Analyse einer konkreten Klasse von Schlüsselgenerierungsprotokollen

Martin Gander
Institut für Theoretische Informatik
ETH Zürich
CH-8092 Zürich

24. August 1993

Inhaltsverzeichnis

1 Einleitung

In meiner Diplomarbeit [7] habe ich Protokolle untersucht, welche durch Mischen von Verbesserungs- und Verschlechterungsschritten versuchen, die Entropiedifferenz zwischen Alice und Bob und Alice und Eve aufzuschaukeln zugunsten von Alice und Bob. Diese Protokolle sind zwar einfach zu verstehen, haben aber den Nachteil, dass ihre Auswirkungen schwierig zu berechnen sind, sowohl numerisch als auch theoretisch. Man kann nur für wenige Runden die Bitfehlerwahrscheinlichkeit und Entropie von Eve vorausrechnen, was nicht ausreichend ist, wenn der Kanal von Alice und Bob massiv schlechter ist als derjenige von Eve.

In der vorliegenden Arbeit zeige ich einen einfachen Weg, den aufwendigen Punkt dieser Protokolle zu umgehen. Damit gelingt es, auch bei sehr schlechten Kanalverhältnissen, zum Beispiel haben Alice und Bob einen 1000 mal schlechteren Kanal als Eve, das Protokoll der betrachteten Protokollklasse zu berechnen, mit dem Alice und Bob einen maximalen Vorteil erzielen. Der erste Teil der Arbeit zeigt die Resultate für verschiedene Ausgangssituationen.

Der zweite Teil befasst sich mit dem Problem, wie der erreichte Vorteil von Alice und Bob ausgenützt werden kann, um einen geheimen Schlüssel zu erzeugen. Ich leite darin einerseits die obere Grenze her für die Kollisionsinformation, welche Eve durch Austauschen von Paritybits bekommen kann und zeige andererseits dieselbe Grenze für beliebige Codes, wenn man von einer Gleichverteilung ausgeht, welche leicht gestört sein darf.

2 Vereinfachung des Protokolls

In der Diplomarbeit [7] habe ich zwei einfache Protokollschritte, nämlich den Parityschritt und den Reduceschritt, eingeführt und Protokolle untersucht, die durch Kombination der beiden in mehreren Runden entstehen. Der Reduceschritt wurde einbezogen, weil man durch abwechslungsweises Verbessern und Verschlechtern die Entropiedifferenz zwischen Alice und Bob und Alice und Eve aufschaukeln wollte. Anstelle der Verschlechterung im Protokoll selber kann man aber auch mit einer schlechten Ausgangssituation starten. Dies ist einfach möglich, da die Sendeleistung des Satelliten (vergleiche dazu [7]) unter Kontrolle von Alice und Bob steht. Dadurch fallen die Reduceschritte, die zur Verschlechterung dienen, weg und es ist möglich, das Protokoll auch über viele Runden vorauszurechnen.

2.1 Bitfehlerwahrscheinlichkeit

Betrachten wir erneut das Satellitenmodell mit den Bitfehlerwahrscheinlichkeiten ϵ_A für Alice, ϵ_B für Bob und ϵ_E für Eve. Die Bitfehlerwahrscheinlichkeit zwischen Alice und Bob ist demnach zu Beginn

$$\epsilon = \delta_A\epsilon_B + \epsilon_A\delta_B,$$

wobei $\delta_i := 1 - \epsilon_i$. Nach der Durchführung von k Parityschritten beträgt die neue Bitfehlerwahrscheinlichkeit zwischen Alice und Bob

$$\beta = \frac{\epsilon^n}{\epsilon^n + (1-\epsilon)^n}, \tag{1}$$

wobei $n = 2^k$. Die Bitfehlerwahrscheinlichkeit γ von Eve ist etwas aufwendiger zu berechnen. Gemäss [7] beträgt sie

$$\gamma_{odd} = \frac{1}{\epsilon^n + (1-\epsilon)^n} \sum_{\omega=\lceil n/2\rceil}^{n} \binom{n}{\omega} p_\omega \tag{2}$$

für ungerade n und für gerade

$$\gamma_{even} = \gamma_{odd} - \frac{1}{2}\frac{1}{\epsilon^n + (1-\epsilon)^n}\binom{n}{n/2} p_{n/2}, \tag{3}$$

wobei die p_ω durch die Gleichung

$$p_\omega = \alpha_{0,0}^{n-\omega}\alpha_{0,1}^{\omega} + \alpha_{1,0}^{n-\omega}\alpha_{1,1}^{\omega}$$

und die $\alpha_{i,j}$ durch

$$\begin{aligned} \alpha_{0,0} &= \delta_A\delta_B\delta_E + \epsilon_A\epsilon_B\epsilon_E \\ \alpha_{0,1} &= \delta_A\delta_B\epsilon_E + \epsilon_A\epsilon_B\delta_E \end{aligned}$$

$$\begin{aligned}\alpha_{1,0} &= \delta_A\epsilon_B\delta_E + \epsilon_A\delta_B\epsilon_E \\ \alpha_{1,1} &= \delta_A\epsilon_B\epsilon_E + \epsilon_A\delta_B\delta_E.\end{aligned}$$

gegeben sind.

Es geht also darum, β von Bob und γ von Eve für ein Protokoll über viele k Runden auszuwerten. Dies ist nicht ganz so einfach, wie es zunächst scheinen mag. Betrachtet man die Formeln für γ genauer, so sieht man, dass für grosse k enorme Binominalkoeffitienten auszuwerten sind. Nehmen wir als Beispiel $k = 14$, dann ist der Binomialkoeffizient des ersten zu summierenden Summanden

$$\binom{2^{14}}{2^{13}} = \binom{16384}{8192} = 0.7416 * 10^{4930},$$

was die gängige Arithmetik unserer Computer bei weitem übersteigt. Der Summand hingegen wird nicht grösser als 1, er wird nämlich durch das p_ω wieder in die Schranken gewiesen, welches im obigen Beispiel bei $\epsilon_A = \epsilon_B = 0.4919$ und $\epsilon_E = 0.2506$ folgenden Wert annimmt:

$$p_{8192} = \alpha_{0,0}^{8192}\alpha_{0,1}^{8192} + \alpha_{1,0}^{8192}\alpha_{1,1}^{8192} = 0.1872 * 10^{-9862}.$$

Der entstehende Summand liegt nun auf der anderen Seite ausserhalb des Rechenbereichs und beträgt

$$\binom{16384}{8192} p_{8192} = 0.1388 * 10^{-4932}.$$

Die ganze Summe wird aber mit $\frac{1}{\epsilon^n+(1-\epsilon)^n}$ skaliert. Der Wert dafür ist in unserem Beispiel

$$\frac{1}{0.4999^{16384} + 0.501^{16384}} = 0.1615 * 10^{4931}.$$

Man sieht sofort, dass der skalierte Summand sehr wohl im Rechenbereich liegt. Er beträgt

$$0.002242.$$

Das im Anhang beschriebene Programm QA ermöglicht die effiziente Berechnung von β und γ mit der gewöhnlichen REAL-Arithmetik auf Sun-Workstations.

Die Abbildung 1 zeigt den Verlauf von β ausgezogen und von γ gestrichelt für drei verschiedene Ausgangssituationen, wobei Bob einen 1000 mal schlechteren Kanal hat als Eve.

Man sieht deutlich, dass bei geschickter Wahl der Ausgangslage nach einer bestimmten Anzahl von Parityschritten γ grösser wird als β, das heisst, Bob gewinnt einen Vorteil gegenüber Eve. Die Abbildung 1 lässt vermuten, dass man die Bitfehlerwahrscheinlichkeit ϵ_B von Bob durch Vermindern der Sendeleistung des Satelliten möglichst nahe bei 0.5

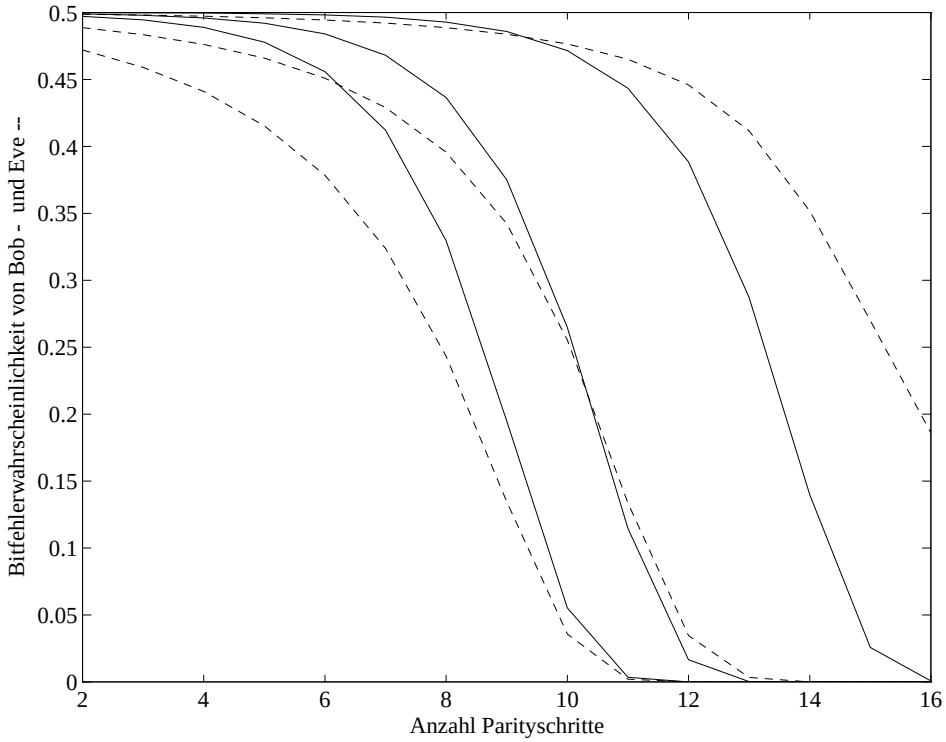

Abbildung 1: Bitfehlerwahrscheinlichkeiten β von Bob und γ von Eve für drei Ausgangssituationen ($\epsilon_B = 0.48, 0.488, 0.496$) bei Kanalverhältnis 1000

wählen muss, weil man so einen grossen Unterschied in der Bitfehlerwahrscheinlichkeit von Bob und Eve erreicht. In der Tat kann man den Vorteil beliebig gross machen, wie die Abbildung 2 zeigt.

Dies ist aber nicht unbedingt erstrebenswert. Die Figur 3 zeigt, dass grosse Unterschiede in der Bitfehlerwahrscheinlichkeit viele Bits kosten.

Der Reduktionsfaktor gibt an, um wieviel die ursprüngliche Anzahl Bits verringert wird bei einem bestimmten Protokoll (vergleiche dazu [7]). Die Abbildung 3 zeigt den Reduktionsfaktor der drei in Abbildung 1 dargestellten Protokolle. Das Protokoll, bei dem Bob ganz nahe bei 0.5 startet, ist gepunktet dargestellt. Man sieht, dass sein Reduktionsfaktor grösser ist als derjenige der anderen Protokolle. Ausserdem erreicht man die grosse Differenz in den Bitfehlerwahrscheinlichkeiten erst nach vielen Parityschritten, während bei besserer Ausgangssituation die optimale Differenz früher erreicht wird. Es gilt also, ein neues Mass zu definieren für die Güte eines Protokolls der betrachteten Protokollklasse. Ich nenne dieses Mass die Qualität eines Protokolls. Weil die Bitfehlerwahrscheinlichkeit nicht das einzige relevante Mass ist, befasst sich der folgende Abschnitt aber vorerst mit der Entropie.

