der Funktionswert festgelegt werden, es ist  $\overline{\text{DISJ}_n}(\vec{Z}_{-i}, z) = \text{AND}(x, y)$ .

Durch die Einsetzung in einzelne Komponenten des Direkte-Summen-Problems auf die beschriebene Weise können wir aus einem Protokoll für das Gesamtproblem  $n$  Kopien von Protokollen für die Teilprobleme erzeugen, deren Informationskomplexität jeweils einen Beitrag zur Gesamtkomplexität liefert. Dies liefert folgendes Hauptergebnis dieses Abschnittes.

**Satz 5.9 (Direkte-Summen-Eigenschaft der bedingten Informationskomplexität):** Für ein Direkte-Summen-Problem  $f$  sei ein Eingabevektor  $\vec{Z} = (Z_1, \dots, Z_n)$  und ein Vektor von Zufallsvariablen  $\vec{D} = (D_1, \dots, D_n)$  vorgegeben, sodass  $\vec{Z}$  zerlegbar bezüglich  $\vec{D}$  und außerdem kollabierend für  $f$  ist. Dann gilt  $\text{IC}_\varepsilon(f; \vec{Z} | \vec{D}) \geq \sum_{i=1}^n \text{IC}_\varepsilon(h; Z_i | D_i)$ .

**Beweis:** Sei  $P$  ein randomisiertes Protokoll für  $f$  mit Fehler  $\varepsilon$  und mit

$$I(P(\vec{Z}) : \vec{Z} | \vec{D}) = \text{IC}_\varepsilon(f; \vec{Z} | \vec{D}).$$

Da  $Z_1, \dots, Z_n$  gemäß Voraussetzung unabhängig voneinander sind, liefert die Superadditivität der wechselseitigen Information:

$$I(P(\vec{Z}) : \vec{Z} | \vec{D}) \geq \sum_{i=1}^n I(P(\vec{Z}) : Z_i | \vec{D}).$$

Weiterhin gilt

$$I(P(\vec{Z}) : Z_i | \vec{D}) = \sum_d \Pr\{\vec{D}_{-i} = d\} I(P(\vec{Z}) : Z_i | \vec{D}_{-i} = d, D_i),$$

wobei sich die Summation über alle möglichen Werte von  $n - 1$  Komponenten des  $\vec{D}$ -Vektors erstreckt. Es reicht damit aus, wenn wir zeigen, dass für alle  $i$  und alle  $d$  der Wert  $I(P(\vec{Z}) : Z_i | \vec{D}_{-i} = d, D_i)$  mit der Informationskomplexität eines randomisierten Protokolls mit Fehler  $\varepsilon$  für eine Kopie der  $h$ -Funktion übereinstimmt.

Wir konstruieren das gewünschte Protokoll  $P'$  wie folgt. Sei  $z = (x, y)$  eine vorgegebene Eingabe für  $P'$ . Alice und Bob generieren zunächst jeweils unabhängig voneinander mit Hilfe ihrer privaten Zufallsbits für alle  $j \neq i$  ihre Hälfte der zufälligen Eingabe  $Z_j$  unter der Bedingung  $D_j = d_j$ . Dies ist möglich, da unter dieser Bedingung nach Voraussetzung beide Hälften unabhängig voneinander sind. Dies liefert einen zufälligen Vektor  $\vec{Z}'_{-i}$ . Die Spieler simulieren nun das ursprüngliche Protokoll  $P$  auf der Eingabe  $(\vec{Z}'_{-i}, z)$ .

Das so erhaltene Protokoll berechnet  $h$  mit Fehler  $\varepsilon$ , da seine Ausgabe aufgrund der Kollaps-Eigenschaft der betrachteten Zufallseingaben mit  $h$  genau dann übereinstimmt, wenn das für  $P$  gilt. Nun wenden wir  $P'$  auf  $Z_i$  anstelle von  $z$  an und betrachten die Informationskomplexität von  $P'$  für die Eingabe  $Z_i$  unter der Bedingung  $D_i = d'$  für irgendein  $d'$ . Da  $\vec{Z}'_{-i}$  genauso verteilt ist wie  $\vec{Z}_{-i}$  unter der Bedingung  $\vec{D}_{-i} = d$ , gilt aufgrund der Definition von  $P'$ :

$$\begin{aligned} I(P'(Z_i) : Z_i | D_i = d') &= I(P(\vec{Z}'_{-i}, Z_i) : Z_i | D_i = d') \\ &= I(P(\vec{Z}) : Z_i | \vec{D}_{-i} = d, D_i = d'). \end{aligned}$$

Indem wir auf beiden Seiten Durchschnitte über alle Wahlen von  $d'$  bilden, erhalten wir

$$\begin{aligned} I(P'(Z_i) : Z_i | D_i) &= \sum_{d'} \Pr\{D_i = d'\} I(P'(Z_i) : Z_i | D_i = d') \\ &= \sum_{d'} \Pr\{D_i = d'\} I(P(\vec{Z}) : Z_i | \vec{D}_{-i} = d, D_i = d') \\ &= I(P(\vec{Z}) : Z_i | \vec{D}_{-i} = d, D_i). \end{aligned}$$

Damit haben wir  $I(P(\vec{Z}) : Z_i | \vec{D}_{-i} = d, D_i)$  wie gewünscht als Informationskomplexität des Protokolls  $P'$  für  $h$  ausgedrückt und insgesamt den Satz bewiesen.  $\square$

## 5.4 Untere Schranke für die Disjunktheitsfunktion

Wie schon im letzten Abschnitt ist es hier günstiger, das Komplement  $\overline{\text{DISJ}}$  der Disjunktionheitsfunktion zu betrachten. Mit den Vorbereitungen reicht es nun zu zeigen, dass die Informationskomplexität der AND-Funktion auf nur zwei Bits durch eine positive Konstante (!) nach unten beschränkt ist.

Seien  $Z = (X, Y)$  und  $D$  Zufallsvariablen, die wie im letzten Abschnitt beschrieben eine zufällige Eingabe für AND liefern, d. h.  $\Pr\{X = 0 | D = 1\} = \Pr\{Y = 0 | D = 2\} = 1$  und  $\Pr\{Y = 0 | D = 1\} = \Pr\{X = 0 | D = 2\} = 1/2$ . Für diese Eingaben gilt:

**Lemma 5.10:**  $\text{IC}_\varepsilon(\text{AND} ; Z | D) \geq (1/4)(1 - 2\sqrt{\varepsilon})$ .

Damit erhalten wir mit Hilfe der Direkten-Summen-Eigenschaft der bedingten Informationskomplexität bereits unser Hauptergebnis:

**Satz 5.11:**  $R_{1/16}(\overline{\text{DISJ}}_n) \geq n/8$  und damit  $R(\text{DISJ}_n) = \Omega(n)$ .

**Beweis:** Wir betrachten randomisierte Protokolle für  $\overline{\text{DISJ}}_n$  mit Fehler höchstens  $\varepsilon := 1/16$ . Den Eingabevektor  $\vec{Z} = (Z_1, \dots, Z_n) \in \{0, 1\}^{2n}$  und den Vektor aus Zufallsvariablen  $\vec{D} = (D_1, \dots, D_n) \in \{1, 2\}^n$  konstruieren wir so, dass die  $Z_i, D_i$  unabhängige Kopien der oben beschriebenen zufälligen Eingabe für AND sind. Wie wir uns bereits überlegt haben, ist dann  $\vec{Z}$  zerlegbar bezüglich  $\vec{D}$  und kollabierend für  $\overline{\text{DISJ}}_n$ . Damit liefern Satz 5.8, die Direkte-Summen-Eigenschaft der bedingten Informationskomplexität (Satz 5.9) sowie das obige Lemma 5.10 die Behauptung:

$$\begin{aligned} R_\varepsilon(\overline{\text{DISJ}}_n) &\geq \text{IC}_\varepsilon(\overline{\text{DISJ}}_n ; \vec{Z} | \vec{D}) \geq n \cdot \text{IC}_\varepsilon(\text{AND} ; Z | D) \\ &\geq n \cdot (1/4)(1 - 2\sqrt{1/16}) = n/8. \end{aligned}$$

$\square$

Es bleibt Lemma 5.10 zu zeigen. Wir bereiten den Beweis durch eine Definition und weitere Lemmas vor. Wir benötigen die folgenden gängigen Abstandsmaße für Wahrscheinlichkeitsverteilungen. Im Unterschied zur relativen Entropie handelt es sich bei beiden um Metriken (ohne Beweis).

**Definition 5.12:** Seien  $p, q : \Omega \rightarrow [0, 1]$  Wahrscheinlichkeitsverteilungen auf dem endlichen Universum  $\Omega$ . Definiere den

- (1) *Totalvariationsabstand* von  $p$  und  $q$ ,  $V(p, q)$ , durch  $V(p, q) := \frac{1}{2} \sum_{x \in \Omega} |p(x) - q(x)|$ ;
- (2) den *Hellingerabstand* von  $p$  und  $q$ ,  $h(p, q)$ , durch  $h(p, q) := \left(1 - \sum_{x \in \Omega} \sqrt{p(x)q(x)}\right)^{1/2}$ .

In diesen Definitionen erlauben wir auch Zufallsvariablen anstelle von Wahrscheinlichkeitsverteilungen mit der nahe liegenden Interpretation. Der Totalvariationsabstand ist bis auf den Vorfaktor  $1/2$  nichts anderes als die  $L_1$ -Metrik für Vektoren im  $\mathbb{R}^n$  mit  $n = |\Omega|$ . Die beiden Abstandsmaße stehen (unter anderem) durch folgende Abschätzung, die wir hier ohne Beweis benutzen, zueinander in Beziehung.

**Lemma 5.13:** *Für beliebige Wahrscheinlichkeitsverteilungen  $p$  und  $q$  auf demselben endlichen Universum gilt  $h^2(p, q) \geq 1 - (1 - V^2(p, q))^{1/2}$ .*

Wir benutzen den Totalvariationsabstand, um Unterschiede zwischen den Verteilungen der Mitschriften eines randomisierten Protokolls auf Eingaben zu messen, die verschiedene Werte der berechneten Funktion liefern.

**Lemma 5.14:** *Sei  $P$  ein randomisiertes Protokoll, das die boolesche Funktion  $f$  mit Fehler  $\varepsilon$  berechnet und seien  $z$  und  $z'$  Eingaben für  $f$  mit  $f(z) = 0$  und  $f(z') = 1$ . Dann gilt  $V(P(z), P(z')) \geq 1 - 2\varepsilon$ .*

**Beweis:** Für  $c \in \{0, 1\}$  sei  $T_c$  die Menge der Mitschriften von  $P$ , für die das Protokoll  $c$  ausgibt (bzw. die Menge der  $c$ -Rechtecke). Sei zur Abkürzung  $p_z(t) := \Pr\{P(z) = t\}$ . Dann gilt

$$\begin{aligned} V(P(z), P(z')) &= \frac{1}{2} \left( \sum_{t \in T_0} |p_z(t) - p_{z'}(t)| + \sum_{t \in T_1} |p_z(t) - p_{z'}(t)| \right) \\ &\stackrel{\Delta\text{-Ungl.}}{\geq} \frac{1}{2} \left( \left| \underbrace{\sum_{t \in T_0} p_z(t)}_{\geq 1-\varepsilon} - \underbrace{\sum_{t \in T_0} p_{z'}(t)}_{\leq \varepsilon} \right| + \left| \underbrace{\sum_{t \in T_1} p_z(t)}_{\leq \varepsilon} - \underbrace{\sum_{t \in T_1} p_{z'}(t)}_{\geq 1-\varepsilon} \right| \right) \\ &\geq \frac{1}{2} \cdot 2(1 - 2\varepsilon) = 1 - 2\varepsilon. \end{aligned}$$

□

Der Hellingerabstand andererseits ist für uns wichtig, weil er es durch seine multiplikative Form erlaubt, Teile eines Rechtecks in einem randomisierten Protokoll zu vertauschen, ohne das sich der Abstand ändert. Das ist der Inhalt des folgenden Lemmas.

**Lemma 5.15 (Cut-and-Paste-Lemma):** *Sei  $P$  ein randomisiertes Protokoll. Dann gilt für alle Eingaben  $(x, y)$  und  $(x', y')$ :  $h(P(x, y), P(x', y')) = h(P(x, y'), P(x', y))$ .*

**Beweis:** Wir überlegen uns zunächst, dass sich in randomisierten Protokollen die Wahrscheinlichkeitsverteilungen über Mitschriften bzw. Rechtecke in folgender Weise schreiben lassen.

