

- Bei großen Multicast-Aufgaben
 - ▶ Versand langer Programmcodes oder
 - ▶ Videos hoher Qualität an mehrere hunderttausende Empfänger müssen Pakete in Echtzeit beim Empfänger ankommen.
- Nicht-rekonstruierbarer Paketverlust muss drastisch reduziert werden!

Deshalb redundante Kodierung mit den Zielvorgaben:

- Rekonstruktion trotz hohen Paketverlusten.

Wir geben allerdings keine Worst-Case Garantie, sondern garantieren Rekonstruktion nur mit hoher Wahrscheinlichkeit.

- Superschnelle Kodierungs- und Dekodierungsalgorithmen, um die hohen Bandbreiten von Hochleistungskanälen

Glasfaser, ATM oder Gigabit Ethernet ausnutzen zu können.

- Verlässliche Kommunikation mit einer Vielzahl von Empfängern über unsichere Kanäle
 - wechselnde Landschaften, Tunnelist ein Schlüsselproblem.
- Eine Rückverbindung zum Empfänger besteht nicht:
 - Ein nicht-kompensierbarer Paket-Verlust muss vermieden werden.

- Für ein Alphabet Σ werden Nachrichten der Form

$$(a_1, 1) \cdot (a_2, 2) \cdots (a_i, i) \cdots (a_n, n)$$

(mit $a_1, \dots, a_n \in \Sigma$) über den Kanal K verschickt.

Das Alphabet Σ kann zum Beispiel aus allen Bitfolgen einer bestimmten Höchstlänge bestehen.

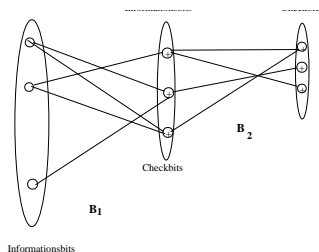
- K löscht ein Paket (a_i, i) mit Wahrscheinlichkeit $\leq \rho$, unabhängig davon, ob vorher andere Pakete gelöscht wurden.
 - ▶ **Ist diese Annahme realistisch?** Die Löschwahrscheinlichkeiten zeitlich benachbarter Pakete sollten stark korreliert sein!
 - ▶ Eine solche Korrelation können wir auflösen, wenn wir „gegen den Kanal randomisieren“, also die Pakete vor dem Versand **zufällig permutieren**.
- Paket (a_i, i) besitzt den Zeitstempel i : Der Empfänger weiß also, welche Pakete verloren gegangen sind.

Tornado Codes

Tornado Codes

- tolerieren hohe Verlustraten und
 - bieten superschnelle Kodier- und Dekodieralgorithmen an.
-
- Um die Beschreibung zu vereinfachen, nehmen wir an, dass jedes Paket nur aus einem Bit und der Paketnummer besteht.
 - Der Tornado Code wird durch eine Folge „dünn-besetzter“ bipartiter Graphen $B_i = (U_i, V_i, E_i)$ (mit $|V_i| = \beta \cdot |U_i|$ für $\beta < 1$ und $|E_i| = O(|U_i|)$) definiert.

Die bipartiten Graphen:



- Wenn $|V_k| \leq \sqrt{n}$ wird eine Reed-Solomon Kodierung, ein fehlerkorrigierender Code, der in quadratischer Zeit kodiert und dekodiert, als letzte Kodierungsstufe der Hierarchie eingefügt.
- Die Folge der Informations- und Checkbits ist die Kodierung der Eingabe.

Die Kodierung

- Die „linken“ Knoten von B_i , also die Knoten in U_i , heißen **Informationsbits** und die „rechten“ Knoten von B_i , also die Knoten in V_i , **Check-Bits**.
Die Informationsbits von B_1 entsprechen den Eingabebits.
- Für die **Kodierung**:
 - 1 Berechne zuerst die Werte aller Checkbits von B_1 durch **Xoring** der inzidenten Informationsbits von B_1 .
 - 2 Sodann interpretiere die rechten Bits von B_1 als die Informationsbits von B_2 :
 - ★ Berechnen die Checkbits von B_2 , also die rechten Knoten von B_2 , durch Xoring der inzidenten Informationsbits von B_2 .