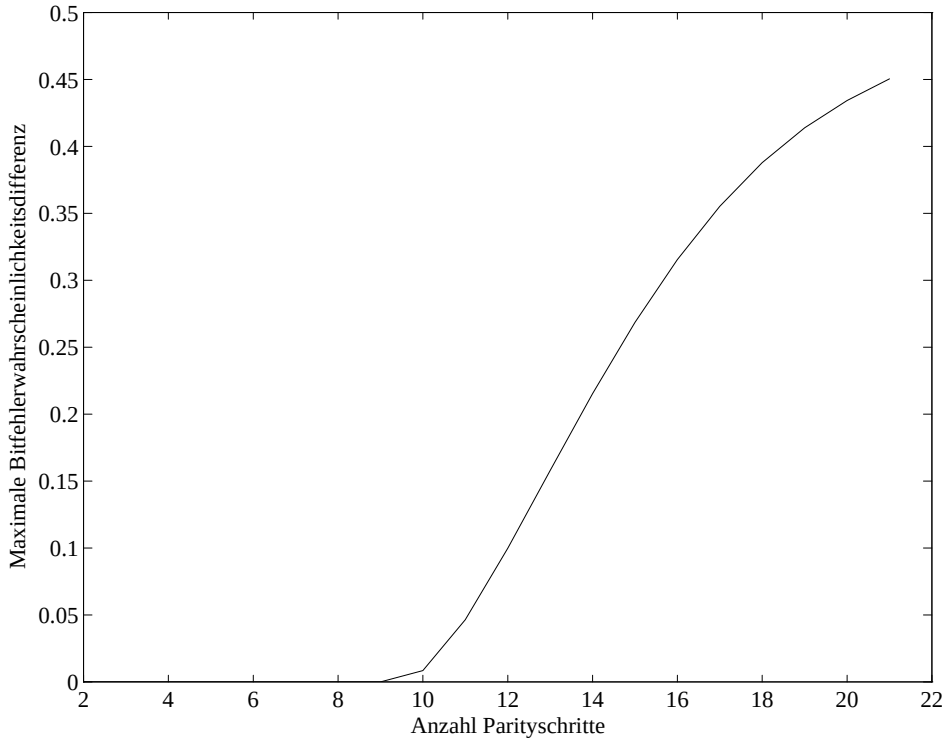

Abbildung 2: Maximal erreichbare Bitfehlerwahrscheinlichkeitsdifferenzen bei Kanalverhältnis 1000

2.2 Entropie

Es gibt zwei Masse für die Entropie. Das übliche Mass ist die Shannonentropie H_s. Für eine binäre Zufallsvariable X mit Parameter p ist sie definiert als

$$h_s(p) := -p \log_2 p - (1-p) \log_2 (1-p). \tag{4}$$

Um die Shannonentropie von Bob zu berechnen, kann man direkt die Bitfehlerwahrscheinlichkeit β in die Formel 4 einsetzen. Die Shannonentropie von Eve hingegen muss gemittelt werden, da Eve für jede eintretende Situation eine andere Bitfehlerwahrscheinlichkeit hat auf ihrem Bit, wenn sie eine harte Entscheidung fällt. Ihre Shannonentropie beträgt somit

$$H_s(Eve) = \frac{1}{\epsilon^2 + (1-\epsilon)^2} \sum_{\omega=0}^{n} \binom{n}{\omega} \frac{1}{2} (p_\omega + p_{n-\omega}) h_s(\frac{p_\omega}{p_\omega + p_{n-\omega}}), \tag{5}$$

nämlich die Summe über die Wahrscheinlichkeiten, dass ein Wort mit Gewicht ω auftritt, multipliziert mit der Entropie in diesem Fall und das Ganze skaliert mit der Wahrscheinlichkeit, dass Bob akzeptiert.

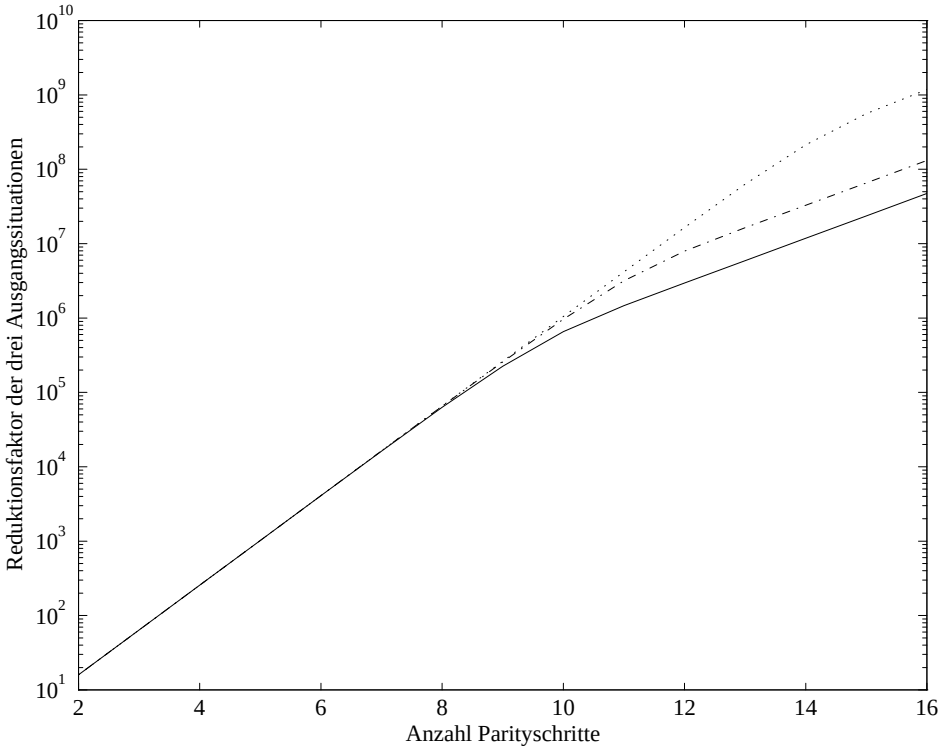

Abbildung 3: Reduktionsfaktoren der drei Ausgangssituationen $\epsilon_B = 0.48, 0.488$ und 0.496 bei Kanalverhältnis 1000

Die Abbildung 4 zeigt den Verlauf der Shannonentropie von Bob ausgezogen und von Eve gestrichelt für drei verschiedene Ausgangssituationen, wobei Bob einen 1000 mal schlechteren Kanal hat als Eve.

Die Entwicklung ist ähnlich wie bei den Wahrscheinlichkeiten in Abbildung 1.

Das andere Mass für die Entropie ist die Kollisionsentropie H_c, welche für das Universal Hashing [8] von Bedeutung ist. Sie ist für eine binäre Zufallsvariable X mit Parameter p definiert als

$$h_c(p) := -\log_2(p^2 + (1-p)^2). \tag{6}$$

Auch hier kann man für die Kollisionsentropie von Bob einfach β in die Formel 6 einsetzen, während bei Eve die Kollisionsentropie gemittelt werden muss. Die Formel ist analog der Formel für die Shannonentropie, nämlich

$$H_c(Eve) = \frac{1}{\epsilon^2 + (1-\epsilon)^2} \sum_{\omega=0}^{n} \binom{n}{\omega} \frac{1}{2}(p_\omega + p_{n-\omega}) h_c(\frac{p_\omega}{p_\omega + p_{n-\omega}}), \tag{7}$$

Die Abbildung 5 zeigt den Verlauf der Kollisionsentropie von Bob ausgezogen und

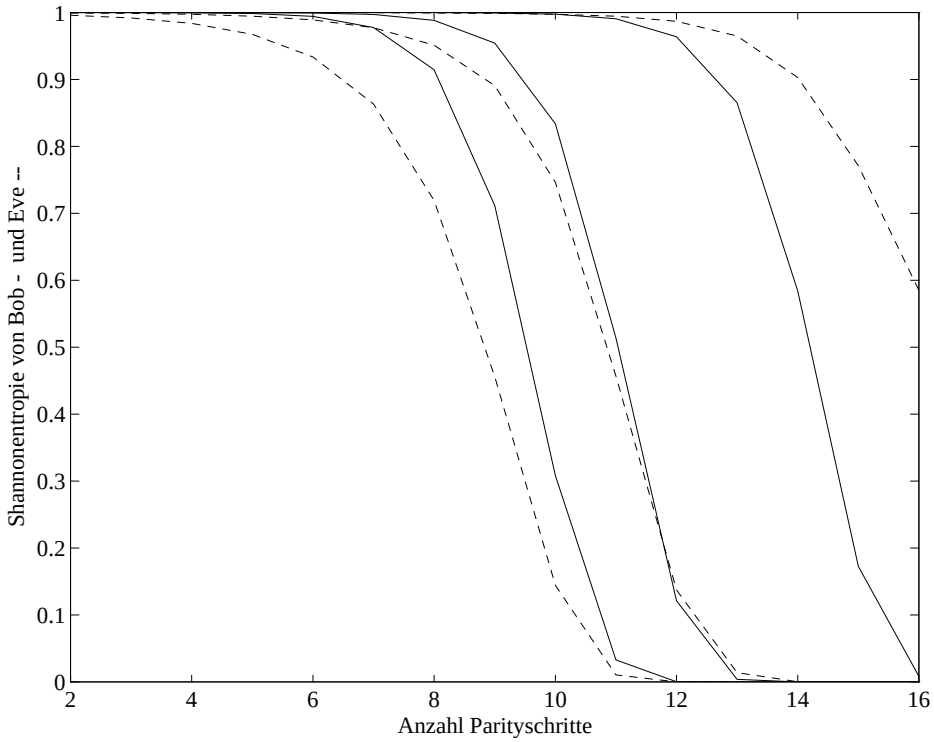

Abbildung 4: Shannonentropie von Bob ausgezogen und von Eve gestrichelt für die drei Ausgangssituationen $\epsilon_B = 0.48$,0.488, 0.496 bei Kanalverhältnis 1000

von Eve gestrichelt für drei verschiedene Ausgangssituationen, wobei Bob einen 1000 mal schlechteren Kanal hat als Eve.

Die Entwicklung ist auch hier ähnlich wie bei den Wahrscheinlichkeiten in Abbildung 1.