*Behauptung.* *Es gibt Funktionen  $q_A, q_B$  auf den Eingaben von Alice und den Mitschriften von  $P$  bzw. den Eingaben von Bob und den Mitschriften, sodass für alle Eingaben  $(x, y)$  und alle Mitschriften  $t$  von  $P$  gilt:  $\Pr\{P(x, y) = t\} = q_A(x, t) \cdot q_B(y, t)$ .*

Achtung: Auch wenn es so aussieht – die Behauptung bedeutet nicht, dass  $\Pr\{P(x, y) = t\}$  eine Rechteckverteilung ist (Produkt unabhängiger Verteilungen für Alice bzw. Bob), da  $q_A, q_B$  im Allgemeinen keine Wahrscheinlichkeitsverteilungen sein werden!

**Beweis der Behauptung:** Wir betrachten die deterministische Version des Protokolls  $P$ , bei der die Gesamteingabe von Alice aus  $x$  sowie ihrem Zufallsbitstring  $z_A$  besteht, analog hat Bob  $y$  und einen Zufallsbitstring  $z_B$ . O. B. d. A. benutzen Alice und Bob die festen Anzahlen  $r_A$  bzw.  $r_B$  von Zufallsbits. Sei  $R_t = R_{t,A} \times R_{t,B}$  das zu  $t$  korrespondierende Rechteck (dabei enthält  $R_{t,A}$  Paare der Form  $(x, z_A)$  und  $R_{t,B}$  Paare  $(y, z_B)$ ).

Dann gilt:

$$\begin{aligned} \Pr\{P(x, y) = t\} &= \sum_{z_A, z_B} 2^{-r_A - r_B} [(x, z_A, y, z_B) \in R_t] \\ &= \sum_{z_A, z_B} 2^{-r_A - r_B} [(x, z_A) \in R_{t,A}] [(y, z_B) \in R_{t,B}] \\ &= \left( \sum_{z_A} 2^{-r_A} [(x, z_A) \in R_{t,A}] \right) \left( \sum_{z_B} 2^{-r_B} [(y, z_B) \in R_{t,B}] \right). \end{aligned}$$

Die beiden Faktoren in der letzten Zeile liefern die Definitionen für die gewünschten Funktionen  $q_A(x, t)$  bzw.  $q_B(y, t)$ .  $\square$

Wir können dies nun direkt anwenden, um die Hellingerabstände im Lemma ineinander umzurechnen:

$$\begin{aligned} 1 - h^2(P(x, y), P(x', y')) &= \sum_t (\Pr\{P(x, y) = t\} \cdot \Pr\{P(x', y') = t\})^{1/2} \\ &= \sum_t (q_A(x, t)q_B(y, t) \cdot q_A(x', t)q_B(y', t))^{1/2} = \sum_t (q_A(x, t)q_B(y', t) \cdot q_A(x', t)q_B(y, t))^{1/2} \\ &= \sum_t (\Pr\{P(x, y') = t\} \cdot \Pr\{P(x', y) = t\})^{1/2} = 1 - h^2(P(x, y'), P(x', y)). \end{aligned}$$

$\square$

Das folgende letzte und zentrale Lemma für den Beweis der informationstheoretischen Schranke für AND erlaubt es, wechselseitige Information mit Hilfe des Hellingerabstandes nach unten abzuschätzen. Wir haben dabei folgendes Szenario. Gegeben sind zwei Zufallsvariablen  $R_0, R_1$  und wir wählen eine mit einem zufällig gleichverteilten Bit  $U$  aus, was durch die Zufallsvariable  $R_U$  beschrieben wird. Intuitiv gilt: Je verschiedener  $R_0$  und  $R_1$  sind, desto leichter sollte es sein, aus gegebenem  $R_U$  das Bit  $U$  zu rekonstruieren, d. h. desto mehr Information über  $U$  liefert  $R_U$ . Wir ersparen uns hier den erstaunlich technischen Beweis der Formalisierung dieser Intuition, der mehrere weitere Abstandsmaße für Wahrscheinlichkeitsverteilungen als Zwischenstationen verwendet.

**Lemma 5.16:** Seien  $R_0, R_1$  Zufallsvariablen und sei  $U$  ein zufällig gleichverteiltes Bit. Dann gilt  $I(R_U : U) \geq h^2(R_0, R_1)$ .

**Beweis von Lemma 5.10:** Sei  $P$  ein  $\varepsilon$ -Fehler-Protokoll für AND mit minimaler Informationskomplexität  $IC_\varepsilon(\text{AND}; Z | D)$ . Aus der Definition der bedingten wechselseitigen Information und der Definition der betrachteten Eingabeverteilung folgt zunächst

$$\begin{aligned} I(P(Z) : Z | D) &= \frac{1}{2}(I(P(Z) : Z | D = 1) + I(P(Z) : Z | D = 2)) \\ &= \frac{1}{2}(I(P(0, U) : U) + I(P(U, 0) : U)). \end{aligned}$$

Dabei sei  $U$  wieder ein zufällig gleichverteiltes Bit. Die auftauchenden Informationswerte sind nun genau von der Form, für die die Abschätzung durch Hellingerabstände aus Lemma 5.16 anwendbar ist. Damit erhalten wir:

$$\begin{aligned} I(P(Z) : Z | D) &= \frac{1}{2}(I(P(0, U) : U) + I(P(U, 0) : U)) \\ &\geq \frac{1}{2}(h^2(P(0, 0), P(0, 1)) + h^2(P(0, 0), P(1, 0))). \end{aligned}$$

Dabei sind die auftauchenden  $P(0, 0)$  usw. als Verteilungen über Mitschriften zu lesen. An dieser Stelle scheint nicht viel gewonnen zu sein, da auf den ersten Blick nichts dafür spricht, dass sich die Verteilungen  $P(0, 0)$  und  $P(0, 1)$  bzw.  $P(0, 0)$  und  $P(1, 0)$  nennenswert unterscheiden, da für die betreffenden Eingaben korrekte Protokolle immer die Ausgabe 0 liefern. Allerdings stellt die Rechteckstruktur der Protokolle eine starke Einschränkung dar, die wir noch nicht ausgenutzt haben.

Dazu halten wir zunächst folgende Abschätzung fest, die sich mit Hilfe der Cauchy-Schwarz-Ungleichung für beliebige  $x, y \in \mathbb{R}$  ergibt:

$$x + y \leq \sqrt{2}\sqrt{x^2 + y^2} \Leftrightarrow \frac{1}{2}(x^2 + y^2) \geq \frac{1}{4}(x + y)^2.$$

Zusammen mit der Tatsache, dass für den Hellingerabstand als Metrik die Dreiecksungleichung gilt sowie dem Cut-and-Paste-Lemma erhalten wir nun:

$$\begin{aligned} I(P(Z) : Z | D) &\geq \frac{1}{2}(h^2(P(0, 0), P(0, 1)) + h^2(P(0, 0), P(1, 0))) \\ &\geq \frac{1}{4}(h(P(0, 0), P(0, 1)) + h(P(0, 0), P(1, 0)))^2 \quad (\text{Cauchy-Schwarz}) \\ &\geq \frac{1}{4}h^2(P(0, 1), P(1, 0)) \quad (\Delta\text{-Ungleichung}) \\ &= \frac{1}{4}h^2(P(0, 0), P(1, 1)). \quad (\text{Cut-and-Paste}) \end{aligned}$$

Dies sind nun zwei Verteilungen, von denen wir sehr wohl annehmen können, dass sie für Protokolle, die AND mit kleinem Fehler berechnen, hinreichend verschieden sind: Lemma 5.14 liefert für Protokolle mit Fehler  $\varepsilon$ , dass

$$V(P(0, 0), P(1, 1)) \geq 1 - 2\varepsilon.$$

Nun brauchen wir nur noch mit Hilfe von Lemma 5.13 den Hellingerabstand durch den Totalvariationsabstand abzuschätzen:

$$\begin{aligned}
\frac{1}{4}h^2(P(0, 0), P(1, 1)) &\geq \frac{1}{4}[1 - (1 - V^2(P(0, 0), P(1, 1)))^{1/2}] \\
&\geq \frac{1}{4}[1 - (1 - (1 - 2\varepsilon)^2)^{1/2}] = \frac{1}{4}[1 - (4\varepsilon(1 - \varepsilon))^{1/2}] \\
&\geq \frac{1}{4}(1 - 2\sqrt{\varepsilon}).
\end{aligned}$$

Damit haben wir insgesamt die Behauptung gezeigt.  $\square$

## 5.5 Rechteckreduktionen

Wenn wir bekannte untere Schranken auf neue Probleme übertragen wollen, können wir dies formal elegant mit einem für 2-Parteien-Kommunikationsprotokolle passenden Reduktionskonzept beschreiben, das wir hier vorstellen.

**Definition:** Seien  $f: X \times Y \rightarrow Z$  und  $g: X' \times Y' \rightarrow Z$  gegeben. Dann heißt  $f$  auf  $g$  *rechteckreduzierbar*,  $f \leq_{\text{rect}} g$ , falls es Abbildungen  $h_A: X \rightarrow X'$  und  $h_B: Y \rightarrow Y'$  gibt, sodass für alle Eingaben  $(x, y) \in X \times Y$  gilt:  $f(x, y) = g(h_A(x), h_B(y))$ . Das Paar der Abbildungen  $(h_A, h_B)$  nennen wir auch *Rechteckreduktion*.

**Lemma:** Es gelte  $f \leq_{\text{rect}} g$ . Dann folgt  $D(f) \leq D(g)$ . Analoge Aussagen gelten für alle anderen hier behandelten Maße für 2-Parteien-Kommunikationskomplexität.

**Beweis:** Wir erhalten aus einem Protokoll für  $g$  offensichtlich ein Protokoll für  $f$  mit höchstens derselben Komplexität wie folgt: Alice und Bob berechnen privat jeweils  $h_A(a)$  bzw.  $h_B(b)$  und simulieren auf den Ergebnissen dann das gegebene Protokoll für  $g$ .  $\square$

Ein Spezialfall von Rechteckreduktionen sind offenbar Konstantsetzungen von Variablen in der Eingabe für das Zielproblem  $g$ . In diesem Fall ist  $f$  einfach eine Subfunktion von  $g$ .



## 7 Variable Eingabepartitionen

In den bisherigen Kapiteln haben wir immer mit einer festen Aufteilung der Eingabe zwischen den Spielern gearbeitet, die durch die Definition der betrachteten Funktion festgelegt war. In Anwendungen für andere Berechnungsmodelle müssen wir aber oft damit klarkommen, dass in beliebiger Reihenfolge auf die Eingabebits zugegriffen wird, sodass in den zugehörigen Kommunikationsproblemen die Aufteilung vorab nicht mehr fest wählbar ist. Wir stellen hier eine Variante unseres Standardmodells vor, die auf diese Situation zugeschnitten ist, und überlegen uns, wie man untere Schranken dafür beweisen kann.

Wir machen zunächst die Aufteilung der Eingabe in Kommunikationsprotokollen explizit. Dazu beschreiben wir Eingabebits durch boolesche Variablen.

**Definition:** Sei  $X$  eine endliche Menge, deren Elemente wir als boolesche Variablen interpretieren, und  $\Pi = (X_1, X_2)$  eine Partition von  $X$ , d. h.  $X = X_1 \cup X_2$  und  $X_1 \cap X_2 = \emptyset$ . Ein  $\Pi$ -*Protokoll* ist ein Kommunikationsprotokoll, bei dem Alice eine Belegung  $x$  der Variablen in  $X_1$  erhält und Bob eine Belegung  $y$  der Variablen in  $X_2$ , formal Abbildungen  $x: X_1 \rightarrow \{0, 1\}$  bzw.  $y: X_2 \rightarrow \{0, 1\}$ . Mit  $D^\Pi(f)$  bezeichnen wir die minimale Komplexität eines  $\Pi$ -Protokolls für  $f$ .

Hier und bei nachfolgenden Definitionen ergeben sich analog auch entsprechende Varianten von anderen Maßen für Kommunikationskomplexität, ohne dass wir das explizit erwähnen, wie z. B. für nichtdeterministische und randomisierte Protokolle.