Dieses Verfahren wird für alle bipartiten Graphen wiederholt.

Die bipartiten Graphen

Erzeugung der bipartiten Graphen $B_i = (U_i, V_i, E_i)$

Erzeuge B_i für $1 \leq i \leq m$, wobei $n \cdot \beta^m \leq \sqrt{n}$ gelte.

- (a) Bestimme Grad-Sequenzen $u = (u_{i,2}, \dots, u_{i,L})$, $v = (v_{i,2}, \dots, v_{i,R})$:
 U_i , bzw. V_i , besitze genau $u_{i,j}$, bzw. $v_{i,j}$, Knoten vom Grad j .

$$B_i \text{ besitzt } e_i = \sum_{j=2}^L j \cdot u_{i,j} = \sum_{j=2}^R j \cdot v_{i,j} \text{ Kanten.}$$

- (b) Setze $U_i^* = \{1, \dots, e_i\} = V_i^*$ und
bestimme eine zufällige Bijektion $b_i : U_i^* \rightarrow V_i^*$.

- (c) Konstruktion der Knotenmenge:

- ▶ Zerlege U_i^* (V_i^*) in beliebiger Weise in $u_{i,j}$ (bzw. $v_{i,j}$) disjunkte Teilmengen der Größe j (für $j = 2, \dots, L$, bzw. $j = 2, \dots, R$).
- ▶ „Kollabiere“ die erhaltenen Teilmengen auf jeweils einen Knoten und nenne die neuen Knotenmengen U_i und V_i .

- (d) Verbinde einen Knoten $u \in U_i$ mit einem Knoten in $v \in V_i$, wenn ein Element der Teilmenge von u auf ein Element der Teilmenge von v durch die Bijektion b abgebildet wird.

/ U_i (bzw. V_i) besitzt genau $u_{i,j}$ (bzw. $v_{i,j}$) Knoten vom Grad j . */*

- Wir haben die bipartiten Graphen B_i eingeführt, haben allerdings die Gradsequenzen nicht spezifiziert.
 - Achtung:** Die Gradsequenzen spielen eine wichtige Rolle!
- Wir haben die „Kodierung durch Xoring“ beschrieben:
 - ▶ Die Kodierung gelingt schnell, nämlich in Zeit linear in der Anzahl aller Kanten.
 - ▶ Die Anzahl der Kanten sollte also nicht zu groß sein!
- Wie ist zu dekodieren, wenn Bits fehlen?

Die Dekodierung

Die kodierte Folge ist nach Verlust einiger Bits zu dekodieren.

- 1 Beginne die Rekonstruktion mit dem Reed-Solomon Code. Setze $i = m$.
- 2 Wiederhole, solange es mindestens ein Check-Bit c in B_i mit **genau einem** fehlenden Vorgänger b_0 gibt:
Setze $b_0 = c \oplus b_1 \oplus \dots \oplus b_k$,
wobei b_1, \dots, b_k die vorhandenen Vorgänger von c seien.
- 3 Wenn ein linker Knoten von B_i nicht korrigiert werden konnte, dann brich mit einer Fehlermeldung ab.
 - ▶ Ansonsten, wenn $i = 1$, dann ist die Korrektur geglückt.
 - ▶ Wenn $i > 1$, dann setze $i = i - 1$ und wiederhole Schritt (3).

Die Gradsequenzen

Warum wählen wir nicht **reguläre** bipartite Graphen, also Graphen in denen alle linken Knoten denselben Grad und alle rechten Knoten denselben Grad besitzen?

- Vom **Standpunkt eines Informationsbits** (also eines linken Knotens):
Ein hoher Grad ist hochwillkommen, denn dann bestehen die größten Chancen von irgendeinem der vielen rechten Nachbarn bestimmt zu werden.
- Vom **Standpunkt eines Checkbits** (also eines rechten Knotens):
Der Grad sollte möglichst klein sein, um höchstens einen fehlenden linken Nachbarn zu besitzen.

Wir versuchen einen Kompromiss und stattdessen wenige linke Knoten mit einem hohen Grad aus.

Die Rekonstruktion sollte in Wellen verlaufen.