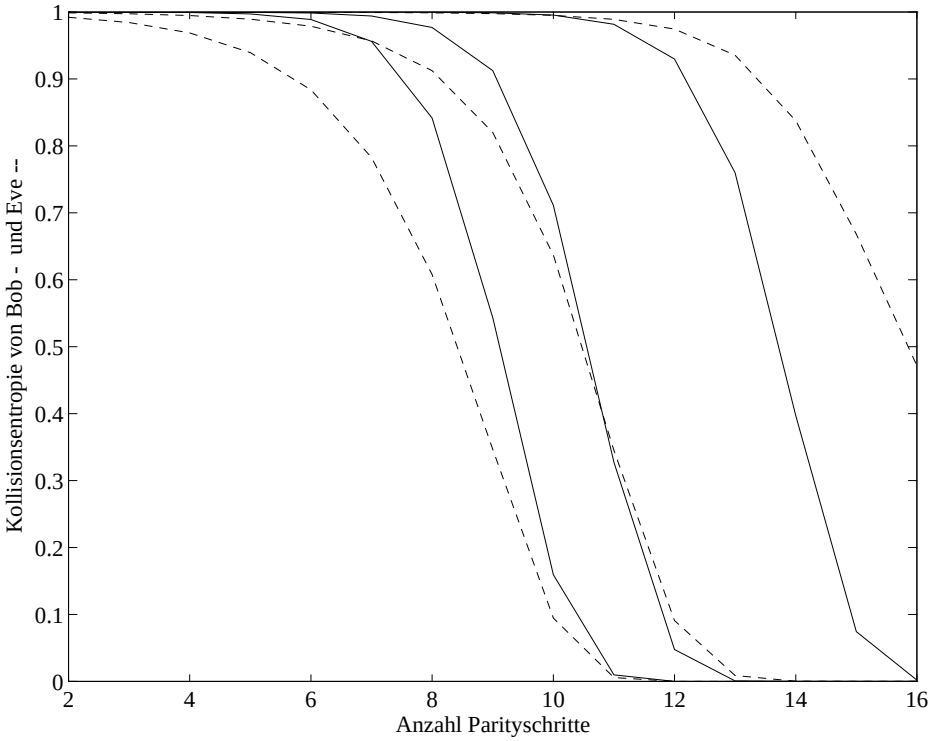

Abbildung 5: Kollisionsentropie von Bob ausgezogen und von Eve gestrichelt für die drei Ausgangssituationen $\epsilon_B = 0.48, 0.488, 0.496$ bei Kanalverhältnis 1000

3 Das Mass der Qualität

3.1 Definition

Wie in Abschnitt 2.1 angetönt, brauchen wir ein Mass für die Güte eines Protokolls der betrachteten Klasse. Ich definiere die folgenden drei Masse :

$$\begin{aligned} Q_s &:= r(H_s(Eve) - H_s(Bob)) \\ Q_c &:= r(H_c(Eve) - H_c(Bob)) \\ Q_o &:= r(2\gamma - H_s(Bob)), \end{aligned}$$

wobei r der Reduktionsfaktor ist (vergleiche dazu [7]). Das erste Mass Q_s nenne ich die Shannonqualität. Sie gibt an, wieviel Shannoninformation aus einem Bit sicher gewonnen werden kann. Die Shannonqualität ist wichtig für ein Theorem von Ueli Maurer, welches in [9] publiziert wird. Er zeigt darin, dass man, in Raten rechnend, diesen Vorteil vollständig ausnutzen kann.

Das zweite Mass Q_c ist die Kollisionsqualität. Sie zeigt, wieviel Kollisionsinformation aus einem Bit sicher erhalten werden kann. Die Kollisionsqualität ist relevant, wenn man konkret Protokolle angeben will, die es gestatten, aus dem Vorsprung einen sicheren Bitstring zu extrahieren (vergleiche dazu [8]).

Das dritte Mass Q_o heisst Orakelqualität und folgt einem interessanten Orakelargument, welches auch in [10] verwendet wird. Es beruht auf folgender Betrachtung: Wenn Eve eine Bitfehlerwahrscheinlichkeit von zum Beispiel 0.1 hat, dann bekommt sie im Durchschnitt jedes zehnte Bit falsch. Hätte nun Eve ein Orakel, welches in 80 Prozent der Fälle Eve das richtige Bit einfach überlässt und in den restlichen 20 Prozent ein Bit für Eve würfelt, so wäre Eve aus ihrer Sicht in der genau gleichen Situation, ihre Bits wären mit der Bitfehlerwahrscheinlichkeit 0.1 behaftet. Daraus folgt, dass die Unsicherheit von Eve über die Bits nicht grösser sein kann als zwei mal die Bitfehlerwahrscheinlichkeit. Soweit die Begründung für die Orakelqualität.

3.2 Auswertung

Die Abbildungen 6 bis 8 zeigen, dass für alle drei Masse nun ein Maximum zu finden ist, wobei die am Satellit einzustellende Bitfehlerwahrscheinlichkeit ϵ_A von Alice und ϵ_B von Bob und die Anzahl Parityschritte k als Parameter frei gewählt werden können. Wir setzen der Einfachheit halber $\epsilon_A = \epsilon_B$.

In der Abbildung 6 sieht man, dass mit 11 Parityschritten die höchste Shannonqualität erzielt wird für ein Kanalverhältnis von 1000 zwischen Bob und Eve. Mit einer anderen Anzahl Parityschritten ist der erreichbare Vorteil viel geringer. Es wird klar, dass, obwohl man mit 21 Parityschritten eine sehr grosse Bitfehlerwahrscheinlichkeitsdifferenz erreicht, dies mitnichten der optimalen Ausbeute entspricht.

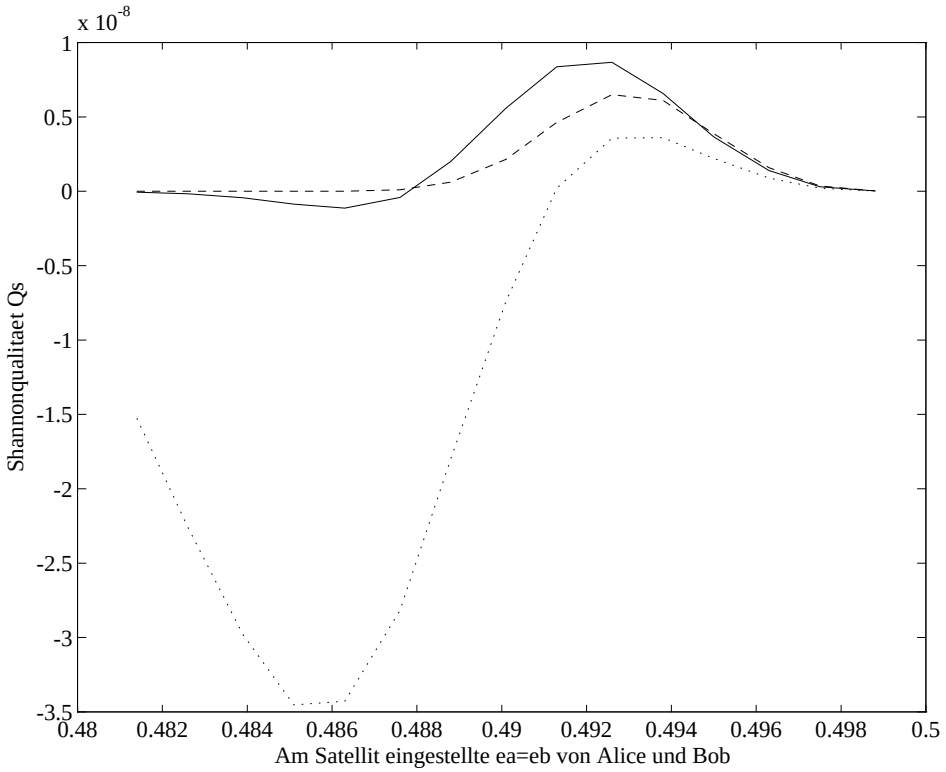

Abbildung 6: Sannonqualität für 10 Parityschritte gepunktet, für 11 ausgezogen und für 12 gestrichelt bei Kanalverhältnis 1000

Auch die Kollisionsqualität erreicht bei einem Kanalveräl tnis von 1000 für 11 Parityschritte ihren höchsten Wert. Die Startsituation ϵ_B ist aber leicht verschieden von der Startsituation bei der maximalen Shannonqualität.

Die Abbildung 8 macht deutlich, dass die Orakelqualität ein strengeres Mass ist als die Shannon- und Kollisionsqualität. Man erreicht ihr Maximum erst mit 12 Parityschritten und es liegt viel tiefer als das Maximum der beiden anderen Masse. Die Orakelqualität ist ein Mass auf der sicheren Seite, welches noch Reserven hat.

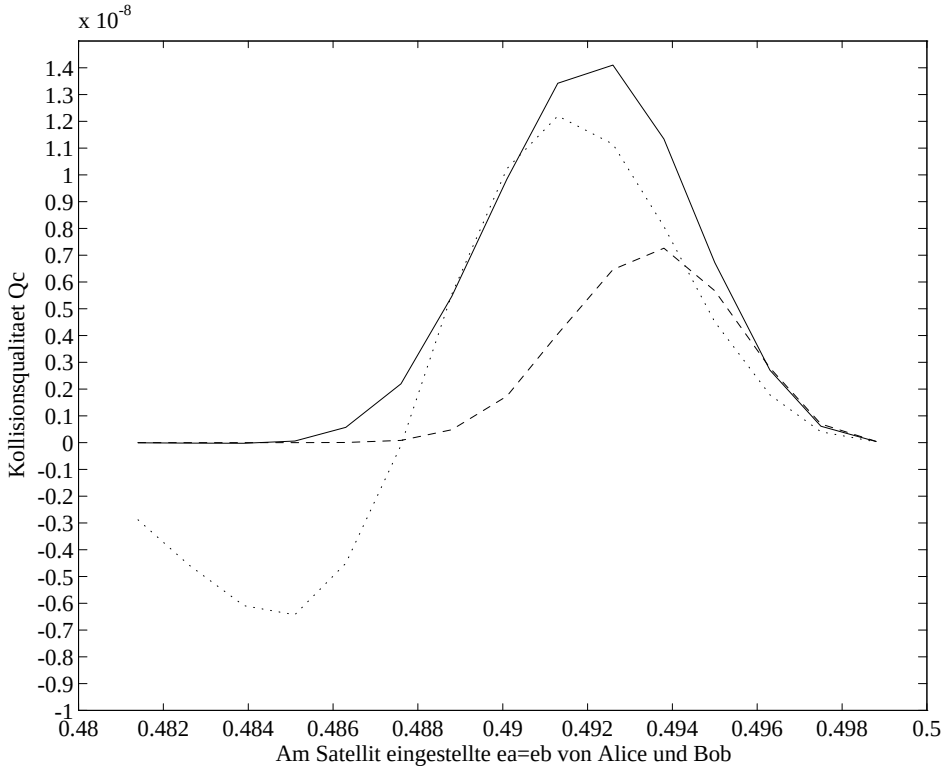

Abbildung 7: Kollisionsqualität für 10 Parityschritte gepunktet, für 11 ausgezogen und für 12 gestrichelt bei Kanalverhältnis 1000