Wir wählen nun für ein gegebenes Problem die Partition  $\Pi$  so, dass die Kommunikationskomplexität minimal wird. Es ist klar, dass es keine nichttrivialen unteren Schranken gibt, wenn wir *beliebige* Wahlen von Partitionen erlauben. Wir müssen uns also auf für die jeweiligen Anwendungen sinnvolle Teilklassen einschränken.

Wir betrachten hier folgende zwei Typen von Partitionen:

- Partitionen  $\Pi = (X_1, X_2)$  von  $X$ , bei denen Alice und Bob jeweils eine feste Anzahl von Variablen aus einer Teilmenge  $W \subseteq X$  von „wichtigen“ Variablen erhalten, d. h.  $|X_1| = k$  und  $|X_2| = |W| - k$  für ein festes  $k$ .
- Partitionen  $\Pi = (X_1, X_2)$  von  $X$ , bei denen beide Spieler höchstens  $\lceil |W|/2 \rceil$  der Variablen aus der Menge  $W$  der wichtigen Variablen erhalten. Diese nennen wir *balanciert bezüglich  $W$*  bzw. für  $W = X$  nur *balanciert*.

Die Komplexität einer Funktion von  $f$  in diesem neuen Szenario ergibt sich jeweils als das Minimum der Komplexität von  $D^\Pi(f)$  über alle erlaubten  $\Pi$ . Wir verzichten darauf, eine Notation für die entsprechenden Komplexitätsmaße einzuführen.

Es ist zu erwarten, dass der Nachweis von unteren Schranken durch die neuen Freiheiten schwieriger wird. Tatsächlich ist es leicht zu sehen, dass die meisten unserer bisherigen Beispielfunktionen für Protokolle, bei denen eine beliebige balancierte Partition der kompletten Variablenmenge gewählt werden darf, trivial berechenbar werden.

Wir erhalten allerdings auf künstliche Weise Varianten der bekannten Funktionen, die auch bei variabler Partition schwierig sind. Dies ist als *Maskentechnik* bekannt. Wir definieren z. B. die „maskierte“ Version der Equality-Funktion  $EQ_n^*$  wie folgt auf Eingaben  $x, x', y, y' \in \{0, 1\}^n$ . Es sei  $x^*$  der Teilvektor, der genau aus den Bits  $i$  in  $x$  besteht, für die  $x'_i = 1$  ist, analog  $y^*$ . (Die Vektoren  $x', y'$  haben die Funktion von Bitmasken, mit denen Teile aus  $x$  bzw.  $y$  ausgeschnitten werden.) Wir setzen  $EQ_n^*(x, x', y, y') := EQ_m(x^*, y^*)$ , falls  $|x^*| = |y^*| = m$  und  $EQ_n^*(x, x', y, y') := 0$  sonst. Seien  $X, Y$  die Variablen in den Vektoren  $x, y$ .

**Satz:** Sei  $\Pi$  eine bezüglich  $X \cup Y$  balancierte Partition. Dann gilt  $N^{1, \Pi}(EQ_n^*) \geq \lceil n/2 \rceil + 1$ .

**Beweis:** Sei  $\Pi$  eine Partition, bei der Alice und Bob jeweils genau  $n$  Variablen aus  $X \cup Y$  besitzen. Dann hat einer der beiden, o. B. d. A. Alice, eine Teilmenge  $U$  von  $\lceil n/2 \rceil$  Variablen aus  $X$  und Bob eine Teilmenge  $V$  von  $\lceil n/2 \rceil$  Variablen aus  $Y$ . Wir fixieren die Bitmasken  $x'$  und  $y'$  so, dass jeweils die  $U$ - bzw.  $V$ -Variablen ausgewählt werden. Alice und Bob müssen damit  $EQ_{\lceil n/2 \rceil}$  bezüglich der Partition  $(U, V)$  lösen, die der durch die Definition gegebenen Standardpartition entspricht.  $\square$

Wir stellen noch ein etwas natürlicheres Beispiel vor. Sei  $n = 2^m$  und  $k = m - \lfloor \log m \rfloor$ . Die Funktion  $ISA_n$  (*indirect storage access*) ist auf Variablenvektoren  $x = (x_0, \dots, x_{n-1})$  und  $y = (y_0, \dots, y_{k-1})$  definiert. Für  $i = 0, \dots, \lfloor n/m \rfloor - 1$  sei  $x(i) := (x_{im}, \dots, x_{im+m-1})$ , genannt *i-ter x-Block*. Falls  $|y|_2 = i \in \{0, \dots, \lfloor n/m \rfloor - 1\}$ , dann setzen wir  $ISA_n(x, y) := x_{|x(i)|_2}$ , sonst sei  $ISA_n(x, y) := 0$ . Seien  $X, Y$  die Mengen der Variablen in den Vektoren  $x$  bzw.  $y$ .

**Satz:** Sei  $\Pi = (X_1, X_2)$  eine Partition der Eingabevariablen von  $ISA_n$ , bei der  $|X_1 \cap X| = m - 1$  ist. Dann gilt  $R^{A \rightarrow B, \Pi}(ISA_n) = \Omega(n/\log n)$ .

**Beweis:** Wir zeigen, dass  $IND_{m-1}$  bezüglich der Standardpartition der Eingabe auf  $ISA_n$  bezüglich  $\Pi$  rechteckreduzierbar ist. Daraus ergibt sich die Behauptung aus der linearen unteren Schranke für die randomisierte 1-Runden-Kommunikationskomplexität der Indexfunktion aus Kapitel 5.

Sei eine Eingabe  $(u, v) \in \{0, 1\}^{m-1} \times \{1, \dots, m-1\}$  für  $IND_{m-1}$  gegeben. Wir konstruieren dazu eine Eingabe  $(x, y)$  für  $ISA_n$ . Da  $|X_1 \cap X| = m - 1$  und damit kleiner als die Anzahl von  $x$ -Blöcken ist, enthält  $X_2$  einen kompletten Block, sagen wir  $x(i)$ . Wir setzen die  $y$ -Variablen in der Eingabe von  $ISA_n$  so konstant, dass  $|y|_2 = i$  ist. Die  $m - 1$   $x$ -Variablen in  $X_1$  belegen wir mit dem Vektor  $u$  und die Variablen in  $x(i)$  belegen wir so, dass  $|x(i)|_2 = v$  ist. Alle restlichen Variablen werden irgendwie beliebig fixiert. Offenbar können die Belegungen  $x$  und  $y$  unabhängig voneinander vorgenommen werden und es gilt

$$ISA_n(x, y) = x_{|x(i)|_2} = u_v = IND_{m-1}(u, v).$$

Damit haben wir die gewünschte Rechteckreduktion konstruiert.  $\square$

Unser letztes Beispiel ist eine Funktion, die schwierig für balancierte Partitionen der kompletten Variablenmenge ist. Die Funktion  $1ROW\text{-}OR\text{-}1COL_n$  ist auf  $n \times n$ -Matrizen  $X = (x_{i,j})_{1 \leq i, j \leq n}$  von booleschen Variablen definiert und 1, falls es eine Zeile oder Spalte aus lauter Einsen gibt und 0 sonst.

**Satz:** Sei  $\Pi$  eine balancierte Partition der Eingabevariablen von  $1\text{ROW-OR-1COL}_n$ . Dann gilt  $N^{1,\Pi}(1\text{ROW-OR-1COL}_n) \geq n/2 + 1$  und  $R^\Pi(1\text{ROW-OR-1COL}_n) = \Omega(n)$ .

**Beweis:** Der Einfachheit halber sei  $n^2$  und damit  $n$  gerade. Unser Ziel ist es, ein passendes bekanntes Problem auf  $1\text{ROW-OR-1COL}$  zu reduzieren.

Nenne eine Zeile oder Spalte von  $X$  *geteilt*, wenn in der gegebenen Partition  $\Pi$  jeder der beiden Spieler mindestens eine Variablen dieser Zeile bzw. Spalte hat. Wir überlegen uns zunächst, dass es für jede balancierte Partition mindestens  $n/2$  geteilte Zeilen oder mindestens  $n/2$  geteilte Spalten gibt. Falls jeder der beiden Spieler eine vollständige Zeile besitzt, sind offensichtlich  $n$  Spalten geteilt und wir sind fertig. O. B. d. A. habe nur Alice vollständige Zeilen. Die Anzahl dieser Zeilen ist aber höchstens  $n/2$ , da sie insgesamt nur  $n^2/2$  Variablen hat. Damit sind mindestens  $n/2$  Zeilen geteilt. (Genauer Zählen zeigt übrigens, dass es sogar immer mindestens  $\lfloor n/\sqrt{2} \rfloor$  geteilte Zeilen bzw. Spalten gibt.)

O. B. d. A. seien die ersten  $n/2$  Zeilen von  $X$  bezüglich der gegebenen Partition geteilt. Wir konstruieren nun eine Rechteckreduktion von  $\text{DISJ}_{n/2}$  auf  $1\text{ROW-OR-1COL}_n$ . Sei  $(x, y) \in \{0, 1\}^{n/2} \times \{0, 1\}^{n/2}$  eine Eingabe für  $\text{DISJ}_{n/2}$ . Für  $i = 1, \dots, n/2$  belegen wir in Zeile  $i$  von  $X$  eine Alice-Variablen mit  $x_i$  und eine Bob-Variablen mit  $y_i$  und alle restlichen Variablen der Zeile mit 1. Die Variablen in den letzten  $n/2$  Zeilen setzen wir alle auf 0. Dann hat die so konstruierte Matrix  $X$  genau dann eine Zeile aus lauter Einsen, wenn es ein  $i$  mit  $x_i = y_i = 1$  gibt. Andererseits gibt es keine Spalten aus lauter Einsen. Damit gilt

$$1\text{ROW-OR-1COL}_n(X) = \text{DISJ}_{n/2}(x, y)$$

und wir haben die gewünschte Rechteckreduktion konstruiert. Damit können wir alle bekannten unteren Schranken für die Disjunktheitsfunktion anwenden, insbesondere die für nichtdeterministische und randomisierte Komplexität.  $\square$



# 8 Anwendungen für Branchingprogramme

## 8.1 Branchingprogramme – Das Modell

Branchingprogramme sind eine sehr viel allgemeinere Variante der OBDDs (*ordered binary decision diagrams*) aus dem Grundstudium. Während OBDDs in der Praxis als Datenstruktur für boolesche Funktionen eingesetzt werden, sind Branchingprogramme eins der wichtigsten nichtuniformen Standardmodelle der Komplexitätstheorie. Dieses Modell ist vor allem durch den Wunsch motiviert, die zur Verfügung stehenden Techniken zum Nachweis von unteren Schranken für den Speicherplatzverbrauch bzw. von Zeit-Platz-Tradeoff-Schranken zu verbessern.

**Definition:** Ein *Branchingprogramm (BP)* über der Variablenmenge  $X = \{x_1, \dots, x_n\}$  ist ein gerichteter, azyklischer Multigraph. Innere Knoten sind mit Variablen aus  $X$  markiert und haben genau zwei ausgehende Kanten, die mit 0 bzw. 1 markiert sind (und oft gestrichelt bzw. durchgezogen gezeichnet werden). Senken sind ebenfalls mit 0 oder 1 markiert. Es gibt einen ausgezeichneten Startknoten. Das BP berechnet eine auf  $X$  definierte boolesche Funktion wie folgt. Für eine Belegung der Variablen  $a = (a_1, \dots, a_n) \in \{0, 1\}^n$  verfolgen wir den eindeutig definierten *Rechenweg* vom Startknoten zu einer Senke, der an  $x_i$ -Knoten der  $a_i$ -Kante folgt. Der Wert der erreichten Senke ist der Ausgabewert des Branchingprogramms.

**Definition:** Wir betrachten folgende Parameter von BPs:

- *Größe:* Anzahl von Knoten im Graphen, für BP  $G$  mit  $|G|$  bezeichnet;
- *Länge:* maximale Länge eines Weges vom Startknoten zu einer Senke;
- *Rechenzeit:* maximale Länge eines *Rechenweges* vom Startknoten zu einer Senke.

Die *Branchingprogrammgröße* von  $f$ ,  $\text{BP}(f)$  ist die minimale Größe eines BPs für  $f$ . Analog sind *Branchingprogramm länge* von  $f$  und *Branchingprogrammrechenzeit* von  $f$  definiert.