- Die wenigen hochgradigen linken Nachbarn werden sofort rekonstruiert,
- gefolgt von ihren etwas niedrig-gradigeren Kollegen und so weiter.
- Die vielen niedrigst-gradigen linken Knoten werden dann in wenigen letzten Wellen rekonstruiert: Der Tornado produziert rasant wachsende Wellen rekonstruierter linker Knoten.

- Aber wie genau sollten die Gradsequenzen gewählt werden?
- Wir führen zuerst eine Analyse durch.

Tornado Codes: Die Informationsrate

Die Informationsrate eines Codes ist

$$\frac{\text{Eingabelänge}}{\text{Codewortlänge}}$$

- Die Codewortlänge beträgt

$$n + \sum_{i=1}^m \beta^i \cdot n + O(\sqrt{n}) = n \cdot \frac{1 - \beta^{m+1}}{1 - \beta} + O(\sqrt{n}) \leq \frac{n}{1 - \beta} + O(\sqrt{n}).$$

- Da n Eingabebits vorliegen, erhalten wir die Informationsrate

$$\frac{1 - \beta}{1 + O\left(\frac{1 - \beta}{\sqrt{n}}\right)}$$

Und wenn der Kanal ein Bit mit Wahrscheinlichkeit ρ verliert?

Wann sind Tornado Codes gut?

Wenn der Kanal ein Bit mit Wahrscheinlichkeit ρ verliert:

- Eine Codewortlänge von mindestens $\frac{n}{1-\rho}$ Bits ist notwendig, um den Bitverlust des Kanals zu kompensieren.
- Die optimale Informationsrate beträgt bestenfalls $1 - \rho$.

- Wir machen den Ansatz $\beta = \rho(1 + \varepsilon)$.
- **Wenn** eine Rekonstruktion in jeder Hierarchiestufe B_i mit hoher Wahrscheinlichkeit gelingt:
 - ▶ Eine Rekonstruktion gelingt hochwahrscheinlich mit der Rate

$$\frac{1 - \beta}{1 + O\left(\frac{1-\beta}{\sqrt{n}}\right)} \approx \frac{1 - \rho - \rho \cdot \varepsilon}{1 + O\left(\frac{1-\beta}{\sqrt{n}}\right)} \approx 1 - \rho - \rho \cdot \varepsilon.$$

Für $\beta = \rho(1 + \varepsilon)$ und kleinem ε zeige, dass eine Rekonstruktion hochwahrscheinlich gelingt, selbst wenn ein Anteil ρ aller Bits fehlt.

Dekodierung: Die Situation

- Wir stellen uns vor, dass ein durch das Löschen von Bits verstümmeltes Codewort w zu dekodieren ist.
 - ▶ Die Wahrscheinlichkeit des Löschens einer Bitposition sei ρ und die Löschungen der Positionen seien stochastisch unabhängig.
- Wir nehmen weiterhin an, dass die Rückwärtsdekodierung den bipartiten Graphen $B = B_j$ erreicht hat.
 - ▶ Jeder rechte Knoten von B ist entweder bereits vorhanden oder wurde erfolgreich rekonstruiert.

- Sei v ein beliebiger linker Knoten von B , dessen Bit verloren ist. Wann können wir das verlorene Bit bestimmen?
- Wir sagen, dass ein rechter Knoten von B **rekonstruiert**, falls höchstens einer seiner linken Nachbarn verloren gegangen ist.

Die Analyse: Und-Oder Bäume

- Wir interpretieren v als einen **ODER-Knoten**:
 - ▶ Das Bit von v kann bestimmt werden, wenn v **mindestens einen rekonstruierenden** rechten Nachbarn w besitzt.
 - Wann ist ein rechter Nachbar w von v rekonstruierend?
 - ▶ Genau dann, wenn **alle** linken Nachbarn von w , mit der Ausnahme von v , vorhanden sind.
- Jeder rechte Nachbar von v spielt die Rolle eines **UND-Knotens**.

Wir haben noch nicht verloren, wenn sich kein rekonstruierender rechter Nachbar von v findet:

„Wer noch kein rekonstruierender Knoten ist, der kann's noch werden“!

Angenommen, v hat keinen rekonstruierenden rechten Nachbarn.