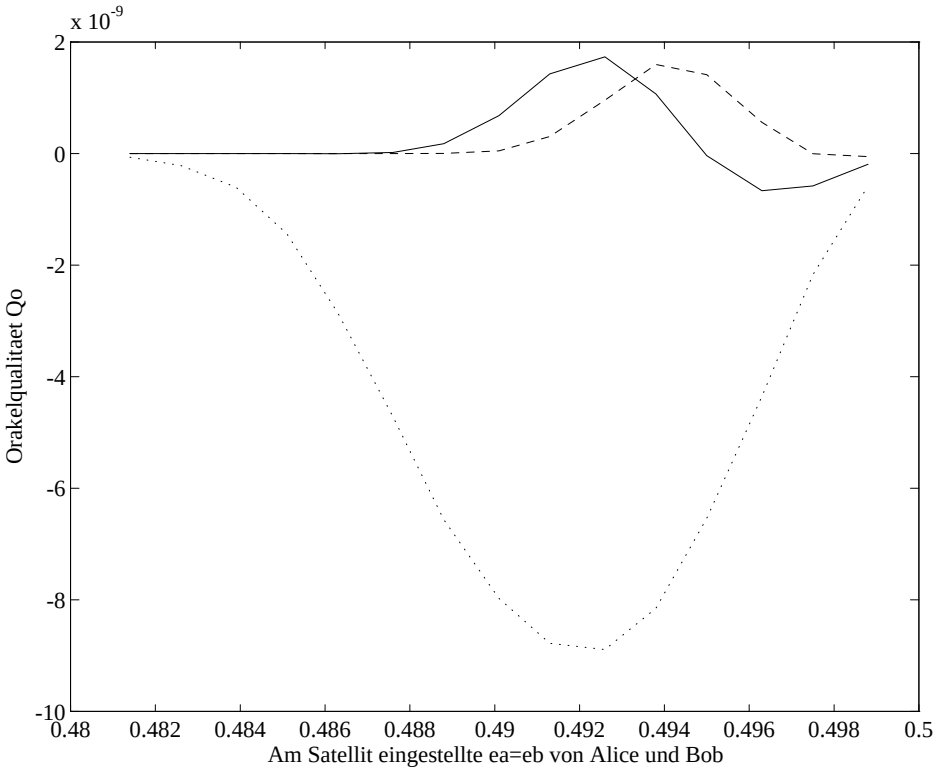

Abbildung 8: Orakelqualität für 11 Parityschritte gepunktet, für 12 ausgezogen und für 13 gestrichelt bei Kanalverhältnis 1000

3.3 Maximal erreichbarer Vorteil

Ich habe nun für eine Reihe konkreter Kanalverhältnisse zwischen Bob und Eve am Anfang die Shannonqualität optimiert. Die Resultate zeigt die Tabelle 1.

Kanalverhältnis	Parityschritte	$\epsilon_A = \epsilon_B$	maximales Q_s
1	2	0.0799	8.778E-3
5	4	0.3817	3.884E-4
10	6	0.4244	8.283E-5
50	8	0.4651	3.535E-6
100	9	0.4754	8.787E-7
500	11	0.4888	3.559E-8
1000	12	0.4921	8.891E-9
5000	14	0.4965	3.233E-10
10000	15	0.4975	8.082E-11
50000	18	0.4989	3.470E-12
100000	19	0.4992	8.674E-13

Tabelle 1: Maximal erreichbare Shannonqualität bei gegebenem Kanalverältnis

Man sieht, dass, je schlechter das Kanalverhältnis ist, desto näher bei 0.5 muss man mit ϵ_A von Alice und ϵ_B von Bob starten, um das Optimum zu erreichen. Interessanter ist aber folgender Zusammenhang: Wenn man das Kanalverhältnis verzehnfacht, so wird die Shannonqualität um ungefähr den Faktor hundert kleiner. Betrachten wir zum Beispiel das Kanalverhältnis 100. Die maximal erreichbare Shannonqualität beträgt $8.787E-7$. Bei zehn mal schlechterem Kanalverältnis, also 1000, beträgt sie $8.891E-9$, rund 100 mal weniger.

Die Abbildung 9 zeigt deutlich, dass bei logarithmischer Skalierung ein linearer Zusammenhang besteht.

Das bedeutet, dass man, um ein sicheres Bit für den Schlüssel zu gewinnen hundert mal mehr Bits braucht bei zehnmal schlechterem Kanal, vorausgesetzt, man kann den hier berechneten Vorteil vollständig ausnützen.

Das gilt auch für die Kollisionsqualität und für die Orakelqualität, wie die Tabellen 2 und 3 zeigen.

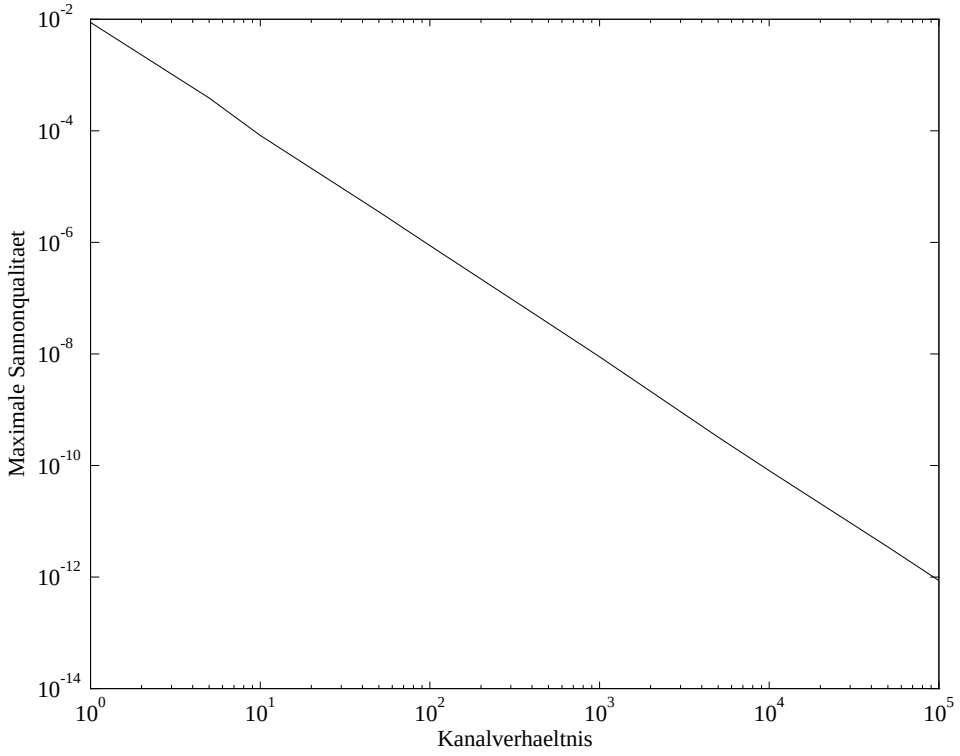

Abbildung 9: Maximal erreichbare Shannonqualität Q_s für verschiedene Kanalverhältnisse

Kanalverhältnis	Parityschritte	$\epsilon_A = \epsilon_B$	maximales Q_c
1	1	0.0799	4.713E-2
5	4	0.3783	7.236E-4
10	5	0.4155	1.704E-4
50	7	0.4613	6.428E-6
100	8	0.4727	1.593E-6
500	11	0.4889	5.711E-8
1000	12	0.4922	1.427E-8
5000	14	0.4963	6.380E-10
10000	15	0.4974	1.595E-10
50000	17	0.4988	6.379E-12
100000	18	0.4991	1.595E-12

Tabelle 2: Maximal erreichbare Kollisionsqualität bei gegebenem Kanalverältnis

Kanalverhältnis	Parityschritte	$\epsilon_A = \epsilon_B$	maximales Q_s
1	2	0.0799	5.769E-3
5	6	0.4085	8.492E-5
10	7	0.4358	2.033E-5
50	9	0.4688	8.277E-7
100	10	0.4780	2.054E-7
500	12	0.4892	7.000E-9
1000	13	0.4924	1.748E-9
5000	16	0.4972	7.686E-11
10000	17	0.4980	1.921E-11
50000	19	0.4990	8.171E-13
100000	20	0.4993	2.043E-13

Tabelle 3: Maximal erreichbare Orakelqualität bei gegebenem Kanalverältnis

4 Ausnützen des entstandenen Vorteils

Durch die in Abschnitt 2 beschriebenen Verfahren ist es in jedem Fall möglich, dass Alice und Bob sich einen Vorteil schaffen gegenüber Eve, was die Bitfehlerwahrscheinlichkeit betrifft. Um aus diesem Vorteil einen geheimen Schlüssel zu extrahieren, gibt es das Verfahren des Universal Hashing, welches in [8] genau beschrieben ist. Damit dieses Verfahren aber funktioniert, müssen die Bitstrings von Alice und Bob mit hoher Wahrscheinlichkeit übereinstimmen, was zum Beispiel durch zusätzliche Paritychecks erreicht werden kann. Diese geben aber wiederum Information für Eve. Eine obere Grenze für die Kollisionsinformation, welche beim Universal Hashing das entscheidende Mass ist, wird in diesem Kapitel für verschiedene Vorbedingungen vorgestellt.

4.1 Maximale Abnahme der Kollisionsentropie durch Paritychecks

In diesem Abschnitt zeige ich, dass durch einen Paritycheck über n unabhängige Bits die Kollisionsentropie um höchstens 2 abnehmen kann. Die Abnahme ist in den meisten Fällen wesentlich geringer und kann auch ganz verschwinden.

4.1.1 Bits gleichverteilt

Gegeben sei eine Verteilung über Wörter aus n zufälligen Bits, gleichverteilt mit der Wahrscheinlichkeit p. Dann ist die Kollisionswahrscheinlichkeit dieser Verteilung vor einem Paritycheck

$$p_c = \sum_{i=0}^{n} \binom{n}{i} \left(p^i(1-p)^{n-i}\right)^2 = p^2 + (1-p)^2$$

Nach einem Paritycheck muss man zwei Fälle unterscheiden, nämlich den Fall, in dem das Paritybit mit dem Parity von Eves Wort übereinstimmt, und denjenigen, in dem es nicht übereinstimmt. Im ersten Fall beträgt die Kollisionswahrscheinlichkeit nach dem Paritycheck

$$\begin{aligned} p_c^1 &= \frac{\sum_{i=0,gerade}^{n} \binom{n}{i} \left(p^i(1-p)^{n-i}\right)^2}{\left(\sum_{i=0,gerade}^{n} \binom{n}{i} \left(p^i(1-p)^{n-i}\right)\right)^2} \\ &= \frac{\frac{1}{2}\left((p^2+(1-p)^2)^n + (p^2-(1-p)^2)^n\right)}{\left(\frac{1}{2}\left((p+(1-p))^n + (p-(1-p))^n\right)\right)^2} \\ &= 2\,\frac{(p^2+(1-p)^2)^n + (2p-1)^n}{(1+(2p-1)^n)^2}. \end{aligned}$$

Satz 1. Durch einen Paritycheck nimmt in diesem Fall die Kollisionsetropie H_c um höchstens 2 ab.
Beweis. Sei $v = \frac{p_c^1}{p_c}$. Dann genügt es, zu zeigen, dass $v \leq 4$.

$$\begin{aligned} v &= 2 \ \frac{\left(p^2+(1-p)^2\right)^n + (2p-1)^n}{\left(1+(2p-1)^n\right)^2 \left(p^2+(1-p)^2\right)^n} \\ &= 2 \ \frac{1+\left(\frac{2p-1}{p^2+(1-p)^2}\right)^n}{\left(1+(2p-1)^n\right)^2}. \end{aligned}$$

Mit $0.5 \leq p \leq 1$ gilt $(2p-1)^n \geq 0$ und $(\frac{p^2-(1-p)^2}{p^2+(1-p)^2})^n \leq 1$. Also wird

$$v \leq 2 \ \frac{1+1}{(1+0)^2} = 4.$$

Damit ist gezeigt, dass die Abnahme der Kollisionsentropie in diesem Fall kleiner gleich 2 ist.