Wie bei Schaltkreisen betrachten wir genaugenommen Folgen  $f = (f_n)$  von booleschen Funktionen,  $f_n: \{0, 1\}^n \rightarrow \{0, 1\}$  für  $n \in \mathbb{N}$ , die durch Folgen von BPs  $G = (G_n)$  dargestellt werden, wobei für  $n \in \mathbb{N}$  jeweils  $G_n$  die Funktion  $f_n$  berechnet. Üblicherweise reden wir aber in Beweisen immer über eine bestimmte, vorab fixierte Eingabelänge  $n$ , was das Modell sehr einfach handhabbar macht (nur eine endliche Menge von möglichen Algorithmen).

Offensichtlich hat jede boolesche Funktion auf  $n$  Variablen ein BP der Größe  $2^n + 1$  und der Länge sowie Rechenzeit  $n$  (benutze ein entsprechendes OBDD). Die Rechenzeit kann tatsächlich kürzer als die Länge sein, da *inkonsistente Wege* mit Kanten, die zu  $x_i = 0$  und  $x_i = 1$  gehören, vorkommen können. Solche Wege sind für keine Eingabe Rechenweg. Allerdings können wir erstaunlicherweise trotzdem ganz allgemein Rechenzeitschranken in angenehmere (weil syntaktisch direkt am Graphen ablesbare) Längenschranken umwandeln, wenn wir einen polynomiellen Blowup in der Größe in Kauf nehmen.

**Satz:** Zu jedem BP  $G$  mit Größe  $s$  und Rechenzeit  $T$  gibt es ein BP  $G'$  für dieselbe Funktion mit Größe höchstens  $(T + 1) \cdot \text{poly}(s)$ , in dem jeder Weg vom Startknoten zu einer Senke Länge exakt  $T$  hat. Insbesondere ist  $G'$  auch in Ebenen eingeteilt (d. h. Kanten verlaufen nur von einer Ebene zur jeweils nächsten).

**Beweis:** Wir konstruieren das gewünschte BP  $G'$  wie folgt. Zunächst erzeugen wir disjunkte Kopien  $G_0, \dots, G_T$  von  $G$ . Für  $i = 0, \dots, T - 1$  verbiegen wir dann in  $G_i$  für jeden Knoten die ausgehenden Kanten auf die zu den Nachfolgern korrespondierenden Knoten in  $G_{i+1}$ . Schließlich werden für  $i = 0, \dots, T - 1$  die Senken in  $G_i$  durch Dummyknoten ersetzt, die mit irgendeiner Variablen markiert sind und deren ausgehende Kanten beide zum Dummyknoten für die Senke desselben Typs in  $G_{i+1}$  (bzw. zur Senke desselben Typs in  $G_T$ ) führen. Offensichtlich ist dann  $|G'| \leq (T + 1) \cdot s$  und jeder Weg in  $G'$  hat Länge exakt  $T$ .  $\square$

Die Bedeutung von BPs ergibt sich aus der Tatsache, dass der Logarithmus der BP-Größe im Wesentlichen den Platzbedarf sequentieller Algorithmen charakterisiert und zudem die BP-Rechenzeit eine untere Schranke für deren Rechenzeit darstellt. Passend zum nichtuniformen Modell der BPs betrachten wir ein entsprechendes nichtuniformes Maschinenmodell, hier Registermaschinen (RAMs, *random access machines*, siehe GTI). Dieses erhalten wir aus gewöhnlichen Registermaschinen, indem wir ein Nur-Lese-Zusatzeingabeband ergänzen, genannt *Hilfsinformationsband*. Dieses Band darf vor der Rechnung instantan (ohne Rechenzeit) mit einer beliebigen, von der Eingabelänge abhängigen Zeichenkette gefüllt werden und es wird der Logarithmus der Länge der Hilfsinformation zum Gesamtspeicherplatz addiert (dies ist die Anzahl der zur Speicherung der Kopfposition notwendigen Bits).

Es ergibt sich folgender genauer Zusammenhang zwischen den Komplexitätsmaßen für BPs und nichtuniformen Registermaschinen.

**Satz:** Sei eine Folge  $f = (f_n)$  von booleschen Funktionen gegeben.

- (1) Sei eine Folge  $G = (G_n)$  von BPs für  $f = (f_n)$  gegeben, wobei für  $n \in \mathbb{N}$  das BP  $G_n$  Größe  $s(n) \geq n$  habe. Dann gibt es einen Algorithmus für nichtuniforme Registermaschine mit Standardbefehlssatz, der die Funktionenfolge mit Speicherplatz  $O(\log s(n))$  berechnet.
- (2) Sei ein Algorithmus für  $f = (f_n)$  auf einer nichtuniformen Registermaschine mit beliebigem Befehlssatz, Speicherplatz  $s(n) = \Omega(\log n)$  und Rechenzeit  $t(n)$  gegeben. Dann gibt es eine Folge  $G = (G_n)$  von BPs, die die Funktionenfolge berechnet und Größe  $2^{O(s(n))}$  sowie Rechenzeit höchstens  $t(n)$  besitzt.

**Beweisskizze:**  $BPs \rightarrow$  nichtuniforme Registermaschine: Als Hilfsinformation für die nichtuniforme Registermaschine wählen wir eine Codierung des BPs  $G_n$ , wobei wir für jeden Knoten Typ (innerer Knoten bzw. Senke), Variablenindex und die beiden Nachfolger notieren. Dann kann auf die nahe liegende Weise der Auswertungsalgorithmus aus der Definition der BPs auf der Registermaschine programmiert werden. Dieser benötigt  $O(\log s(n))$  Arbeitsspeicher für die Nummer des aktuell erreichten Knotens, zusätzlich einen Beitrag  $O(\log s(n))$  für das Hilfsinformationsband (die Hilfsinformation hat Länge  $s(n) \cdot O(\log s(n))$ ).

*Nichtuniforme Registermaschine*  $\rightarrow$  *BPs*: Die Registermaschine hat bei Speicherplatz  $s(n)$  höchstens  $2^{O(s(n))}$  Konfigurationen (verschiedene Kopfpositionen für Eingabe- und Hilfsinformationsband sowie verschiedene Speicherinhalte auf dem Arbeitsband). Jede Konfiguration bilden wir auf einen Knoten im BP ab, die Startkonfiguration wird der Startknoten, terminierende Konfigurationen Senken. Bei Konfigurationsübergängen wird das aktuell unter dem Eingabekopf befindliche Zeichen der Eingabe gelesen und davon abhängig eine Nachfolgekonfiguration erreicht, dies bilden wir durch passende Kanten im BP nach. Jeder Konfigurationsübergang entspricht damit einer Kante auf einem Rechenweg, sodass sich auch die Aussage über die Rechenzeit ergibt.  $\square$

Mit Hilfe dieser Aussage erhalten wir insbesondere aus unteren Schranken für BP-Größe und BP-Rechenzeit untere Schranken für Speicherplatzbedarf bzw. Rechenzeit von Algorithmen auf nichtuniformen Registermaschinen (sogar mit beliebigem Befehlssatz!). Da uniforme Registermaschinen höchstens eingeschränkter sind, gelten diese Schranken erst recht für den vertrauteren, uniformen Fall.

Ein wichtiges, offenes Problem der Komplexitätstheorie ist es, superpolynomielle untere Schranken für die BP-Größe und damit superlogarithmische Platzschranken z. B. für Probleme in P oder NP zu zeigen. Damit würde sich eine Trennung der mit logarithmischem Platz berechenbaren Probleme von denen in P bzw. NP ergeben. Das Fernziel ist weiterhin, solche superpolynomiellen oder besser sogar exponentiellen unteren Schranken für „natürliche“, „realistische“ (nicht konstruierte) Probleme zu zeigen, wie wir sie z. B. aus Vorlesungen wie „Effiziente Algorithmen“ kennen. Es ist klar, dass dies noch schwieriger ist als das erstgenannte Problem.

Was für Aussagen über die Rechenzeit können wir uns erhoffen? Zunächst beobachten wir, dass wir keine nichttrivialen absoluten Zeitschranken auf dem Umweg über BPs beweisen können, da jede boolesche Funktion auf  $n$  Variablen ein BP der Rechenzeit höchstens  $n$  hat. Interessant wird es allerdings wieder, wenn wir Speicherplatz und Rechenzeit gleichzeitig betrachten. Zum einen können wir versuchen, superlineare untere Schranken für die Rechenzeit bei sublinearem (z. B. logarithmischem) Speicherplatz zu zeigen. Zum anderen sind Zeit-Platz-Tradeoff-Schranken interessant, wie z. B.  $T(n) \cdot S(n) = \Omega(n^2)$ , wobei  $T(n)$  und  $S(n)$  Rechenzeit bzw. Speicherplatz des betrachteten Problems seien. Die hier beschriebenen Ziele sind typischerweise einfacher zu erreichen als absolute untere Schranken – und tatsächlich können solche Ergebnisse bereits heute mit den fortgeschrittensten Techniken für BPs und damit sehr allgemeine Registermaschinen-Algorithmen gezeigt werden!

Abschließend bemerken wir noch, dass sich *randomisierte BPs* (und damit auch *nichtdeterministische BPs*, randomisierte BPs mit unbeschränktem einseitigen Fehler) auf die nahe liegende Weise definieren lassen. Ein solches BP darf zusätzlich zu den bisher gewohnten Knoten, die jetzt *Entscheidungsknoten* heißen, auch *randomisierte Knoten* ohne Variablenmarkierung enthalten. An solchen Knoten wird bei einer Berechnung der Nachfolger durch einen fairen Münzwurf bestimmt. Der für eine Eingabebelegung berechnete Wert des BPs ist dann eine Zufallsvariable. Mit Hilfe dieser Zufallsvariablen lassen sich die üblichen Fehlertypen (einseitiger, beschränkter und unbeschränkter zweiseitiger Fehler usw.) definieren. Man kann dann auch

wieder zeigen, dass die solchermaßen definierten randomisierten BP-Modelle sich analog zum obigen Satz durch die zugehörigen Maschinenmodelle simulieren lassen und umgekehrt.

Im Unterschied zum deterministischen Fall kann es auch sinnvoll sein, Kreise in randomisierten BPs zuzulassen. Wir beschränken uns hier im Folgenden aber auf die azyklische Variante, da für uns sowieso BP-Modelle mit polynomiell in der Eingabelänge beschränkter Rechenzeit am wichtigsten sind und aus diesen Kreise bei nur polynomieller Vergrößerung entfernt werden können (ohne Beweis).

Da für das allgemeine BP-Modell Techniken zum Nachweis von superpolynomiellen unteren Schranken fehlen (und vermutlich auch noch länger nicht zu haben sein werden), verfolgt man in der BP-Theorie den Weg, solche Techniken für immer weniger eingeschränkte Varianten zu entwickeln. Die bekannten Techniken bedienen sich entweder direkt Maßen für Kommunikationskomplexität oder zumindest kombinatorischen Maßen für Rechtecküberdeckungen. Wir stellen im Folgenden einige Meilensteine dieser Entwicklung vor.

## 8.2 OBDDs

OBDDs sind eine der eingeschränktesten BP-Varianten, die ausführlich untersucht wurden. Wir wiederholen zunächst die Definition (siehe auch Rechnerstrukturen und DAP 2).

**Definition:** Sei  $X$  eine endliche Variablenmenge und  $\pi$  eine Permutation von  $X$ . Ein  $\pi$ -OBDD ist ein BP, bei dem die Variablen auf jedem Weg vom Startknoten zu einer Senke eine Teilfolge der Folge  $(x_{\pi(1)}, \dots, x_{\pi(n)})$  sind. Die Permutation  $\pi$  nennen wir *Variablenreihenfolge* des OBDDs. Ein OBDD ist ein BP, das  $\pi$ -OBDD für geeignetes  $\pi$  ist.

OBDDs sind für die Praxis als Datenstrukturen für boolesche Funktionen wichtig, da sie viele grundlegende Operationen auf booleschen Funktionen durch Algorithmen mit polynomieller Rechenzeit in der Darstellungsgröße unterstützen, wie z. B. Synthese und Erfüllbarkeitstest. Viele in der Praxis auftretenden Funktionen haben außerdem polynomiell große OBDDs, aber es ist natürlich klar, dass wir dies nicht für alle Funktionen erwarten können.

Wir überlegen uns hier als Aufwärmübung für die folgenden Abschnitte, wie man kann auf einfache Weise mit Hilfe von Kommunikationskomplexität untere Schranken für die OBDD-Größe zeigen kann.