- Nach der ersten Generation, den rechten Nachbarn von v , schauen wir uns die zweite Generation, die Enkelkinder von v , an.
 - ▶ Auch die Enkelkinder von v lassen sich als ODER-Knoten auffassen:
 - ▶ Falls ein Enkelkind v' nicht vorhanden ist, gelingt die Bestimmung des zugehörigen Bits, wenn mindestens einer der rechten Nachbarn von v' rekonstruierend ist.
- Wir wiederholen diesen Prozess, bis wir einen Baum $T_l(v)$ der Tiefe $2l$ erhalten:
 - ▶ Der ODER-Knoten v ist die Wurzel, gefolgt von den rechten Nachbarn als UND-Knoten, gefolgt von deren linken Nachbarn (mit Ausnahme von v) als ODER-Knoten und so weiter.

Wann kann die Wurzel rekonstruiert werden?

Der Baum $T_l(v)$

Die Blätter von $T_l(v)$ sind linke Knoten, die wir mit 0 (nicht vorhanden) oder 1 (vorhanden) beschriften.

- Die Knoten der Tiefe $2l - 1$ rekonstruieren genau dann, wenn, bis auf den Elternknoten, **alle** linken Nachbarn vorhanden sind.
 - ▶ Ein Knoten der Tiefe $2l - 1$ ist rekonstruierend, wenn der Knoten als **UND-Knoten** den Wert 1 berechnet.
- Wann ist ein Knoten v' der Tiefe $2l - 2$ vorhanden oder rekonstruierbar?
 - ▶ Friere v' auf 1 fest, wenn das entsprechende Bit nicht verloren ist.
 - ▶ v' ist genau dann vorhanden oder rekonstruierbar,
 - ★ wenn v' festgefroren ist, oder
 - ★ wenn **mindestens eines** seiner Kinder rekonstruierend ist, (bzw. wenn v' als **ODER-Knoten** den Wert 1 annimmt).

Die Wurzel v ist genau dann vorhanden oder rekonstruierbar, wenn der UND-ODER Baum $T_l(v)$ den Wert 1 berechnet!

Wähle die Tiefe $2l$ so klein, dass der **induzierte** Teilgraph von B mit hoher Wahrscheinlichkeit ein Baum ist.

- Was ist die Wahrscheinlichkeit q_j (bzw. p_j), dass eine Kante in B einen linken (bzw. rechten) Knoten mit einem rechten (bzw. linken) Knoten vom Grad j verbindet?
 - ▶ Wir besitzen insgesamt $\sum_{j=2}^L j \cdot u_{i,j} = \sum_{j=2}^R j \cdot v_{i,j}$ Kanten.
 - ▶ Von diesen Kanten verbinden
 - ★ genau $j \cdot v_{i,j}$ Kanten einen linken Knoten mit einem rechten Knoten vom Grad j , und
 - ★ genau $j \cdot u_{i,j}$ Kanten einen rechten Knoten mit einem linken Knoten vom Grad j .
- Es ist

$$p_j = \frac{j \cdot u_{i,j}}{\sum_{j=2}^L j \cdot u_{i,j}} \quad \text{und} \quad q_j = \frac{j \cdot v_{i,j}}{\sum_{j=2}^R j \cdot v_{i,j}}.$$

- Sei v' ein beliebiger Knoten in $T_l(v)$ ungerader Tiefe und $e = (v', u)$ eine Kante zu einem Kind u .
 - ▶ v' ist ein rechter Knoten und u ein linker Nachbar.
 - ▶ Mit Wahrscheinlichkeit p_j verbindet die Kante $e = (v', u)$ den Knoten v' mit einem Knoten vom Grad j :
 - Ein innerer Knoten gerader Tiefe besitzt genau $j - 1$ Kinder mit Wahrscheinlichkeit p_j .
 - Eine Kante von u wurde bereits vom Elternknoten v' geschluckt.
- Mit analoger Argumentation:

- (a) Ein innerer Knoten gerader Tiefe besitzt genau $j - 1$ Kinder mit Wahrscheinlichkeit p_j ,
- (b) ein innerer Knoten ungerader Tiefe besitzt genau $j - 1$ Kinder mit Wahrscheinlichkeit q_j .