Die Abbildung 10 zeigt den Verlauf von v für $n = 5$, $n = 50$ und $n = 500$. Man sieht, dass für grosse n ein Paritycheck nur für p nahe bei 1 die Kollisionsentropie um mehr als 1 abnehmen lässt. Die Grenze 2 wird nur asymtotisch erreicht.

Im zweiten Fall, wenn der Paritycheck mit dem Parity von Eve nicht übereinstimmt, beträgt die Kollisionswahrscheinlichkeit nach dem Paritycheck

$$\begin{aligned} p_c^2 &= \frac{\sum_{i=0,ungerade}^{n} \binom{n}{i} \left(p^i(1-p)^{n-i}\right)^2}{\left(\sum_{i=0,ungerade}^{n} \binom{n}{i} \left(p^i(1-p)^{n-i}\right)\right)^2} \\ &= 2 \ \frac{\left(p^2+(1-p)^2\right)^n - (2p-1)^n}{\left(1-(2p-1)^n\right)^2}. \end{aligned}$$

Satz 2. Durch einen Paritycheck nimmt in diesem Fall die Kollisionsetropie H_c um höchstens 1 ab.
Beweis. Sei $v = \frac{p_c^2}{p_c}$. Dann genügt es, zu zeigen, dass $v \leq 2$.

$$v = 2 \ \frac{1-\left(\frac{2p-1}{p^2+(1-p)^2}\right)^n}{1-\left(2p-1\right)^n)^2}.$$

Lemma: Für $0.5 \leq p \leq 1$ gilt

$$1-\left(\frac{2p-1}{p^2+(1-p)^2}\right)^n \leq \left(1-(2p-1)^2\right)^2.$$

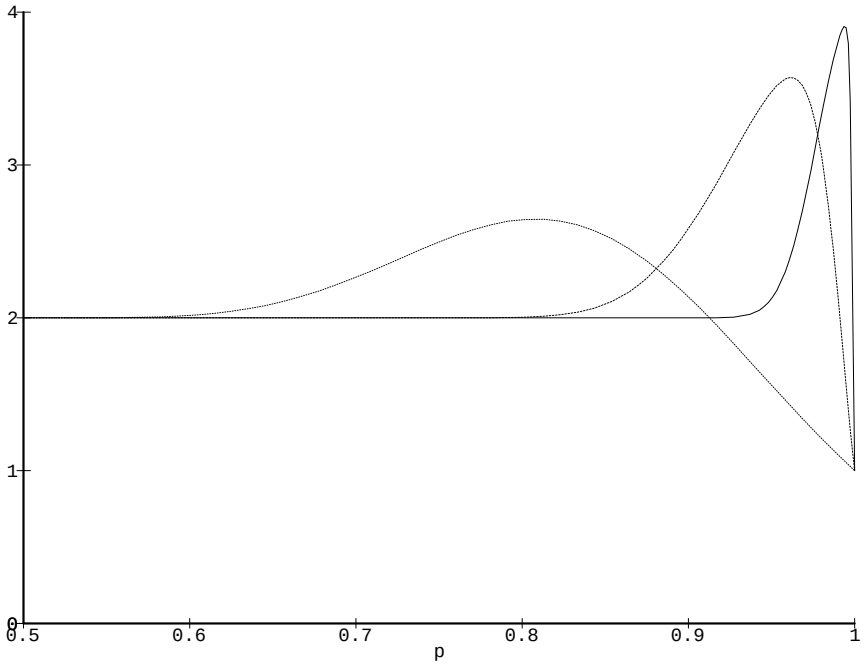

Abbildung 10: Verhältnis der Kollisionswahrscheinlichkeiten $v = p_c^1/p_c$

Beweis des Lemmas:

$$\begin{aligned} 1 - \left(\frac{2p-1}{p^2+(1-p)^2}\right)^n &\leq \left(1-2(2p-1)^2\right)^2 + (2p-1)^{2n} \\ -\left(\frac{1}{p^2+(1-p)^2}\right)^n &\leq -2 + (2p-1)^n \\ \frac{1}{(p^2+(1-p)^2)^n} + \left(p^2-(1-p)^2\right) &\geq 2 \end{aligned}$$

Um die letzte Ungleichung zu beweisen, suche ich das Minimum der Funktion

$$f = \frac{1}{(p^2+(1-p)^2)^n} + \left(p^2-(1-p)^2\right).$$

Ihre Ableitung beträgt

$$f' = -\frac{2n(2p-1)}{(p^2+(1-p)^2)^{n+1})} + 2n(2p-1)^{n-1}.$$

Man sieht leicht, dass $f'(p) = 0$ für $p = 0$, $p = \frac{1}{2}$ und $p = 1$. Wie die zweite Ableitung zeigt, ist bei $p = \frac{1}{2}$ ein lokales Maximum, wärend die anderen beiden Nullstellen Minima sind. Die Funktion f nimmt an den Minima gerade den Wert 2 an, womit das Lemma bewiesen ist.

Mit Hilfe des Lemmas folgt sofort

$$v \leq 2.$$

Damit ist gezeigt, dass die Abnahme der Kollisionsentropie in diesem Fall kleiner gleich 1 ist.

Die Abbildung 11 zeigt den Verlauf von v für $n = 5$, $n = 50$ und $n = 500$. Man sieht, dass für grosse n ein Paritycheck nur für p nahe bei 1 die Kollisionsentropie um weniger als 1 abnehmen lässt.

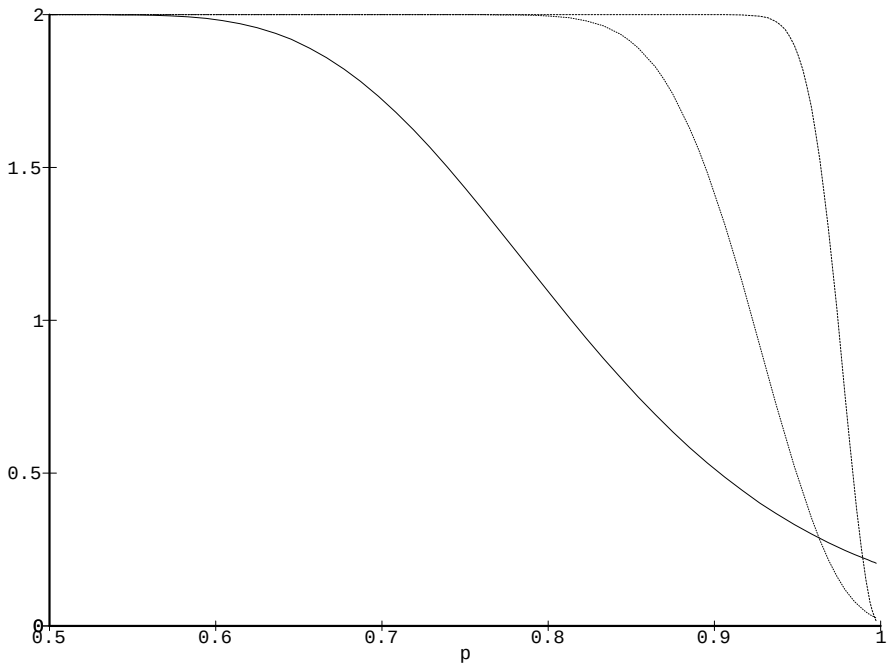

Abbildung 11: Verhältnis der Kollisionswahrscheinlichkeiten $v = p_c^2/p_c$

4.1.2 Bits beliebig verteilt

Gegeben sei eine Verteilung über Wörter aus n zufälligen Bits, beliebig verteilt mit der Wahrscheinlichkeit p_i. Dann ist die Kollisionswahrscheinlichkeit dieser Verteilung vor einem Paritycheck

$$p_c = \prod_{i=1}^{n} \left(p_i^2 + (1-p_i)^2 \right).$$

Nach einem Paritycheck unterscheidet man die gleichen Fälle wie in Abschnitt 4.1.1. Im ersten Fall beträgt die Kollisionswahrscheinlichkeit nach dem Paritycheck

$$p_c^1 = 2\ \frac{\prod_{i=1}^{n} \left(p_i^2 + (1-p_i)^2 \right) + \prod_{i=1}^{n} (2p_i - 1)}{\left(1 + \prod_{i=1}^{n} (2p_i - 1) \right)^2}.$$

Satz 3. Durch einen Paritycheck nimmt in diesem Fall die Kollisionsetropie H_c um höchstens 2 ab.
Beweis. Der Beweis erfolgt wie in Satz 1.

Im zweiten Fall beträgt die Kollisionswahrscheinlichkeit nach dem Paritycheck

$$p_c^1 = 2\ \frac{\prod_{i=1}^{n} \left(p_i^2 + (1-p_i)^2 \right) - \prod_{i=1}^{n} (2p_i - 1)}{\left(1 - \prod_{i=1}^{n} (2p_i - 1) \right)^2}.$$

Satz 4. Durch einen Paritycheck nimmt in diesem Fall die Kollisionsetropie H_c um höchstens 1 ab.
Beweis. Der Beweis erfolgt wie in Satz 2.

4.1.3 Überlappende Paritychecks

Die bisher betrachteten Fälle schliessen sich überlappende Paritychecks aus. Ich habe für den Fall von zwei Paritychecks über je n Bits, wobei ein gemeinsames Bit in beide Paritychecks einbezogen wurde, folgenden Satz gefunden:
Satz 5. Falls sich zwei Paritychecks um ein Bit überlappen, so nimmt die Kollisionsentropie H_c um höchstens 4 ab.
Beweis. Die Kollisionsentropie vor den Paritychecks beträgt

$$p_c = \left(p^2 + (1-p)^2 \right)^{2n-1}$$

Nach den Paritychecks beträgt sie mit $\epsilon = (1-p)$

$$p_c^{'} = \frac{p^2 \left(\frac{1}{2} \left(p^2 + \epsilon^2 \right)^{n-1} + \left(p^2 - \epsilon^2 \right)^{n-1} \right)^2 + \epsilon^2 \left(\frac{1}{2} \left(p^2 + \epsilon^2 \right)^{n-1} - \left(p^2 - \epsilon^2 \right)^{n-1} \right)^2}{\left(p \left(\frac{1}{2} (p+\epsilon)^{n-1} + (p-\epsilon)^{n-1} \right)^2 + \epsilon \left(\frac{1}{2} (p+\epsilon)^{n-1} - (p-\epsilon)^{n-1} \right)^2 \right)^2}$$

was nach einer ziemlich aufwendigen Rechnung und Abschätzung zeigt, dass $v = \frac{p_c^{'}}{p_c} \leq 16$ ist und somit $\Delta H_c \leq 4$, womit der Satz bewiesen ist.

4.1.4 Beliebige Paritychecks

Vermutung. Für alle Codes, welche als Paritycheckcodes aufgefasst werden können, gilt, dass die Kollisionsentropie H_c um höchstens $2k$ abnimmt, wenn der Code k Paritybits verwendet. Diese Schranke ist dicht.

Beweis. Der Beweis erfolgt durch Induktion, ich habe ihn aber nicht ausgeführt.