**Satz (Untere-Schranken-Technik für OBDDs):** Sei  $G$  ein OBDD für  $f$  mit durch  $(x_{\pi(1)}, \dots, x_{\pi(n)})$  gegebener Variablenreihenfolge,  $\pi$  Permutation von  $\{1, \dots, n\}$ . Sei  $X_1 := \{x_{\pi(1)}, \dots, x_{\pi(\ell)}\}$  und  $X_2 := \{x_{\pi(\ell+1)}, \dots, x_{\pi(n)}\}$  mit  $1 \leq \ell \leq n - 1$ . Dann gilt für die  $l$ -Runden-Kommunikationskomplexität von  $f$  bezüglich der Eingabepartition  $\Pi = (X_1, X_2)$  zwischen Alice und Bob:

$$D^{A \rightarrow B, \Pi}(f) \leq \lceil \log |G| \rceil + 1.$$

Analoge Aussagen gelten auch für randomisierte und nichtdeterministische OBDDs und die entsprechenden Maße für  $l$ -Runden-Kommunikationskomplexität.

**Beweis:** Wir geben ein passendes Kommunikationsprotokoll für  $f$  an, in dem beide Spieler das vorgegebene OBDD benutzen. Sei  $(x, y)$  eine Eingabe für das Protokoll, wobei  $x$  und  $y$  Belegungen der Variablen in  $X_1$  bzw.  $X_2$  seien.

- Alice verfolgt den Anfang des Rechenweg für  $(x, y)$  in  $G$ , beginnend mit dem Startknoten, bis zum ersten Knoten, der nicht mehr mit einer Variablen in  $X_1$  markiert ist. Sie sendet die Nummer dieses Knotens an Bob.
- Bob verfolgt den Rest des Rechenweges, beginnend mit dem von Alice erhaltenen Knoten, bis zu einer Senke. Die Variablen an den entsprechenden Knoten gehören alle zu  $X_2$ . Seine Ausgabe ist der Wert der erreichten Senke.

Dieses Protokoll hat offensichtlich Komplexität höchstens  $\lceil \log |G| \rceil + 1$ . □

Damit wir diesen Satz für beliebige Permutationen  $\pi$  und die resultierenden Partitionen anwenden können, brauchen wir offensichtlich untere Schranken für Kommunikationskomplexität im Modell mit variablen Partitionen. Unsere entsprechenden Ergebnisse aus Kapitel 7 liefern:

**Satz:**

- (1) Jedes nichtdeterministische OBDD, das  $\text{EQ}_n^*$  berechnet, hat Größe  $2^{\Omega(n)}$ .
- (2) Jedes randomisierte OBDD mit beschränktem zweiseitigen Fehler, das  $\text{ISA}_n$  berechnet, hat Größe  $2^{\Omega(n/\log n)}$ .
- (2) Jedes nichtdeterministische OBDD und jedes randomisierte OBDD mit zweiseitigem beschränktem Fehler, das  $\text{1ROW-OR-1COL}_n$  berechnet, hat Größe  $2^{\Omega(n)}$ .

**Beweis:** Wir schreiben die Variablen als Liste gemäß der Variablenreihenfolge des vorgegebenen OBDDs auf und zerschneiden die Liste so, dass der erste Teil genau die für die beiden Beispiele jeweils erforderliche Anzahl wichtiger Variablen enthält. Für die entstehenden Partitionen sind dann die unteren Schranken aus Kapitel 7 und der vorhergehende Satz anwendbar. □

Im Hinblick auf die nachfolgenden, weiterführenden Techniken stellen wir noch eine alternative Sichtweise der Beweistechnik für OBDDs vor. Hier und im Folgenden gehen wir (o. B. d. A.) davon aus, dass die untersuchten BPs genau eine 0- bzw. 1-Senke haben. Ein *Schnitt* in einem OBDD ist eine Menge von Knoten, sodass auf jedem Startknoten-Senken-Weg genau ein Knoten des Schnitts liegt. Wir definieren den Schnitt  $C$  in einem gegebenen OBDD als die Menge der Knoten, bei der Bob gemäß dem Beweis des Satzes über die OBDD-Beweistechnik die Berechnung fortsetzt. Für einen Knoten  $v \in C$  definieren wir  $R_v$  als die Menge der Eingaben, deren Rechenweg vom Startknoten über  $v$  zur (einzig) 1-Senke verläuft. Offensichtlich handelt es sich dabei um ein 1-Rechteck, da wir die Teileingaben, die zu den Wegabschnitten „Startknoten  $\rightarrow v$ “ und „ $v \rightarrow$  1-Senke“ gehören, unabhängig voneinander wählen können und natürlich nur 1-Eingaben in  $R_v$  liegen können. Die Mengen  $R_v$  für die Knoten  $v \in C$  korrespondieren in eindeutiger Weise zu den 1-Rechtecken im oben konstruierten Kommunikationsprotokoll und bilden insgesamt eine Partition der 1-Eingaben der dargestellten Funktion. Analoges gilt für die 0-Eingaben. Wir haben damit eine rein kombinatorische Charakterisierung der OBDD-Struktur für die Untere-Schranken-Technik erreicht.

### 8.3 Read-once-Branchingprogramme

Ein *stereotypes BP* ist ein BP, bei dem gefordert wird, dass die Zugriffsreihenfolge auf die Variablen nur von der Eingabelänge, aber nicht vom Inhalt der Eingabe abhängt, analog zu stereotypen Turingmaschinen. Zusammen mit Längenbeschränkungen ist diese technisch motivierte Eigenschaft sehr hilfreich für den Nachweis von unteren Schranken. OBDDs stellen den Spezialfall von stereotypen BPs mit Länge  $n$  dar.

Wenn wir für praxisnähere Berechnungsmodelle neue Untere-Schranken-Techniken entwickeln wollen, besteht eine wesentliche Herausforderung darin, auch *nichtstereotype* Modelle behandeln zu können. Eins der einfachsten nichtstereotypen BP-Modelle sind die in diesem Abschnitt untersuchten Read-once-BPs.

**Definition:** Ein *Read-once-BP* ist ein BP, in dem auf jedem Weg vom Startknoten zu einer Senke jede Variable höchstens einmal auftaucht.

Offensichtlich sind OBDDs spezielle Read-once-BPs. Auf den ersten Blick sieht die Abänderung in der Definition auch relativ harmlos aus. Tatsächlich treten aber viele grundlegende Probleme bei der Behandlung von allgemeineren nichtstereotypen BP-Modellen mit Unteren-Schranken-Techniken bereits bei Read-once-BPs auf, während das Modell andererseits kombinatorisch noch relativ elegant handhabbar ist. Wir bemerken noch, dass Read-once-BPs wie OBDDs die Eigenschaft haben, dass jeder Weg vom Startknoten zu einer Senke tatsächlich auch Rechenweg für eine Eingabe ist, d. h. es gibt keine inkonsistenten Wege.

Wir überlegen uns im Folgenden, wie wir untere Schranken für das Modell nachweisen können. Zunächst vereinfachen wir die Struktur der Read-once-BPs etwas, wie in folgendem Lemma beschrieben.

**Lemma und Definition:** *Ein beliebiges Read-once-BP  $G$  lässt sich in ein Read-once-BP  $G'$  für dieselbe Funktion umwandeln, sodass folgende Eigenschaften erfüllt sind:*

- Für jeden Knoten  $v$  kommen auf jedem Weg vom Startknoten zu  $v$  dieselben Variablen vor (im Allgemeinen allerdings in verschiedenen Reihenfolgen);
- auf jedem Weg vom Startknoten zu einer Senke kommen alle Variablen vor; und
- $|G'| \leq 2n|G|$ .

Wir nennen Read-once-BPs mit den ersten beiden Eigenschaften regulär.

**Beweis:** Wir konstruieren induktiv das neue BP  $G'$  aus dem gegebenen. Für den Knoten  $v$  sei die zweite Eigenschaft bereits an allen Vorgängern erfüllt. Sei  $X_v$  die Vereinigung der Mengen von Variablen, die auf irgendwelchen Wegen vom Startknoten zu  $v$  vorkommen. Jede Kante von einem Vorgänger  $u$  zu  $v$  ersetzen wir durch eine Kette aus Dummy-Knoten mit den Variablen, die in  $X_v$  vorkommen, aber weder auf den Wegen zu  $u$ , noch an  $u$  selbst vorkommen. Damit ist genauer eine Folge von Knoten  $d_0 = u, d_1, \dots, d_{k-1}, d_k = v$  gemeint, wo für  $i = 0, \dots, k-1$  die 0- und 1-Kanten von  $d_i$  zu  $d_{i+1}$  führen und  $d_1, \dots, d_{k-1}$  mit den fehlenden Variablen markiert sind. Auf dieselbe Weise ergänzen wir für jede Senke alle noch fehlenden Variablen durch

Ersetzen der zu ihr führenden Kanten. Aufgrund der Read-once-Eigenschaft wissen wir, dass auf Wegen von  $v$  zu einer Senke, inklusive  $v$  selbst, keine Variablen aus  $X_v$  mehr vorkommen, sodass diese Konstruktion wieder ein korrektes Read-once-BP liefert. Es ist auch klar, dass wir die dargestellte Funktion nicht verändern. Da die Anzahl der Kanten in  $G$  durch  $2|G|$  nach oben abgeschätzt werden kann und für jede Kante höchstens  $n - 1$  Dummy-Knoten ergänzt werden, stimmt die Abschätzung der Größe.  $\square$

Wir können uns also bei der Entwicklung einer Unteren-Schranken-Technik auf reguläre Read-once-BPs beschränken, da für uns hier polynomielle Faktoren bei der Größe nicht wichtig sind. Ein guter Ausgangspunkt ist die kombinatorische Sichtweise der Beweistechnik für OBDDs. Wir definieren hier einen Schnitt  $C$  durch ein gegebenes reguläres Read-once-BP  $G$ , der aus den Knoten besteht, die nach Lesen von genau  $n/2$  Variablen auf den Rechenwegen erreicht werden. Sei wieder  $R_v$  die Menge  $R_v$  von Eingaben, deren Rechenweg vom Startknoten über  $v$  zur (einzigsten) 1-Senke führt. Da auf allen Teilwegen vom Typ „Startknoten  $\rightarrow v$ “ dieselbe Variablenmenge vorkommt, analog auf allen Teilwegen vom Typ „ $v \rightarrow$  1-Senke“, erweist sich  $R_v$  genauso wie bei OBDDs als 1-Rechteck (hier haben wir die Regularität von  $G$  ausgenutzt). Außerdem bilden alle 1-Rechtecke zusammen eine Partition der Menge der 1-Eingaben der dargestellten Funktion. Der große Unterschied zu OBDDs ist, dass jedes Rechteck zu einer anderen Aufteilung der Eingabe gehören kann, da die Menge der Variablen auf den Anfangsstücken der Wege (bzw. analog auf den Endstücken) je nach Knoten  $v$  unterscheiden können. Wir haben also kein Kommunikationsprotokoll bezüglich einer festen Partition mehr, das zu den Rechtecken korrespondiert.

Wir beschreiben zunächst die auftretenden Rechtecke genauer und halten den Zusammenhang zu Read-once-BPs dann in einem Satz fest.

**Definition:** Sei  $\Pi = (X_1, X_2)$  eine Partition der endlichen Variablenmenge  $X$ . Ein (*kombinatorisches*) Rechteck bezüglich  $\Pi$  ( $\Pi$ -Rechteck) wird beschrieben durch eine charakteristische Funktion  $R$  auf den Variablen  $X$ , die sich in der Form  $R = R_1 \wedge R_2$  schreiben lässt, wobei für  $i = 1, 2$  die Funktion  $R_i$  nur von den Variablen in  $R_i$  abhängt.

**Satz:** Jedes Read-once-BP  $G$  für eine Funktion  $f$  liefert eine Partition der Eingabemenge in höchstens  $2n|G|$  Rechtecke, wobei jedes Rechteck seine eigene balancierte Partition der Eingabvariablen haben darf.