- Die Kinderzahlen entsprechen **nicht** unabhängigen Zufallsvariablen:
 - ▶ Wenn bereits viele Knoten mit fixierter Kinderzahl im Baum auftreten, wird diese Kinderzahl entsprechend unwahrscheinlicher,
 - ▶ denn unsere bipartiten Graphen besitzen eine festgelegte Anzahl von Knoten für jeden Grad.
- Die Wahrscheinlichkeiten für die Kinderzahlen sind nur als approximativ anzusehen.
- Die Bäume $T_l(v)$ sollten klein bleiben, damit die Approximation nur mit einem kleinen, vernachlässigbaren Fehler behaftet ist.

Der Baum $T_i(v)$ und der Erasure Kanal

Der Baum $T_i(v)$ ist ein Teilbaum des zufälligen bipartiten Graphen B und somit selbst zufällig:

- Ein ODER-Knoten v_{ODER} hat mit Wahrscheinlichkeit p_i genau $i - 1$ Kinder und
- ein UND-Knoten u_{UND} mit Wahrscheinlichkeit q_i genau $i - 1$ Kinder.

Wir modellieren den zufällig löschenden Erasure Kanal:

- Markiere die Blätter zufällig durch 0 (mit Wahrscheinlichkeit ρ) oder durch 1 (mit Wahrscheinlichkeit $1 - \rho$).
- Linke Knoten, also Knoten gerader Tiefe, werden ebenfalls mit Wahrscheinlichkeit $1 - \rho$ auf den Wert 1 festgefroren.

Bestimme die Wahrscheinlichkeit $\text{Null}_l(\rho)$, dass die angelegte zufällige Eingabe die Ausgabe 0 an der Wurzel produziert.

- Der Wahrscheinlichkeitsraum besteht aus allen Paaren (T, x) , so dass
 - ▶ T ein zufälliger Baum der Tiefe $2l$ mit den Kinder-Wahrscheinlichkeiten \vec{p} und \vec{q} ist
 - ▶ x eine zufällige Eingabe mit $\text{prob}[x_i = 0] = \rho$ für alle Positionen i ist.
- Erstaunlicherweise lässt sich die Wahrscheinlichkeit $\text{Null}_l(\rho)$ leicht rekursiv bestimmen.

Für die Verteilungen $\vec{p} = (p_1, \dots, p_L)$ und $\vec{q} = (q_1, \dots, q_R)$ definiere die Polynome

$$p(x) = \sum_{i=2}^L p_i \cdot x^{i-1} \quad \text{und} \quad q(x) = \sum_{i=2}^R q_i \cdot x^{i-1}$$

sowie das Iterationspolynom $f(x) = \rho \cdot p(1 - q(1 - x))$.

Behauptung: Setze $\text{Null}_0(\rho) = \rho$. Dann ist

$$\text{Null}_{l+1}(\rho) = f(\text{Null}_l(\rho)).$$

Bestimmung von $\text{Null}(\rho)$

Es ist $f(\text{Null}_l(\rho)) = \rho \cdot p(1 - q(1 - \text{Null}_l(\rho)))$.

- Betrachte einen UND-Knoten u der Tiefe 1.
 - ▶ Der UND-Knoten besitzt $i - 1$ Kinder mit Wahrscheinlichkeit q_i .
 - ▶ u erhält nur dann der Wert 1, wenn alle $i - 1$ ODER-Kinder den Wert 1 erreichen:
 - ★ Dies passiert mit Wahrscheinlichkeit $(1 - \text{Null}_l(\rho))^{i-1}$.
 - ▶ $q(1 - \text{Null}_l(\rho))$ ist die Wahrscheinlichkeit, dass ein UND-Knoten der Tiefe 1 wahr wird.
- Wann erhält eine Wurzel mit $j - 1$ Kindern den Wert 0?
 - ▶ Die Wurzel darf nicht festgefroren sein,
 - ★ und dies passiert mit Wahrscheinlichkeit ρ ,
 - und** kein Kind darf den Wert 1 annehmen,
 - ★ und dies passiert mit Wahrscheinlichkeit $(1 - q(1 - \text{Null}_l(\rho)))^{j-1}$.
 - ▶ Die Wurzel besitzt $j - 1$ Kinder mit Wahrscheinlichkeit p_j und die Behauptung folgt.