4.2 Abschätzung bei um maximal μ gestörter Gleichverteilung

Gegeben sei eine Menge S von gleichverteilten Wörtern und ein durch einen Code bestimmter Subset $S^{'}$. Die Kollisionswahrscheinlichkeit p_c in S beträgt

$$p_c = \sum_{s \epsilon S} p_s^2 = \sum_{s \epsilon S} \frac{1}{\mid S \mid^2} = \frac{1}{|S|}$$

Die Kollisionswahrscheinlichkeit $p_c^{'}$ in $S^{'}$ beträgt

$$\begin{aligned} p_c^{'} &= \frac{\sum_{s \epsilon S^{'}} p_s^2}{\left(\sum_{s \epsilon S^{'}} p_s \right)^2} \\ &= \frac{\sum_{s \epsilon S^{'}} \frac{1}{|S|^2}}{\left(\sum_{s \epsilon S^{'}} \frac{1}{|S|} \right)^2} \\ &= \frac{1}{|S^{'}|} \end{aligned}$$

Die Kollisionswahrscheinlichkeit nimmt also um den Faktor

$$v = \frac{|S|}{|S^{'}|} \tag{8}$$

zu.

Wenn man nun zulässt, dass die Gleichverteilung um maximal μ gestört wird, das heisst

$$p_s = \frac{1}{|S|} + \epsilon_s,$$

wobei $|\epsilon_s| \leq \mu$, dann wird die Kollisionswahrscheinlichkeit p_c auf S

$$p_c = \sum_{s \epsilon S} p_s^2 = \sum_{s \epsilon S} \left(\frac{1}{|S|} + \epsilon_s \right)^2 = \frac{1}{|S|} + 2 \frac{1}{|S|^2} \sum_{s \epsilon S} \epsilon_s + \sum_{s \epsilon S} \epsilon_s^2$$

Die Kollisionswahrscheinlichkeit auf $p_c^{'}$ auf $S^{'}$ beträgt nun

$$
\begin{aligned}
p_c^{'} &= \frac{\sum_{s\epsilon S'} p_s^2}{\left(\sum_{s\epsilon S'} p_s\right)^2} \\
&= \frac{\sum_{s\epsilon S'} \left(\frac{1}{|S|} + \epsilon_s\right)^2}{\left(\sum_{s\epsilon S'} \frac{1}{|S|} + \epsilon_s\right)^2} \\
&= \frac{\frac{|S^{'}|}{|S|^2} + \frac{2}{|S|^2}\sum_{s\epsilon S'} \epsilon_s + \sum_{s\epsilon S'} \epsilon_s^2}{\frac{|S'|^2}{|S|^2} + 2\frac{|S'|}{|S|^2}\sum_{s\epsilon S'} \epsilon_s + \left(\sum_{s\epsilon S'} \epsilon_s\right)^2}
\end{aligned}
$$

Nun kann man wieder eine obere Schranke für den Faktor $v = \frac{p_c^{'}}{p_c}$ abschätzen

$$
\begin{aligned}
v &= \frac{\frac{|S^{'}|}{|S|^2} + \frac{2}{|S|^2}\sum_{s\epsilon S'} \epsilon_s + \sum_{s\epsilon S'} \epsilon_s^2}{\left(\frac{|S'|^2}{|S|^2} + 2\frac{|S'|}{|S|^2}\sum_{s\epsilon S'} \epsilon_s + \left(\sum_{s\epsilon S'} \epsilon_s\right)^2\right)\left(\frac{1}{|S|} + \frac{2}{|S|^2}\sum_{s\epsilon S} \epsilon_s + \sum_{s\epsilon S} \epsilon_s^2\right)} \\
&\leq \frac{\frac{|S^{'}|}{|S|^2} + 2\frac{|S^{'}|}{|S|}\mu + |S^{'}|\mu^2}{\left(\frac{|S'|^2}{|S|^2} - 2\frac{|S'|^2}{|S|}\mu\right)\left(\frac{1}{|S|} - 2\mu\right)} \\
&= \frac{|S|\left(\frac{1}{|S|} + \mu\right)^2}{|S'|\left(\frac{1}{|S|} - 2\mu\right)^2}
\end{aligned}
$$

Damit ist gezeigt, dass das Verhältnis v der Kollisionswahrscheinlichkeiten bei um maximal μ gestörter Gleichverteilung die Bedingung

$$
v \leq \frac{|S|\left(\frac{1}{|S|} + \mu\right)^2}{|S'|\left(\frac{1}{|S|} - 2\mu\right)^2} \tag{9}
$$

erfüllt.

Sei zum Beispiel $\mu = 0$, also eine perfekte Gleichverteilung, dann ist das Verhältnis

$$
v \leq \frac{|S|}{|S'|},
$$

wobei die Gleichheit in Formel 8 gezeigt wurde. Wenn zum Beispiel bei n-Bitwörtern k Paritychecks gemacht würden, so wäre das Verhältnis

$$v \leq \frac{2^n}{2^{n-k}} = 2^k.$$

Das bedeutet, dass die Kollisionsentropie H_c in diesem Fall höchstens um k abnimmt.

Wählt man die maximale Störung $\mu = \frac{1}{|S|}$, das heisst, man erlaubt Wahrscheinlichkeiten $0 \leq p_s \leq \frac{1}{2|S|}$, dann wird das Verhältnis

$$v \leq 4\frac{|S|}{|S'|},$$

ist also um höchstens einen konstanten Faktor 4 grösser als bei exakter Gleichverteilung. Allgemein kann man für eine maximale Störung von $\mu = \delta\frac{1}{|S|}$ sagen, dass das Verhältnis v durch

$$v \leq \frac{|S|}{|S'|}\frac{(1+\delta)^2}{(1-2\delta)^2} \tag{10}$$

beschränkt ist.

Literatur

[1] U.M. Maurer, Secret key agreement by public discussion from common information, to appear in *IEEE Transactions on Information Theory.*

[2] U.M. Maurer, Protocols for secret key agrreement by public discussion from common information, to appear in *Advances in Cryptology -CRYPTO '92*, Lecture Notes in Computer Science, Berlin: Springer-Verlag.

[3] C.H.Bennett, G. Brassard and J.-M. Robert, "Privacy amplification by public discussion", *SIAM Jurnal on Computing*, Vol. 17, No. 2,pp.210-229,1988.

[4] R.G. Gallager, *Information Theory and Reliable Communication*, New York: John Wiley & Sons, 1968.

[5] R.E.Blahut, *Principles and Practice of Information Theory*, Reading, MA: Addison-Wesley, 1987.

[6] C.E. Shannon, Communication theory of secrecy systems, *Bell System Technical Journal*, Vol. 28, Oct. 1949, pp. 656-715.

[7] M.Gander, Analyse beweisbar sicherer Schlüsselgenerierungsprotokolle, *Diplomarbeit am Institut für theoretische Informatik an der ETH Zürich.*

[8] J.L.Carter, M.N.Wegman, *Universal classes of hash functions*, J. Comput. System Sci.,18 (1979), pp. 143-154.

[9] Ueli M. Maurer, *The Strong Secret Key Rate of Discrete Random Triples*, to appear.

[10] Charles H. Bennet *Experimental Quantum Cryptography*, Journal of Cryptography (1992) 5: 3-28.

[11] William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery, *Numerical Recipies in C*, Cambridge University Press 1992.

A Programmbeschreibungen

Das Programm QA besteht aus drei Modulen. Der unterste Modul Math enthält Standardalgorithmen aus der Numerik. Der Modul Protocol enthält den numerisch interesannten Algorithmus zur Berechnung der Bitfehlerwahrscheinlichkeit und der Entropien und der oberste Modul QA stellt ein einfaches Menu zur Verfügung zur Berechnung der vorgestellten Resultate.

A.1 QA

Ohne Programmieraufwand kann man folgende Resultate direkt mit QA berechnen :

1. Die Bitfehlerwahrscheinlichkeiten β und γ, sowie die Shannonentropie H_s, die Kollisionsentropie H_c und den Reduktionsfaktor r für eine konkrete Startsituation aus ϵ_A von Alice, ϵ_B von Bob und ϵ_E von Eve und ein Parityprotokoll mit k Schritten.

2. Eine Tabelle der Bitfehlerwahrscheinlichkeiten β und γ und des Reduktionsfaktors r für ein gegebenes Kanalverhältnis und ein Parityprotokoll mit k Schritten. Die Tabelle enthält dann die Resultate für 30 äquidistant verteilte Anfangssituationen $\epsilon_A = \epsilon_B$. Die Anzahl der Einträge kann mit der Konstanten $TableLength$ im Programm bestimmt werden.

3. Eine Tabelle der Shannonentropie H_s und der Kollisionsentropie H_c für ein gegebenes Kanalverhältnis und ein Parityprotokoll mit k Schritten. Die Tabelle enthält auch 30 Einträge wie in unter Punkt 2.

4. Eine Tabelle mit der Shannonqualität Q_s, der Kollisionsqualität Q_c und der Orakelqualität für ein gegebenes Kanalverhältnis und ein Parityprotokoll mit k Schritten. Die Tabelle enthält 30 Einträge.

5. Eine Tabelle mit der maximal erreichbaren Bitfehlerwahrscheinlichkeitsdifferenz $\gamma - \beta$ für ein Intervall von Parityprotokollen mit $k1$ bis $k2$ Schritten und ein gegebenes Kanalverhältnis.

6. Eine Tabelle mit der maximal erreichbaren Shannonentropiedifferenz, das heisst $H_s(Eve) - H_s(Bob)$ für ein Intervall von Parityprotokollen mit $k1$ bis $k2$ Schritten und ein gegebenes Kanalverhältnis.

7. Eine Tabelle mit der maximal erreichbaren Kollisionsentropiedifferenz, also $H_c(Eve) - H_c(Bob)$ für ein Intervall von Parityprotokollen mit $k1$ bis $k2$ Schritten und ein gegebenes Kanalverhältnis.

8. Eine Tabelle mit der maximal erreichbaren Shannonqualität Q_s für ein Intervall von Parityprotokollen mit $k1$ bis $k2$ Schritten und ein gegebenes Kanalverhältnis.

9. Eine Tabelle mit der maximal erreichbaren Kollisionsqualität Q_c für ein Intervall von Parityprotokollen mit $k1$ bis $k2$ Schritten und ein gegebenes Kanalverhältnis.

10. Eine Tabelle mit der maximal erreichbaren Orakelqualität Q_o für ein Intervall von Parityprotokollen mit $k1$ bis $k2$ Schritten und ein gegebenes Kanalverhältnis.

Es ist aber durch einfach Programmänderungen in QA mögliche, weitere gewünschte Resultate zu berechnen.

A.2 Protocol

Stellt die eigentlichen Rechnungsprozeduren zur Verfügung:

1. Die Prozeduren Hs und Hc, welche für eine binäre Zufallsvariable X mit Parameter p die Shannon- und Kollisionsentropie berechnen.

2. Die Prozeduren *Getee* und *Geteb*, die bei gegebenem ϵ_B und Kanalverhältnis v ϵ_E berechnen und umgekehrt.

3. Die Hauptprozedur *Parity*, welche für eine gegeben Ausgangssituation ϵ_A, ϵ_B und ϵ_E und ein Parityprotokoll aus k Schritten β nach der Formel 1, *gamma* nach der Formel 3, die Shannonentropie von Eve nach der Formel 5, die Kollisionsentropie von Eve nach der Formel7 und den Reduktionsfaktor r nach der Formel 12 aus der Diplomarbeit [7] berechnet.

A.3 Math

Stellt eine Prozedur Power zum Potenzenrechnen, einen Bisektionsalgorithmus und einen Algorithmus zum Maximieren von Funktionen aus [11] zur Verfügung.