**Beweis:** Wir wandeln  $G$  in ein reguläres Read-once-BP der Größe höchstens  $2n|G|$  um und definieren dann die Rechteckzerlegung wie vorab beschrieben. Da es höchstens  $2n|G|$  Schnittknoten gibt, folgt auch die Aussage über die Anzahl der Rechtecke.  $\square$

Wir können mit Hilfe dieses Satzes untere Schranken für  $|G|$  zeigen, wenn wir nachweisen können, dass jede Rechtecküberdeckung für die gewünschte Funktion mit variablen Partitionen der Eingabe viele Rechtecke benötigt. Solche Schranken erhalten wir gemäß einer nahe liegenden Verallgemeinerung der Rechteckmaßmethode, wenn wir zeigen können, dass 1-Rechtecke

für die betrachteten Funktion  $f$  mit einer *beliebigen balancierten Partition* nur wenige Eingaben enthalten können, während andererseits die Zahl der 1-Eingaben von  $f$  groß ist. Typischerweise sind die entsprechenden Argumente natürlich anspruchsvoller als für eine feste, für den Nachweis von unteren Schranken gut geeignete Partition.

Wir halten außerdem fest, dass wir aus unteren Schranken für die Anzahl von Rechtecken in Rechtecküberdeckungen analog zur obigen Konstruktion tatsächlich sogar untere Schranken für nichtdeterministische Read-once-BPs erhalten. Außerdem kann man mit entsprechende Verallgemeinerungen von Korruptions- und Diskrepanztechnik auch untere Schranken für randomisierte Read-once-BPs nachweisen (hier ohne Details).

Wir betrachten als konkretes Beispiel einer unteren Schranke für Read-once-BPs charakteristische Funktionen von Codes. Ein (*binärer*) *Code der Länge  $n$*  ist eine Teilmenge  $C \subseteq \{0, 1\}^n$ . Der *Minimalabstand von  $C$*  ist definiert als Minimum des Hammingabstandes zwischen zwei verschiedenen Codewörtern. Falls  $d = 2t + 1$  der Minimalabstand ist und ein Codewort über einen unsicheren Kanal übertragen wird, dann kann man bei bis zu  $t$  fehlerhaften Bits immer noch das ursprüngliche Codewort eindeutig rekonstruieren.

Wir beobachten, dass sich durch den Minimalabstand eine obere Schranke für die Bepackung von  $\{0, 1\}^n$  durch Codewörter mit folgendem einfachen Volumenargument ergibt. Für je zwei verschiedene Codewörter  $x, y \in C$  eines solchen Codes gilt, dass die Kugeln um  $x, y$  mit Radius  $t$  bezüglich des Hammingabstandes sich nicht schneiden. Die Größe einer solchen Kugel ist genau  $\sum_{i=0}^t \binom{n}{i}$ . Damit erhalten wir:

**Lemma (Hammingsschranke):** *Sei  $C$  ein Code der Länge  $n$  mit Minimalabstand  $d = 2t + 1$ . Dann gilt  $|C| \leq 2^n / B(n, t)$ , wobei  $B(n, t) := \sum_{i=0}^t \binom{n}{i}$ .*

Für Anwendungen ist es gut, Codes mit hohem Minimalabstand und gleichzeitig vielen Codewörtern zu haben. Wir zeigen hier, dass es für Read-once-BPs schwer ist, zu überprüfen, ob ein Wort zu solch einem Code gehört.

**Lemma:** *Sei  $C$  ein Code der Länge  $n$  mit Minimalabstand  $2t + 1$ . Sei  $f_C$  die charakteristische Funktion von  $C$ . Dann hat jedes Read-once-BP für  $f_C$  Größe mindestens  $(1/2n) \cdot |C|/2^n \cdot B(\lfloor n/2 \rfloor, t)^2$ .*

**Beweis:** Nach unseren Vorüberlegungen reicht es aus, wenn wir zeigen, dass jede Überdeckung der 1-Eingaben von  $f_C$  durch 1-Rechtecke mit variabler Eingabepartition mindestens  $|C|/2^n \cdot B(\lfloor n/2 \rfloor, t)^2$  solcher Rechtecke enthält. Dazu wenden wir die verallgemeinerte Rechteckmaßmethode an. Zunächst halten wir fest, dass natürlich  $f_C^{-1}(1) = |C|$  gilt. Wir zeigen nun, dass jedes 1-Rechteck für  $f_C$  bezüglich einer beliebigen Partition der Eingabevariable von  $f_C$  höchstens  $2^n / B(\lfloor n/2 \rfloor, t)^2$  Eingaben enthält. Dann folgt die behauptete untere Schranke.

Sei also  $R = R_1 \wedge R_2$  ein 1-Rechteck für  $f_C$  bezüglich einer balancierten Partition  $\Pi = (X_1, X_2)$  der  $n$  Eingabevariablen. Also gilt insbesondere  $R^{-1}(1) \subseteq f_C^{-1}(1)$ . Sei  $R(a, b) = 1$  für Belegungen  $a, b$  von  $X_1$  bzw.  $X_2$ . Dann gilt für jede Belegung  $b'$  von  $X_2$  mit  $b' \neq b$  und  $R(a, b') = 1$ , dass ihr Hammingabstand zu  $b$  mindestens  $d$  beträgt. Also stellen alle Belegungen

in  $R_2^{-1}(1)$  einen Code der Länge  $|X_2|$  mit Minimalabstand höchstens  $d$  dar, woraus gemäß der Hammingsschranke

$$|R_2^{-1}(1)| \leq 2^{|X_2|}/B(|X_2|, t) \leq 2^{|X_2|}/B(\lfloor n/2 \rfloor, t)$$

folgt. Analog erhalten wir  $|R_1^{-1}(1)| \leq 2^{|X_1|}/B(\lfloor n/2 \rfloor, t)$ . Insgesamt gilt also

$$|R^{-1}(1)| \leq 2^n/B(\lfloor n/2 \rfloor, t)^2.$$

□

Es bleibt nun, das Lemma auf konkrete Codes anzuwenden. Wir betrachten BCH-Codes (Bose-Chaudhuri-Hocquenghem-Codes), von denen wir nur die benötigten Eigenschaften zitieren. Für  $n = 2^m - 1$  und  $t \in \mathbb{N}$  gibt es Codes dieses Typs mit Länge  $n$ , Minimalabstand mindestens  $d = 2t + 1$  und mindestens  $2^n/(n+1)^t$  Codewörtern. Sei  $BCH_n$  die charakteristische Funktion eines solchen Codes mit  $t = t(n) = \lfloor n^{1/2} \rfloor$ . Dann folgt:

**Satz:** *Jedes Read-once-BP für  $BCH_n$  hat Größe  $2^{\Omega(n^{1/2})}$ .*

**Beweis:** Es ist zunächst

$$B(\lfloor n/2 \rfloor, t) = \sum_{i=0}^t \binom{\lfloor n/2 \rfloor}{i} \leq (t+1) \binom{\lfloor n/2 \rfloor}{t}.$$

Mit Hilfe der stirlingschen Formel erhalten wir außerdem (wir sparen uns hier die etwas häßlichen Rechnungsdetails) für eine geeignete Konstante  $\alpha$  mit  $0 < \alpha < \log(e/2)$  und hinreichend große  $n$ :

$$\binom{\lfloor n/2 \rfloor}{t} = \frac{1}{e\sqrt{2\pi}} \cdot \frac{1}{n^{1/4}} \cdot ((e/2)n^{1/2})^{n^{1/2}} \cdot (1 \pm o(1)) \geq 2^{\alpha n^{1/2}} \cdot n^{(1/2)n^{1/2}}.$$

Damit ergibt sich insgesamt:

$$|C|/2^n \cdot B(\lfloor n/2 \rfloor, t)^2 \geq \frac{1}{(n^{1/2} + 1)^2} \frac{1}{(n+1)^{n^{1/2}}} \cdot 2^{2\alpha n^{1/2}} \cdot n^{n^{1/2}} = 2^{2\alpha n^{1/2}} \cdot (1 \pm o(1))$$

und damit zusammen mit dem vorhergehenden Lemma die Behauptung des Satzes. □

## 8.4 Längenbeschränkte Branchingprogramme

Das Ziel in diesem Abschnitt ist es, einen kleinen Einblick in die fortgeschrittenste Beweistechnik für BPs zu geben, mit deren Hilfe sich insbesondere erstmals superlineare Zeitschranken bei sublinear beschränktem Speicherplatz für allgemeine Registermaschinen-Algorithmen zeigen ließen, die boolesche Funktionen berechnen bzw. Entscheidungsprobleme lösen.

Wir stellen hier eine einfache Version der Technik vor, die nur für Eingaben funktioniert, die durch Variablen aus einem Wertebereich  $\{0, \dots, d\}$  mit großem, von der Eingabelänge abhängigen  $d$  codiert werden – also insbesondere nicht für boolesche Variablen. Ein Beispiel für ein Problem, das sich auf natürliche Weise so codieren lässt, ist das *Element-Distinctness-Problem*, bei dem für Eingaben  $x_1, \dots, x_n \in \{0, \dots, d\}$  entschieden werden soll, ob alle Zahlen  $x_1, \dots, x_n$  paarweise verschieden sind.

Für die Beweistechnik ist es einfacher, dual zum eigentlich gewünschten Ergebnis vorzugehen und untere Schranken für den Platzverbrauch bei beschränkter Rechenzeit zu zeigen. Wie wir aus dem Einleitungsabschnitt wissen, können wir außerdem Rechenzeitschranken durch gleich große Zeitschranken ersetzen, wenn wir einen polynomiellen Größen-Blowup tolerieren. Damit können wir uns im Folgenden auf die Untersuchung von längenbeschränkten BPs zurückziehen, wobei die Längenschranken für die hier betrachtete Technik von der Form  $n \cdot \text{poly}(\log n)$  sind.

Die Technik ist wiederum eine Variante der Rechteckmaß-Methode und eine deutliche Verallgemeinerung der Ideen für Read-once-BPs. Wir verwenden hier eine neue Art von Rechtecken, die die üblichen kombinatorischen Rechtecke als Spezialfall enthält. Wir werden nachweisen, dass ein allgemeines BP eine Überdeckung der 1-Eingaben der betrachteten Funktion  $f$  mit nicht zu vielen solcher Rechtecke liefert.

Der Beweis dieser Aussage besteht aus zwei Schritten. Im ersten Schritt zeigen wir, dass das BP einfacher strukturierte Teil-BPs liefert, die den Eingaberaum partitionieren, so genannte *Entscheidungswälder*. Im zweiten Schritt überdecken wir die Eingaben von Entscheidungswäldern durch passende Rechtecke.

Wir betrachten wir im Folgenden Funktionen der Bauart  $f: D^n \rightarrow \{0, 1\}$ , die über Variablen aus  $X = \{x_1, \dots, x_n\}$  mit Werten aus  $D = \{0, \dots, d\}$  definiert sind.

**Definition 8.1:** Ein *Entscheidungsbaum* ist ein BP, in dem jeder Knoten Eingangsgrad 1 hat. Wir bezeichnen die von einem Entscheidungsbaum  $T$  berechnete Funktion ebenfalls mit  $T$ . Ein  $(r, k)$ -*Entscheidungswald* ist eine Menge  $F = \{T_1, \dots, T_r\}$  von Entscheidungsbäumen, wo für jede Eingabe  $a$  und jedes  $T_i$  die Anzahl der Variablen auf dem für  $a$  in  $T_i$  durchlaufenen Rechenweg durch  $\lceil kn/r \rceil$  nach oben beschränkt ist. Die von  $F$  berechnete Funktion sei  $F := \bigwedge_{1 \leq i \leq r} T_i$ . Wir sagen, dass für eine Eingabe  $a$  eine Variable von einem Entscheidungsbaum (oder Entscheidungswald) *gelesen* wird, wenn sie auf dem Rechenweg für  $a$  (bzw. auf einem der Rechenwege in den im Entscheidungswald enthaltenen Entscheidungsbäumen) vorkommt.

Wir müssen im Folgenden darauf achten, dass wir tatsächlich hier immer nur Rechenwege bzw. bei Rechnungen gelesene Variablen betrachten – anders als bei OBDDs und Read-once-BPs können hier jetzt inkonsistente Wege vorkommen. Der erste Schritt der Zerlegung von BPs wird nun in folgendem Lemma beschrieben. Die verlangte Voraussetzung über die Länge erhalten wir aus dem bereits erwähnten Lemma im Einführungsabschnitt, mit dem wir Rechenzeitschranken in Längenschranken übersetzen.