Wie verhält sich f für kleine Werte?

- Die Polynome $p(x)$ und $q(x)$ haben nicht-negative Koeffizienten: Die Polynome sind monoton wachsend für $x \geq 0$.
- Also ist auch $f(x) = \rho \cdot p(1 - q(1 - x))$ monoton wachsend.

Wir möchten erreichen,
dass $\text{Null}_l(\rho) = f^{(l)}(\rho)$ schnell kleine Werte annimmt.

- Wir fordern $f(x) < x$ für $0 < x \leq \rho$.
 - ▶ In diesem Fall konvergiert die Folge $f^{(l)}(\rho)$ gegen Null.
 - ▶ Ist die Forderung für $x_0 \leq \rho$ verletzt, dann gilt
 $f^{(l)}(x_0) > x_0$ für jedes l .

Die Gradverteilungen \vec{p} und \vec{q}

Für die Verteilungen

$$p_i = \frac{1}{(i-1) \cdot \sum_{i=1}^d \frac{1}{i}}$$

($i = 2, \dots, d+1$) und, für ein geeignetes α ,

$$q_i = \frac{e^{-\alpha} \cdot \alpha^{i-1}}{(i-1)!}$$

($i \geq 2$) gilt $f(x) < x$ für $x \leq \rho$, falls $\rho \leq \beta \cdot \frac{d}{d+1}$.

Für die Argumentation siehe das Skript.

Für jede Bitposition:

Die Wahrscheinlichkeit der erfolglosen Rekonstruktion ist durch eine beliebig kleine Konstante nach oben beschränkt.

Aber das reicht doch nicht!

Es ist $p_i = \frac{1}{(i-1) \cdot \sum_{i=1}^d \frac{1}{i}}$ und $q_i = \frac{e^{-\alpha} \cdot \alpha^{i-1}}{(i-1)!}$ sowie $\rho \leq \beta \cdot \frac{d}{d+1}$.

- β ist nur unwesentlich größer als ρ , falls d hinreichend groß.
 - ▶ Wir haben unser Ziel $\beta = \rho \cdot (1 + \varepsilon)$ erreicht
 - ▶ und die Codewortlänge ist fast optimal.
- Ein linker Knoten besitzt den Grad i mit Wahrscheinlichkeit $\Theta\left(\frac{1}{\ln(d)} \cdot \frac{1}{i}\right)$, der durchschnittliche Grad ist $\Theta(d / \ln(d))$.
- Nur sehr wenige rechte Knoten von hohem Grad.

Das verbleibende Problem:

Ein kleiner Prozentsatz aller Informationsbits ist **nicht rekonstruierbar**.

denn der Anteil nicht rekonstruierbarer Informationsbits ist klein!

- Füge neue Checkbits für jedes B_i hinzu und verbinde jedes Informationsbit mit genau g neuen Checkbits über linear viele, zufällig eingesetzte Kanten.
- Sei U' die Menge nicht-rekonstruierbarer Informationsbits.
 - ▶ U' besitzt mindestens $\frac{g}{2} \cdot |U'| + 1$ Nachbarn:
Bei zufälliger Kantenwahl wird die **kleine** Menge U' *expandieren*.
 - ▶ Es gibt mindestens einen Nachbarn von U' , der nur mit genau einem Knoten $u \in U'$ verbunden ist!
 - ★ Wenn jeder Nachbar mit mindestens zwei Knoten aus U' verbunden ist, dann kann U' höchstens $\frac{g}{2} \cdot |U'|$ Nachbarn besitzen.
 - ▶ Dieser Knoten u ist jetzt rekonstruierbar! Wiederhole das Argument, um vollständige Rekonstruierbarkeit zu erhalten.

- + Tornado Codes sind schon bei kurzen Dateien um den Faktor 100 schneller in Kodierung und Dekodierung,
 - ▶ für lange Dateien wächst der Geschwindigkeitsvorteil auf den Faktor 10.000 an.
- Tornado Codes verstärken sogar den Effekt verfälschter Pakete, allerdings werden Pakete nur äußerst selten verfälscht.

Online Codes, LT (Luby Transform) Codes and Raptor Codes sind Varianten von Tornado Codes.