B Programme

B.1 QA

```
MODULE PTable;

  FROM InOut IMPORT Read,ReadInt,WriteString,WriteLn,WriteInt;
  FROM LongRealIO IMPORT ReadLongReal,WriteLongReal;
  FROM Math IMPORT Golden,Function;
  FROM Protocol IMPORT Parity,Geteb,Getee,Hs,Hc;

  CONST
    Fields=11;
    Digits=-3;
    TableLength=30;
    Tol=0.0001;         (* Toleranz bei der Maximierung *)

  VAR
    ch:CHAR;
    done:BOOLEAN;
    v:LONGREAL;         (* Globale Variable fuer Kanalverhaeltnis !      *)
    k:LONGINT;          (* Globale Variable fuer Anzahl Parityschritte ! *)
    n:INTEGER;

  PROCEDURE fp(eb:LONGREAL):LONGREAL;
    VAR
      beta,gamma,hs,hc,r:LONGREAL;
  BEGIN
    Parity(eb,eb,Getee(eb,v),k,beta,gamma,hs,hc,r);
    RETURN gamma-beta;
  END fp;

  PROCEDURE fhs(eb:LONGREAL):LONGREAL;
    VAR
      beta,gamma,hs,hc,r:LONGREAL;
  BEGIN
    Parity(eb,eb,Getee(eb,v),k,beta,gamma,hs,hc,r);
    RETURN hs-Hs(beta);
  END fhs;

  PROCEDURE fhc(eb:LONGREAL):LONGREAL;
    VAR
      beta,gamma,hs,hc,r:LONGREAL;
  BEGIN
    Parity(eb,eb,Getee(eb,v),k,beta,gamma,hs,hc,r);
    RETURN hc-Hc(beta);
  END fhc;

  PROCEDURE fqs(eb:LONGREAL):LONGREAL;
    VAR
      beta,gamma,hs,hc,r:LONGREAL;
```

```
BEGIN
  Parity(eb,eb,Getee(eb,v),k,beta,gamma,hs,hc,r);
  RETURN r*(hs-Hs(beta));
END fqs;

PROCEDURE fqc(eb:LONGREAL):LONGREAL;
  VAR
    beta,gamma,hs,hc,r:LONGREAL;
BEGIN
  Parity(eb,eb,Getee(eb,v),k,beta,gamma,hs,hc,r);
  RETURN r*(hc-Hc(beta));
END fqc;

PROCEDURE fqo(eb:LONGREAL):LONGREAL;
  VAR
    beta,gamma,hs,hc,r:LONGREAL;
BEGIN
  Parity(eb,eb,Getee(eb,v),k,beta,gamma,hs,hc,r);
  RETURN r*(2.0*gamma-Hs(beta));
END fqo;

PROCEDURE One;
  VAR
    ea,eb,ee,beta,gamma,hs,hc,r:LONGREAL;
BEGIN
  WriteString("Geben sie die Bitfehlerwahrscheinlichkeiten ein:");
  WriteLn;
  WriteLn;
  WriteString("Alice:  ea = ");
  ReadLongReal(ea,done);
  Read(ch);
  WriteString("Bob  :  eb = ");
  ReadLongReal(eb,done);
  Read(ch);
  WriteString("Eve  :  ee = ");
  ReadLongReal(ee,done);
  Read(ch);
  WriteLn;
  WriteString("Anzahl Parityschritte = ");
  ReadInt(k);
  WriteLn;
  WriteLn;
  Parity(ea,eb,ee,k,beta,gamma,hs,hc,r);
  WriteString("                                Bob        Eve");
  WriteLn;
  WriteLn;
  WriteString("Bitfehlerwahrscheinlichkeit:");
  WriteLongReal(beta,Fields,Digits);
  WriteLongReal(gamma,Fields,Digits);
  WriteLn;
  WriteLn;
```

```
  WriteString("Shannonentropie            :");
  WriteLongReal(Hs(beta),Fields,Digits);
  WriteLongReal(hs,Fields,Digits);
  WriteLn;
  WriteLn;
  WriteString("Kollisionsentropie         :");
  WriteLongReal(Hc(beta),Fields,Digits);
  WriteLongReal(hc,Fields,Digits);
  WriteLn;
  WriteLn;
  WriteLn;
  WriteString("Reduktionsfaktor           :");
  WriteLongReal(r,Fields,Digits);
END One;

PROCEDURE Table1;
  VAR
    eb,deb,ee,beta,gamma,hs,hc,r:LONGREAL;
BEGIN
  WriteString("Kanalverhaeltnis = ");
  ReadLongReal(v,done);
  Read(ch);
  WriteString("Anzahl Parityschritte  = ");
  ReadInt(k);
  WriteLn;
  WriteLn;
  ee:=1.0E-10;
  eb:=Geteb(ee,v);
  deb:=(0.5-eb)/FLOAT(TableLength);
  WriteString("         Bitfehlerwahrscheinlichkeiten");
  WriteLn;
  WriteString("         vorher               nachher");
  WriteLn;
  WriteString("     Bob        Eve        Bob        Eve      Red.Faktor");
  WriteLn;
  WriteLn;
  WHILE eb<0.5 DO
    Parity(eb,eb,ee,k,beta,gamma,hs,hc,r);
    WriteLongReal(eb,Fields,Digits);
    WriteLongReal(ee,Fields,Digits);
    WriteLongReal(beta,Fields,Digits);
    WriteLongReal(gamma,Fields,Digits);
    WriteLongReal(r,Fields,Digits);
    WriteLn;
    eb:=eb+deb;
    ee:=Getee(eb,v);
  END;
END Table1;

PROCEDURE Table2;
  VAR
```

```
    eb,deb,ee,beta,gamma,hs,hc,r:LONGREAL;
BEGIN
  WriteString("Kanalverhaeltnis = ");
  ReadLongReal(v,done);
  Read(ch);
  WriteString("Anzahl Parityschritte  = ");
  ReadInt(k);
  WriteLn;
  WriteLn;
  ee:=1.0E-10;
  eb:=Geteb(ee,v);
  deb:=(0.5-eb)/FLOAT(TableLength);
  WriteString(" BitFehlerWS   Shannonentropie      Kollisionsentropie");
  WriteLn;
  WriteString("    vorher         nachher              nachher");
  WriteLn;
  WriteString("     Bob        Bob        Eve        Bob        Eve");
  WriteLn;
  WriteLn;
  WHILE eb<0.5 DO
    Parity(eb,eb,ee,k,beta,gamma,hs,hc,r);
    WriteLongReal(eb,Fields,Digits);
    WriteLongReal(Hs(beta),Fields,Digits);
    WriteLongReal(hs,Fields,Digits);
    WriteLongReal(Hc(beta),Fields,Digits);
    WriteLongReal(hc,Fields,Digits);
    WriteLn;
    eb:=eb+deb;
    ee:=Getee(eb,v);
  END;
END Table2;

PROCEDURE Table3;
  VAR
    eb,deb,ee,beta,gamma,hs,hc,r:LONGREAL;
BEGIN
  WriteString("Kanalverhaeltnis = ");
  ReadLongReal(v,done);
  Read(ch);
  WriteString("Anzahl Parityschritte  = ");
  ReadInt(k);
  WriteLn;
  WriteLn;
  ee:=1.0E-10;
  eb:=Geteb(ee,v);
  deb:=(0.5-eb)/FLOAT(TableLength);
  WriteString("  BitFehlerWS");
  WriteLn;
  WriteString("    vorher");
  WriteLn;
  WriteString("     Bob    Hs-Quality Hc-Quality Orakel-Quality");
```

```
    WriteLn;
    WriteLn;
    WHILE eb<0.5 DO
      Parity(eb,eb,ee,k,beta,gamma,hs,hc,r);
      WriteLongReal(eb,Fields,Digits);
      WriteLongReal(r*(hs-Hs(beta)),Fields,Digits);
      WriteLongReal(r*(hc-Hc(beta)),Fields,Digits);
      WriteLongReal(r*(2.0*gamma-Hs(beta)),Fields,Digits);
      WriteLn;
      eb:=eb+deb;
      ee:=Getee(eb,v);
    END;
  END Table3;

  PROCEDURE Maximize(s:ARRAY OF CHAR;f:Function);
    VAR
      fmax,xmax:LONGREAL;
      k1,k2:LONGINT;
  BEGIN
    WriteString("Kanalverhaeltnis = ");
    ReadLongReal(v,done);
    Read(ch);
    WriteString("Minimale Anzahl Parityschritte  = ");
    ReadInt(k1);
    WriteString("Maximale Anzahl Parityschritte  = ");
    ReadInt(k2);
    WriteLn;
    WriteLn;
    WriteString("Anzahl");
    WriteLn;
    WriteString("Parity-");
    WriteLn;
    WriteString("schritte   eb      ");
    WriteString(s);
    WriteLn;
    WriteLn;
    FOR k:=k1 TO k2 DO
      fmax:=Golden(Geteb(1.0E-10,v),0.5,Tol,f,xmax);
      WriteInt(k,5);
      WriteLongReal(xmax,Fields,Digits);
      WriteLongReal(fmax,Fields,Digits);
      WriteLn;
    END;
  END Maximize;

BEGIN
  WriteLn;
  WriteLn;
  WriteLn;
  WriteString("Qualitaetsanalyse von Parity-Protokollen");
  WriteLn;
```

```
WriteString("----------------------------------------");
WriteLn;
WriteLn;
WriteLn;
WriteString("1 => Berechne Bitfehlerwahrscheinlichkeiten, Entropien und");
WriteLn;
WriteString("     Reduktionsfaktor fuer eine gegebene Ausgangslage ea,eb");
WriteLn;
WriteString("     und ee und eine feste Anzahl Parityschritte.");
WriteLn;
WriteLn;
WriteString("2 => Berechne Tabelle der Bitfehlerwahrscheinlichkeiten p");
WriteLn;
WriteString("     und des Reduktionsfaktors r fuer ein gegebenes Kanalver-");
WriteLn;
WriteString("     haeltnis und fuer eine feste Anzahl Parityschritte.");
WriteLn;
WriteLn;
WriteString("3 => Berechne Tabelle der Shannonentropie Hs und der");
WriteLn;
WriteString("     der Kollisionsentropie Hc fuer eine feste Anzahl");
WriteLn;
WriteString("     Parityschritte.");
WriteLn;
WriteLn;
WriteString("4 => Berechne Tabelle der Groessen");
WriteLn;
WriteString("       Qs := Reduktionsfaktor*(Hs(Eve)-Hs(Bob))");
WriteLn;
WriteString("       Qc := Reduktionsfaktor*(Hc(Eve)-Hc(Bob))");
WriteLn;
WriteString("       Qo := Reduktionsfaktor*(2*p(Eve)-Hs(Bob))");
WriteLn;
WriteString("     fuer ein gegebenes Kanalverhaeltnis und");
WriteLn;
WriteString("     fuer eine feste Anzahl Parityschritte.");
WriteLn;
WriteLn;
WriteString("5 => Maximiere dp  = Bitfehlerwahrscheinlichkeitsdifferenz");
WriteLn;
WriteString("6 => Maximiere dHs = Hs(Eve)-Hs(Bob)");
WriteLn;
WriteString("7 => Maximiere dHc = Hc(Eve)-Hc(Bob)");
WriteLn;
WriteString("8 => Maximiere Qs  = Reduktionsfaktor*(Hs(Eve)-Hs(Bob))");
WriteLn;
WriteString("9 => Maximiere Qc  = Reduktionsfaktor*(Hc(Eve)-Hc(Bob))");
WriteLn;
WriteString("10=> Maximiere Qo  = Reduktionsfaktor*(2*p(Eve)-Hs(Bob))");
WriteLn;
WriteLn;
```

```
  WriteString("     fuer ein gegebenes Kanalverhaeltnis und");
  WriteLn;
  WriteString("     fuer einen Bereich von Parityschritten.");
  WriteLn;
  WriteLn;
  WriteString("Bitte waehlen sie eine Ziffer von 1 bis 10 : ");
  ReadInt(n);
  WriteLn;
  WriteLn;
  CASE n OF
    1: One;
  | 2: Table1;
  | 3: Table2;
  | 4: Table3;
  | 5: Maximize("max dp",fp);
  | 6: Maximize("max dHs",fhs);
  | 7: Maximize("max dHc",fhc);
  | 8: Maximize("max Qs",fqs);
  | 9: Maximize("max Qc",fqc);
  |10: Maximize("max Qo",fqo);
  END;
  WriteLn;
  WriteLn;
  WriteLn;
END PTable.
```