**Lemma 8.2 (Zerlegung von BPs in Entscheidungswalder):** *Seien  $r, k \in \mathbb{N}$  mit  $r \leq kn$ . Sei  $G$  ein BP fur  $f : D^n \rightarrow \{0, 1\}$ , in dem jeder Weg Lange genau  $\ell \leq kn$  hat (insbesondere ist  $G$  in Ebenen eingeteilt, d. h. Kanten verlaufen nur von einer Ebene zur nachsten). Dann gibt es eine Kollektion aus hochstens  $|G|^{r-1}$   $(r, k)$ -Entscheidungswaldern, sodass die Mengen der von diesen akzeptierten Eingaben eine Partition von  $f^{-1}(1)$  bilden.*

**Beweis:** Der Beweis funktioniert ahnlich wie die Beweistechnik-Lemmas fur OBDDs und Read-once-BPs, nur benutzen wir diesmal  $r - 1$  Schnitte im Graphen anstatt nur einem einzigen. Wir beginnen mit einigen Definitionen. Fur zwei verschiedene Knoten  $v, w$  in  $G$  definieren wir die boolesche Funktion  $f_{v,w}$  auf den Eingabevariablen wie folgt. Fur eine Eingabe  $a$  sei  $f_{v,w}(a) := 1$ , falls ein am Knoten  $v$  gestarteter Rechenweg gema der BP-Semantik den Knoten  $w$  erreicht, und  $f_{v,w}(a) := 0$  sonst. Seien die gema Voraussetzung gegebenen Ebenen in  $G$  mit  $0, \dots, \ell$  durchnummeriert. Fur  $i = 1, \dots, r - 1$  definiere den Schnitt  $C_i$  als die Menge aller Knoten der Ebene  $i \lceil \ell/r \rceil$ . Sei  $v_0$  der Startknoten von  $G$  und  $v_r$  die 1-Senke. Dann ist  $f$  offenbar die Disjunktion der Funktionen  $F_{v_1, \dots, v_{r-1}} = \bigwedge_{0 \leq i \leq r-1} f_{v_i, v_{i+1}}$  uber alle Schnittknotenfolgen  $(v_1, \dots, v_{r-1}) \in C_1 \times \dots \times C_{r-1}$ . Weiterhin sind die Mengen der von diesen Funktionen akzeptierten Eingaben disjunkt (jede Eingabe durchlauft genau eine Knotenfolge). Die Anzahl der Funktionen kann durch  $|C_1| \cdot \dots \cdot |C_{r-1}| \leq |G|^{r-1}$  nach oben abgeschatzt werden. Schlielich wird jede der Funktionen  $f_{v_i, v_{i+1}}$  von einem Entscheidungsbaum der Tiefe  $\lceil kn/r \rceil$  berechnet, den wir durch Expandieren des Teil-BPs aus allen Wegen von  $v_i$  nach  $v_{i+1}$  zu einem Baum erhalten, wobei alle zu  $v_{i+1}$  korrespondierenden Senken 1-Senken werden und alle restlichen Senken 0-Senken. Damit wird auch jede Funktion  $F_{v_1, \dots, v_{r-1}}$  von einem  $(r, k)$ -Entscheidungswald berechnet und wir haben insgesamt das Lemma gezeigt.  $\square$

Der zweite Schritt ist nun, jedem Entscheidungswald eine Rechteckuberdeckung der Eingaben zuzuordnen (bei OBDDs und Read-once-BPs hatten wir an dieser Stelle nur genau ein zugeordnetes Rechteck und der Schritt war trivial). In Analogie zur Kommunikationskomplexitat und den Techniken fur OBDDs und Read-once-BPs wurden wir am liebsten Teilgraphen im BP finden, sodass wir die dort auf den Rechenwegen auftauchenden Variablen wieder exklusiv einem der zwei Spieler Alice und Bob zuordnen konnen. Unsere Idee ist daher, zunachst auf irgendeine Weise jeden der Entscheidungsbaume in einem Entscheidungswald einem der beiden Spieler zuzuordnen. Allerdings wird es dann typischerweise (in einem allgemeinen BP) viele Variablen geben, die sowohl in Alices' als auch in Bobs Baumen gelesen werden und wir erhalten keine disjunkte Partition der Variablen.

Trotzdem besteht die Hoffnung, dass es zumindest einige Variablen gibt, die exklusiv nur bei Berechnungen in Alices' Baumen vorkommen und einige nur bei Berechnungen in Bobs Baumen. Alle anderen Variablen mussen wir dann irgendwie geschickt durch Konstantsetzungen eliminieren. Der Grund fur diese Hoffnung liegt darin, dass die Baume nicht sehr tief sind und es daher nicht zu viele Variablenpaare gibt, deren Bestandteile gemeinsam in einem festen Baum vorkommen. Im Folgenden werden wir diese Intuition prazisieren. Dazu definieren wir zunachst die Mengen der uns interessierenden, exklusiv gelesenen Variablen.

**Definition 8.3:** Für einen Entscheidungswald  $F$  und einen Teilwald  $F' \subseteq F$ , sei der  $F'$ -Kern für  $a$ ,  $\text{core}_F(a, F')$ , die Menge der Variablen, die bei der Berechnung für  $a$  exklusiv in den Bäumen in  $F'$  gelesen werden (d. h. nur in  $F'$  und nicht in  $F - F'$ ).

Im Folgenden lassen wir üblicherweise den Index  $F$  bei dieser Notation weg, da wir sowieso immer nur über einen festen Entscheidungswald reden. Wir haben bereits festgelegt, dass wir die Menge der Entscheidungsbäume  $F$  zwischen den Spielern aufteilen wollen. Sei  $F_A, F_B \subseteq F$  eine Partition von  $F$ , wobei Alice die Bäume in  $F_A$  erhält und Bob die in  $F_B$ . Unser Ziel ist es, zu zeigen, dass sowohl in  $F_A$  als auch in  $F_B$  viele Variable exklusiv gelesen werden, also formal mit der obigen Definition, dass  $\text{core}(a, F_A)$  und  $\text{core}(a, F_B)$  groß sind.

Wir haben allerdings keine Ahnung, wie man  $F_A$  und  $F_B$  geschickt wählen sollte, damit das gilt. Wir verwenden daher die probabilistische Methode zur Auswahl. Wir vergeben jeden der Bäume unabhängig und zufällig mit einem fairen Münzwurf an einen der beiden Spieler. Damit erhalten wir eine zufällige Partition  $(F_A, F_B)$  von  $F$ . Wir zeigen nun, dass tatsächlich dann mit hoher Wahrscheinlichkeit  $\text{core}(a, F_A)$  und  $\text{core}(a, F_B)$  groß sind.

**Lemma 8.4:** Sei  $F$  ein  $(r, k)$ -Entscheidungswald, wobei  $r \leq n$  ist. Sei  $a \in \{0, 1\}^n$  eine Eingabe für  $F$ . Sei  $(F_A, F_B)$  eine zufällige, wie oben beschrieben erzeugte Partition von  $F$ . Dann haben  $\text{core}(a, F_A)$  und  $\text{core}(a, F_B)$  die gleiche erwartete Größe  $\mu(a) \geq n/2^{k+1}$  und es gilt  $|\text{core}(a, F_A)| - \mu(a)| \geq \mu(a)/2$  mit Wahrscheinlichkeit höchstens  $4(k+1)^2 2^{2(k+1)}/r$ . Eine analoge Aussage erhalten wir für  $F_B$ .

**Beweis:** Wegen Symmetrie reicht es aus, nur  $F_A$  zu betrachten. Sei  $t(i)$  die Anzahl von Bäumen in  $F$ , die die Variable  $x_i$  während der Berechnung für  $a$  lesen. Sei  $Z_i$  die Indikator-Zufallsvariable für das Ereignis  $x_i \in \text{core}(a, F_A)$  und sei  $Z := Z_1 + \dots + Z_n$ . Dann ist offensichtlich  $\mu(a) = E(Z)$ . Aufgrund der Definition der zufälligen Partition  $(F_A, F_B)$  gilt für alle  $i$ , dass  $\Pr\{Z_i = 1\} = 2^{-t(i)}$  ist. Damit haben wir

$$\mu(a) = E(Z) = \sum_{i=1}^n 2^{-t(i)}.$$

Da  $F$  ein  $(r, k)$ -Entscheidungswald ist, haben wir weiterhin

$$t(1) + \dots + t(n) \leq r \lceil kn/r \rceil \leq r(kn/r + (r-1)/r) \stackrel{r \leq n}{\leq} (k+1)n.$$

Wir wollen die Summe  $\sum_i 2^{-t(i)}$  nach unten beschränken und berechnen daher ihr Minimum als Funktion von reellen Variablen  $t(1), \dots, t(n)$  unter der Nebenbedingung  $t(1) + \dots + t(n) \leq (k+1)n$ . Es ist aufgrund der Symmetrie der Funktion in den Variablen klar, dass das maximal mögliche Gesamtgewicht von  $(k+1)n$  möglichst gleichmäßig auf die Variablen verteilt werden sollte, d. h. das Minimum wird für  $t(1) = \dots = t(n) = k+1$  angenommen und beträgt damit  $n/2^{k+1}$ . Damit haben wir den ersten Teil des Lemmas gezeigt.

Für den zweiten Teil benutzen wir die tschebyscheffsche Ungleichung. Dazu leiten wir zunächst eine obere Schranke für die Varianz von  $Z$  her. Es ist aufgrund der Definition der Varianz und von  $Z$  zunächst

$$\begin{aligned} V(Z) &= E(Z^2) - (EZ)^2 = \sum_{1 \leq i, j \leq n} E(Z_i Z_j) - \sum_{1 \leq i, j \leq n} (EZ_i)(EZ_j) \\ &= \sum_{1 \leq i, j \leq n} (\Pr\{Z_i \wedge Z_j = 1\} - \Pr\{Z_i = 1\} \cdot \Pr\{Z_j = 1\}). \end{aligned}$$

Falls es keinen Baum in  $F$  gibt, in dem beiden Variablen  $x_i$  und  $x_j$  gelesen werden, dann sind  $Z_i$  und  $Z_j$  unabhängige Zufallsvariablen und für dieses Paar  $(i, j)$  ist der entsprechende Term in der obigen Summe 0. Wir schätzen die restlichen Terme brutal mit 1 nach oben ab und zählen, wieviele wir davon haben. Für ein festes  $i$  gibt es höchstens  $t(i) \cdot \lceil kn/r \rceil$  Variablen  $x_j$ , die zusammen mit  $x_i$  in einem der Bäume gelesen werden. Damit gibt es in der obigen Summe insgesamt höchstens

$$\sum_{i=1}^n t(i) \lceil kn/r \rceil \leq (k+1)^2 n^2 / r$$

von 0 verschiedene Terme. Dies ist also auch eine obere Schranke für  $V(Z)$ . Gemäß der Tschebyscheff-Ungleichung erhalten wir nun

$$\Pr\{|Z - E(Z)| \geq E(Z)/2\} \leq 4V(Z)/E(Z)^2 \leq 4(k+1)^2 2^{2(k+1)} / r,$$

wie behauptet. □

Sei  $F$  ein fester Entscheidungswald. Wir setzen im obigen Lemma  $r = 16(k+1)^2 2^{2(k+1)}$ . Dann erhalten wir für eine feste Eingabe  $a$  mit Wahrscheinlichkeit mindestens  $1/2$  eine Partition  $(F_A, F_B)$  von  $F$ , sodass für  $X_1 := \text{core}(a, F_A)$  und  $X_2 := \text{core}(a, F_B)$  gilt, dass  $|X_1|, |X_2| \geq m$  mit  $m = n/2^{k+2}$ . Damit können wir eine solche Partition fixieren. Wir haben nun die gewünschten Mengen der exklusiv in Alice- bzw. Bob-Bäumen gelesenen Variablen gefunden. Wir müssen nun noch überlegen, wie wir die restlichen Variablen in  $X - (X_1 \cup X_2)$  behandeln.

Dazu fassen wir zunächst die Eingaben zusammen, für die die dieselben Mengen  $(X_1, X_2)$  als Kerne auftreten. (Bei OBDDs hatten wir an dieser Stelle nur eine feste Wahl für alle Eingaben.) Für zwei disjunkte Teilmengen  $X_1, X_2$  der Variablen sei  $P := P(X_1, X_2, F_A, F_B)$  die Menge aller Eingaben  $a \in F^{-1}(1)$  mit  $X_1 \subseteq \text{core}(a, F_A)$  und  $X_2 \subseteq \text{core}(a, F_B)$ . Gemäß dem letzten Lemma ist jede Eingabe  $a \in F^{-1}(1)$  in mindestens einer solchen Menge für  $X_1, X_2$  mit  $|X_1| = |X_2| = m$  enthalten. Beachte, dass es im Allgemeinen mehr als eine  $P$ -Menge gibt, die eine Eingabe überdeckt. Sei  $z$  eine partielle Belegung aller „nicht erwünschten“ Variablen in  $X - (X_1 \cup X_2)$  und sei  $P_z$  die Menge aller partiellen Belegungen von  $X_1 \cup X_2$ , die zusammen mit  $z$  eine Belegung ergeben, die in  $P$  enthalten ist.