B.2 Protocol

```
DEFINITION MODULE Protocol;

  PROCEDURE Hs(p:LONGREAL):LONGREAL;
  PROCEDURE Hc(p:LONGREAL):LONGREAL;
  PROCEDURE Geteb(ee,factor:LONGREAL):LONGREAL;
  PROCEDURE Getee(eb,factor:LONGREAL):LONGREAL;
  PROCEDURE Parity(ea,eb,ee:LONGREAL;k:LONGINT;VAR b,g,hs,hc,r:LONGREAL);

END Protocol.


IMPLEMENTATION MODULE Protocol;

  FROM LongMathLib IMPORT Ln;
  FROM Math IMPORT Power, Bisect, Golden;

  VAR
    expo:LONGINT;
    v,base:LONGREAL;
    hseb,hsee:LONGREAL;

  PROCEDURE Hs(p:LONGREAL):LONGREAL;
  BEGIN
    RETURN (-p*Ln(p)-(1.0-p)*Ln(1.0-p))/Ln(2.0);
  END Hs;

  PROCEDURE Hc(p:LONGREAL):LONGREAL;
  BEGIN
    RETURN -Ln(p*p+(1.0-p)*(1.0-p))/Ln(2.0);
  END Hc;

  PROCEDURE feb(x:LONGREAL):LONGREAL;
  BEGIN
    RETURN v-v*Hs(x)-1.0+hsee;
  END feb;

  PROCEDURE Geteb(ee,factor:LONGREAL):LONGREAL;
  BEGIN
    hsee:=Hs(ee);
    v:=factor;
    RETURN Bisect(ee,0.5,feb);
  END Geteb;

  PROCEDURE fee(x:LONGREAL):LONGREAL;
  BEGIN
    RETURN v-v*hseb-1.0+Hs(x);
  END fee;

  PROCEDURE Getee(eb,factor:LONGREAL):LONGREAL;
  BEGIN
```

```
  hseb:=Hs(eb);
  v:=factor;
  RETURN Bisect(1.0E-10,eb,fee);
END Getee;

PROCEDURE KeepInRange(VAR x:LONGREAL;VAR f:LONGINT);
BEGIN
  IF (x>1.0E200) THEN
    f:=f-expo;
    x:=x/base;
  ELSIF (x<1.0E-200) THEN
    f:=f+expo;
    x:=x*base;
  END;
END KeepInRange;

PROCEDURE Normalize(VAR x:LONGREAL;f:LONGINT);
  VAR
    i:LONGINT;
BEGIN
  IF f>0 THEN
    FOR i:=1 TO f DO
      x:=x/2.0;
    END;
  ELSIF f<0 THEN
    FOR i:=1 TO -f DO
      x:=x*2.0;
    END;
  END;
END Normalize;

PROCEDURE Parity(ea,eb,ee:LONGREAL;k:LONGINT;VAR b,g,hs,hc,r:LONGREAL);
  VAR
    da,db,de,a00,a01,a10,a11,pcorrect,perror,paccept:LONGREAL;
    p1,p2,p3,p4,p1c,p2c,p3c,p4c,x:LONGREAL;
    f1,f2,f3,f4,w,i,n:LONGINT;
BEGIN
  n:=1;
  FOR i:=1 TO k DO n:=n*2 END;
  da:=1.0-ea;
  db:=1.0-eb;
  de:=1.0-ee;
  r:=1.0;             (* Berechnung des Reduktionsfaktors r *)
  x:=1.0-da*db-ea*eb;
  FOR i:=1 TO k DO
    r:=r*(x*x+(1.0-x)*(1.0-x))/2.0;
    x:=x*x/(x*x+(1.0-x)*(1.0-x));
  END;                                  (* Berechnung von beta      *)
  pcorrect:=Power((da*db+ea*eb)*2.0,n);     (* aus numerischen Gruenden *)
  perror:=Power((1.0-da*db-ea*eb)*2.0,n);   (* mit Faktor 2^n erweitert *)
  paccept:=pcorrect+perror;                 (* => um Faktor 2^n zu gross*)
```

```
b:=perror/paccept;
a00:=da*db*de+ea*eb*ee;                    (* Berechnung von gamma     *)
a01:=da*db*ee+ea*eb*de;
a10:=da*eb*de+ea*db*ee;
a11:=da*eb*ee+ea*db*de;
g:=0.0;
hs:=0.0;
hc:=0.0;
w:=(n+1) DIV 2;
f1:=0;
f2:=0;
f3:=0;
f4:=0;
p1:=1.0;
p2:=1.0;
p3:=1.0;
p4:=1.0;
FOR i:=w+1 TO n DO                        (* Berechnung des ersten p[w] *)
  x:=FLOAT(i)/FLOAT(i-w);
  p1:=p1*x*a00;
  p2:=p2*x*a10;
  p3:=p3*x*a01;
  p4:=p4*x*a11;
  KeepInRange(p1,f1);
  KeepInRange(p2,f2);
  KeepInRange(p3,f3);
  KeepInRange(p4,f4);
END;
FOR i:=1 TO w DO
  p1:=p1*a01;
  p2:=p2*a11;
  p3:=p3*a00;
  p4:=p4*a10;
  KeepInRange(p1,f1);
  KeepInRange(p2,f2);
  KeepInRange(p3,f3);
  KeepInRange(p4,f4);
END;
p1c:=p1;                            (* Kopien der pi werden normalisiert *)
p2c:=p2;
p3c:=p3;
p4c:=p4;
Normalize(p1c,f1-n);    (* n aus numerischen Gruenden schon in paccept *)
Normalize(p2c,f2-n);    (* weil sonst paccept ausserhalb Rechenbereich *)
Normalize(p3c,f3-n);
Normalize(p4c,f4-n);
g:=g+0.5*(p1c+p2c);                 (* Anpassung der Formel fuer gerade n *)
hs:=hs+0.5*(p1c+p2c+p3c+p4c)*Hs((p1c+p2c)/(p1c+p2c+p3c+p4c));
hc:=hc+0.5*(p1c+p2c+p3c+p4c)*Hc((p1c+p2c)/(p1c+p2c+p3c+p4c));
w:=w+1;                         (* Berechnung der weiteren relevanten p[w] *)
WHILE (w<=n) AND (g<g+(p1c+p2c)) DO
```

```
      x:=FLOAT(n-w+1)/FLOAT(w);
      p1:=p1*x/a00*a01;
      p2:=p2*x/a10*a11;
      p3:=p3*x/a01*a00;
      p4:=p4*x/a11*a10;
      KeepInRange(p1,f1);
      KeepInRange(p2,f2);
      KeepInRange(p3,f3);
      KeepInRange(p4,f4);
      p1c:=p1;                        (* Kopien der pi werden normalisiert *)
      p2c:=p2;
      p3c:=p3;
      p4c:=p4;
      Normalize(p1c,f1-n);
      Normalize(p2c,f2-n);
      Normalize(p3c,f3-n);
      Normalize(p4c,f4-n);
      g:=g+p1c+p2c;
      hs:=hs+(p1c+p2c+p3c+p4c)*Hs((p1c+p2c)/(p1c+p2c+p3c+p4c));
      hc:=hc+(p1c+p2c+p3c+p4c)*Hc((p1c+p2c)/(p1c+p2c+p3c+p4c));
      w:=w+1;
    END;
    g:=g/paccept;           (* jetzt ist gamma wieder im korrekten Bereich *)
    hs:=hs/paccept;
    hc:=hc/paccept;
  END Parity;

BEGIN
  expo:=662;
  base:=Power(2.0,expo);
END Protocol.
```

B.3 Math

```
DEFINITION MODULE Math;

  TYPE
    Function=PROCEDURE(LONGREAL):LONGREAL;

  PROCEDURE Power(x:LONGREAL;n:LONGINT):LONGREAL;
  PROCEDURE Bisect(a,b:LONGREAL;f:Function):LONGREAL;
  PROCEDURE Golden(ax,cx,tol:LONGREAL;f:Function;VAR xmax:LONGREAL):LONGREAL;

END Math.

IMPLEMENTATION MODULE Math;

  CONST
    R=0.61803399;
    C=1.0-R;

  PROCEDURE Power(x:LONGREAL;n:LONGINT):LONGREAL;
    VAR
      i:LONGINT;
      p:LONGREAL;
  BEGIN
    p:=1.0;
    FOR i:=1 TO n DO
      p:=p*x;
    END;
    RETURN p;
  END Power;

  PROCEDURE Bisect(a,b:LONGREAL;f:Function):LONGREAL;
    VAR
      fb:BOOLEAN;
      x:LONGREAL;
  BEGIN
    x:=(a+b)/2.0;
    fb:=f(b)>0.0;
    WHILE (x>a) AND (x<b) DO
      IF (f(x)>0.0)=fb THEN b:=x ELSE a:=x END;
      x:=(a+b)/2.0;
    END;
    RETURN x;
  END Bisect;

  PROCEDURE Golden(ax,cx,tol:LONGREAL;f:Function;VAR xmax:LONGREAL):LONGREAL;
    VAR
      bx,f1,f2,x0,x1,x2,x3:LONGREAL;
  BEGIN
    bx:=R*ax+C*cx;
    x0:=ax;
    x3:=cx;
```

```
    IF ABS(cx-bx)>ABS(bx-ax) THEN
      x1:=bx;
      x2:=bx+C*(cx-bx);
    ELSE
      x2:=bx;
      x1:=bx-C*(bx-ax);
    END;
    f1:=f(x1);
    f2:=f(x2);
    WHILE ABS(x3-x0)>tol*(ABS(x1)+ABS(x2)) DO
      IF f2>f1 THEN
        x0:=x1;x1:=x2;x2:=R*x1+C*x3;
        f1:=f2;
        f2:=f(x2);
      ELSE
        x3:=x2;x2:=x1;x1:=R*x2+C*x0;
        f2:=f1;
        f1:=f(x1);
      END;
    END;
    IF f1>f2 THEN
      xmax:=x1;
      RETURN f1;
    ELSE
      xmax:=x2;
      RETURN f2;
    END;
  END Golden;

END Math.
```