**Lemma:** Für beliebige  $z$  ist die Menge  $P_z$  ein (eventuell leeres) kombinatorisches Rechteck bezüglich der Partition  $(X_1, X_2)$ .

Für Belegungen  $u$  und  $v$  von disjunkten Variablenmenge bezeichnen wir im Folgenden mit  $uv$  die aus  $u$  und  $v$  zusammengesetzte Belegung von allen Variablen in der Vereinigung dieser Mengen.

**Beweis:** Seien  $x_0$  und  $y_0$  beliebige Belegungen der Variablen in  $X_1$  bzw.  $X_2$ . Sei  $P_{z,A}$  die Menge aller Belegungen  $x$  der Variablen in  $X_1$  mit  $F_A(x_0y_0z) = 1$  und  $X_1 \subseteq \text{core}(x_0y_0z)$ . Sei analog  $P_{z,B}$  die Menge aller Belegungen  $y$  von  $X_2$  mit  $F_B(x_0y_0z) = 1$  und  $X_2 \subseteq \text{core}(x_0y_0z)$ . Wir behaupten, dass dann

$$P_z = P_{z,A} \times P_{z,B},$$

womit das Lemma gezeigt ist, da die Menge auf der rechten Seite offensichtlich ein Rechteck bezüglich  $(X_1, X_2)$  ist. Dazu beobachten wir, dass gemäß Definition  $P_z$  genau aus den Eingaben  $xyz$  besteht, wo  $x$  eine Belegung von  $X_1$  und  $y$  eine Belegung von  $X_2$  ist, sodass

$$\begin{aligned} xyz \in F_A^{-1}(1) \wedge X_1 \subseteq \text{core}(xyz, F_A) \wedge \\ xyz \in F_B^{-1}(1) \wedge X_2 \subseteq \text{core}(xyz, F_B). \end{aligned}$$

Da  $X_2 \subseteq \text{core}(xyz, F_B)$  ist, können wir in den Bedingungen der ersten Zeile  $y$  durch eine beliebige Belegung  $y_0$  von  $X_2$  ersetzen (diese Bedingungen hängen nicht von den nur in  $F_B$ -Bäumen gelesenen  $X_2$ -Variablen ab), analog für die zweite Zeile und eine beliebige Belegung  $x_0$  von  $X_1$ . Die resultierenden Bedingungen sind dann aber offensichtlich äquivalent dazu, dass  $P_z$  von der oben angegebenen Form ist.  $\square$

Gemäß unserer bisherigen Erkenntnisse können wir also nun  $F^{-1}(1)$  durch Mengen der Bauart  $P(X_1, X_2, F_A, F_B)$  überdecken (im Allgemeinen nicht disjunkt) und letztere dann wiederum durch Fixieren der Variablen in  $X - (X_1 \cup X_2)$  auf alle möglichen Weisen in kombinatorische Rechtecke bezüglich  $(X_1, X_2)$  zerlegen. Wir müssen uns nur noch überlegen, wieviele Rechtecke auf diese Weise entstehen.

Zunächst führen wir einen Namen für die Objekte  $P(X_1, X_2, F_A, F_B)$  ein, wobei es sich um verallgemeinerte Rechtecke handelt, bei denen es einen öffentlich fixierten Eingabeteil gibt, der beiden Spielern bekannt ist.

**Definition 8.5:** Seien  $X_1, X_2 \subseteq X$  disjunkte Variablenmengen mit  $|X_1| = |X_2| = m$ . Eine Eingabemenge  $R$  heißt *m-Rechteck bezüglich  $(X_1, X_2)$* , falls es eine Belegung  $z$  von  $X - (X_1 \cup X_2)$  sowie Mengen von Belegungen  $R_A$  und  $R_B$  von  $X_1$  bzw.  $X_2$  gibt, sodass  $R = R_A \times R_B \times \{z\}$ . Eine Eingabemenge  $P$  heißt *m-Pseudorechteck bezüglich  $(X_1, X_2)$* , falls für jede Belegung  $z$  von  $X - (X_1 \cup X_2)$  die Menge  $P_z \times \{z\}$  ein *m-Rechteck* bezüglich  $(X_1, X_2)$  ist.

Damit können wir nun den zweiten Schritt der Beweistechnik auch quantitativ beschreiben:

**Lemma 8.6:** Sei  $k \leq n$ ,  $r = 16(k+1)^2 2^{2(k+1)} \leq n$  und  $m = n/2^{k+2}$ . Sei  $F$  ein  $(r, k)$ -Entscheidungswald. Dann gibt es eine Kollektion von höchstens  $2^{4(k+2)m+r}$  *m-Pseudorechtecken*, die  $F^{-1}(1)$  überdecken.

**Beweis:** Wir brauchen nur noch die Anzahl der Pseudorechtecke zu zählen, die in unserer bereits beschriebenen Konstruktion vorkommen. Jede Eingabe  $a \in F^{-1}(1)$  gehört gemäß Lemma 8.4 zu (mindestens) einer Menge  $P(X_1, X_2, F_A, F_B)$  mit  $|X_1| = |X_2| = m$ . Offensichtlich gibt es höchstens  $2^r \binom{n}{m}^2$  solcher Mengen. Gemäß einer üblichen Abschätzung für Binomialkoeffizienten gilt (siehe z. B. „Extremal Combinatorics“, S. Jukna):

$$\binom{n}{m} \leq 2^{H_2(m/n)n},$$

wobei  $H_2$  die binäre Entropiefunktion ist (siehe Kapitel 4). Weiterhin erhält man mit Hilfe der Taylorreihe für  $\ln(1 - x)$  für  $x \leq 1/2$  die Abschätzung

$$H(x) \leq -2x \log x.$$

Insgesamt gilt damit

$$2^r \binom{n}{m}^2 \leq 2^r \cdot \left(2^{2 \cdot 2^{-(k+2)} \cdot (k+2) \cdot n}\right)^2 = 2^r \cdot \left(2^{2m(k+2)}\right)^2 = 2^{4(k+2)m+r}.$$

□

Analog zur Rechteckmaßmethode für die Gleichverteilung liefert uns die hier hergeleitete Beweistechnik folgenden Zusammenhang zwischen dem Maß von 1-Rechtecken und der Größe von allgemeinen Branchingprogrammen:

**Satz 8.7:** Sei  $2 \leq k \leq n$ ,  $r = 16(k+1)^2 2^{2(k+1)} \leq n$  und  $m = n/2^{k+2}$ . Sei  $G$  ein Branchingprogramm für  $f: D^n \rightarrow \{0, 1\}$  der Länge höchstens  $kn$ . Dann gibt es ein  $m$ -Rechteck  $R \subseteq f^{-1}(1)$ , sodass

$$\frac{|R|}{|D|^{2m}} \geq 2^{-4(k+2)m-r} \cdot |G|^{-r} \cdot \frac{|f^{-1}(1)|}{|D|^n}.$$

**Beweis:** Gemäß dem Zerlegungslemma 8.2 und dem Lemma 8.6 über die Anzahl von Pseudorechtecken bei der Überdeckung eines Entscheidungswaldes erhalten wir eine Überdeckung von  $f^{-1}(1)$  mit höchstens  $2^{4(k+1)m+r} \cdot |G|^r$   $m$ -Pseudorechtecken. Damit gibt es mindestens ein solches Rechteck  $P$  mit einer Partition  $(X_1, X_2)$  mit  $|X_1| = |X_2| = m$  und

$$|P| \geq 2^{-4(k+2)m-r} \cdot |G|^{-r} \cdot |f^{-1}(1)|.$$

Durch Fixieren der Variablen in  $X - (X_1 \cup X_2)$  können wir  $P$  in genau  $|D|^{n-2m}$   $m$ -Rechtecke partitionieren. Dann gibt es mindestens ein solches  $m$ -Rechteck  $R$  mit

$$|R| \geq \frac{|P|}{|D|^{n-2m}} \geq 2^{-4(k+2)m-r} \cdot |G|^{-r} \cdot \frac{|f^{-1}(1)|}{|D|^n} \cdot |D|^{2m}.$$

Dividieren durch  $|D|^{2m}$  liefert die Behauptung.

□

Beame, Saks, Sun und Vee (2001) haben diesen Satz auf Funktionen angewendet, die aus quadratischen Formen über endlichen Körpern konstruiert sind. Wir betrachten ein einfaches Beispiel. Sei  $n = 2^d$  und sei  $H_n$  die  $n \times n$ -Hadamardmatrix mit  $H_n(x, y) = (-1)^{\langle x, y \rangle \bmod 2}$  für  $x, y \in \{0, 1\}^n$ . Sei  $H_n^*$  die Matrix, die aus  $H_n$  entsteht, indem wir die Diagonale durch lauter Nullen ersetzen. Für jede ungerade Primzahlpotenz  $q$  und  $x \in \mathbb{Z}_q^n$  sei  $\text{SQF}_{q,n}(x) = 1$ , wenn  $x^\top H_n^* x \equiv 0 \pmod q$  und  $\text{SQF}_{q,n}(x) = 0$  sonst.

Die Hadamardmatrix  $H_n$  hat die bemerkenswerte Eigenschaft, dass jede ihrer Submatrizen großen Rang hat verglichen mit ihrer Anzahl von Einträgen. Beame, Saks, Sun und Vee haben eine entsprechende untere Schranke für den Submatrixrang von  $H_n$  und elementare lineare Algebra benutzt, um zu zeigen dass für jedes  $m$ -Rechteck  $R \subseteq \text{SQF}_{q,n}^{-1}(1)$  gilt:

$$|R|/|D|^{2m} \leq |D|^{-m^2/n}.$$

Weiterhin kann man sich leicht überlegen, dass selbst nach Fixieren von  $n - 2$  Einträgen im Variablenvektor  $x$  die Funktion  $x \mapsto x^\top S^* x$  immer noch jeden möglichen Wert in  $\mathbb{Z}_q$  annehmen kann. Damit folgt  $|\text{SQF}_{q,n}^{-1}(1)| \geq q^{n-2}$ . Wenn wir diese beiden Fakten in Satz 8.7 einsetzen, erhalten wir:

**Satz 8.8:** *Für jede Konstante  $\varepsilon > 0$  gibt es eine Konstante  $c > 0$ , sodass für  $k, n, q$  mit  $\log \log q \geq ck$  und  $n \geq 8(k+1)^2 2^{2(k+1)}$  gilt, dass jedes Branchingprogramm für  $\text{SQF}_{q,n}$  der Länge höchstens  $kn$  Größe mindestens  $2^{n \log^{1-\varepsilon} p}$  benötigt.*

Um dieses Ergebnis besser interpretieren zu können, wählen wir  $q$  als die kleinste ungerade Primzahlpotenz größer oder gleich  $n$  und setzen  $k$  auf einen Wert von der Größenordnung  $\log \log n$ . Dann ergibt sich aus dem Zusammenhang zwischen Länge und Größe von Branchingprogrammen und Rechenzeit bzw. Speicherplatz für Registermaschinen (siehe den Einleitungsabschnitt):

**Folgerung 8.9:** *Für jede Konstante  $\varepsilon > 0$  gibt es eine Konstante  $c' > 0$ , sodass jeder Algorithmus, der  $\text{SQF}_{q,n}$  auf einer Registermaschine mit Speicherplatz  $n \log^{1-\varepsilon} n$  berechnet, mindestens Rechenzeit  $c' n \log \log n$  benötigt.*

Wir erwähnen abschließend noch, dass sich die hier geschilderten Argumente direkt auch auf nichtdeterministische Branchingprogramme anwenden lassen und damit die obigen Aussagen tatsächlich sogar für dieses allgemeinere Modell bzw. nichtdeterministische Algorithmen gelten. Schließlich haben Beame, Saks, Sun und Vee auch noch eine (weitaus kompliziertere) Variante der Technik vorgestellt, die untere Schranken ähnlicher Bauart auch für allgemeine randomisierte Branchingprogramme auf *booleschen* Variablen liefert und diese (unter anderem) wiederum auf Funktionen angewendet, die auf quadratischen Formen basieren.