Tehnici de codare de tip Digital Fountain

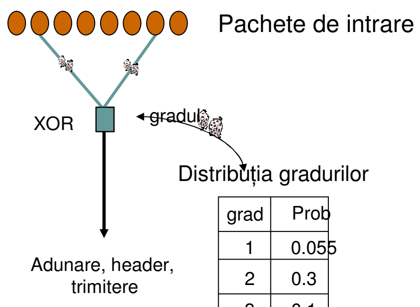
TACCFDRT Curs 2

Tehnici de codare de tip Digital Fountain

- Coduri LT
  - Distribuția Soliton ideală(recapitulare)
  - Distribuția Soliton robust
- Coduri Tornado
- Coduri Raptor

Principiul de codarea LT [3]

- Pentru pachetul codat care urmează să fie generat se alege aleator un grad *d*, conform unei distribuţii predefinite
- Se aleg aleator *d* pachete din mulţimea pachetelor informaţionale
- Pachetul codat se obţine adunând modulo doi pachete selecţionate la pasul anterior



grad	Prob
1	0.055
2	0.3
3	0.1
4	0.08
⋮	⋮
100000	0.0004

[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.

- Procesul de decodare:
  - La început toate simbolurile informaționale sun descoperite
  - În prima etapă toate simbolurile codate care sunt formate dintr-un singur simbol informațional, “acoperă” singurul lor vecin. Mulțimea formată din simbolurile acoperite, care încă nu au fost procesate, se numeşte “riplu”
  - În paşii următor este luat câte un simbol informațional din riplu, este adunat la simbolurile codate la care este vecin şi se reduce gradul acestor simboluri.
  - Dacă un simbol codat, astfel va avea grad 1, acest simbol va acoperi vecinul său, iar acest simbol codat va fi “achitat” Dacă acest vecin acoperit nu a fost acoperit mai devreme, de un alt simbol codat, atunci dimensiunea riplu-ului creşte.
  - Procesul se termină când riplul se goleşte
    - Procesul de decodare este cu succes dacă la golirea riplului nu mai sunt simboluri de intrare neacoperite.

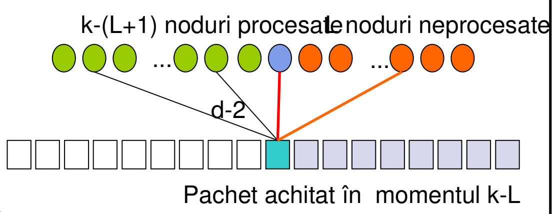
Distribuția *Soliton* ideală

- Cerinţele impuse unei distribuţii de graduri sunt:
  - Un număr mediu de simboluri codate cât mai mic posibil pentru a asigura succesul procesului LT.
  - Gradul mediu al simbolurilor cât mai mic posibil. Gradul mediu defineşte numărul operațiilor de simbol necesare pentru generarea unui simbol codat, iar k*(grad mediu) este numărul operațiilor necesare pentru recuperarea completă a datelor.

Distribuția *Soliton* ideală

- O proprietate elementară a unei distribuţii “ideale” este ca la procesul de decodare, la procesarea unui simbol informaţional la riplu să fie adăugat un simbol acoperit.
- Asta asigură că dimensiunea riplului să nu fie niciodată prea mică sau prea mare.
- Dimensiunea riplului în medie este 1 dacă distribuţia gradurilor este:

$$p(d)=\begin{cases}\frac{1}{k}; & pt\ d=1\\ \frac{1}{d(d-1)}; & pt\ d=2,\ldots,k\end{cases}$$



Distribuția *Soliton* ideală

- Sunt necesare exact k simboluri codate pentru reconstruirea celor k simboluri de intrare
- Dimensiunea riplului este 1 pe toată durata decodări
- PERFORMAŢE FOARTE SLABE în practică
  - Deoarece riplul este foarte scurt, riplul poate să se golească înaintea decodării tuturor mesajelor informaționale

Distribuția Soliton Robust

- Cu cât dimensiunea riplului este mai mare cu atât probabilitatea ca riplul să se golesacă înaintea recuperării tuturor simbolurilor informaţionale este mai mică
- Pentru a minimiza numărul total de simbolul codate utilizate la recuperarea simbolurilor informaţionale dimensiunea riplului trebuie să fie cât mai mic posibil

Distribuția Soliton Robust

- Cât trebuie să fie suma gradurilor pachetelor recepţionate ca fiecare pachet informaţional să fie acoperit cu probabilitate 1-δ?
  - Problema este echivalentă cu problema “clasică” coşuri şi mingi:
    - avem k coşuri
    - se aruncă K mingi, fiecare minge intră într-unul dintre coşuri
  - Câte mingi trebuie arucate ca probabilitatea ca în fiecare coş să intre cel puţin o minge să fie 1-δ

Distribuția Soliton Robust

- Câte mingi trebuie arucate ca probabilitatea ca în fiecare coş să intre cel puţin o minge să fie 1-δ
- Probabilitatea ca o minge să intre în coşul i este 1/k
- Probabilitatea ca după N încercări să existe un coş gol este:

$$k\left(1-\frac{1}{k}\right)^N$$

- Probabilitatea ca după N încercări să existe un coş gol trebuie să fie δ

$$k\left(1-\frac{1}{k}\right)^N=\delta$$

- Logaritmând ecuaţia obţinem:

$$\ln\left(1-\frac{1}{k}\right)^N=\ln\left(\frac{\delta}{k}\right)$$

Distribuția Soliton Robust

$$\ln\left(1-\frac{1}{k}\right)^{N} = \ln\left(\frac{\delta}{k}\right)$$

- Dar....

$$N\ln\left(1-\frac{1}{k}\right)^{\frac{k}{k}} = \ln\left(\frac{\delta}{k}\right)$$

- Adică....

$$N\frac{1}{k}\ln\left(1-\frac{1}{k}\right)^{k} = \ln\left(\frac{\delta}{k}\right)$$

- Dacă k este suficient de mare atunci:
- $\left(1-\frac{1}{k}\right)^{k} \approx \frac{1}{e}$ şi $N\frac{1}{k}\ln\left(\frac{1}{e}\right) = \ln\left(\frac{\delta}{k}\right)$

Distribuția Soliton Robust

$$N\frac{1}{k}\ln\left(\frac{1}{e}\right) = \ln\left(\frac{\delta}{k}\right)$$

- Dar....

$$N\ln(e) = k\cdot\ln\left(\frac{k}{\delta}\right)$$

- Adică $N = k\cdot\ln\left(\frac{k}{\delta}\right)$

Distribuția Soliton Robust

- Dacă dimensiunea riplului este $\ln\left(\frac{k}{\delta}\right)\sqrt{k}$ atunci probabilitatea de golire a riplului înainte de terminarea decodării este maxim δ
  - Cu procesarea unui nod riplul poate să scadă sau poate să crească cu unu
  - Asta este echivalent cu un "random walk" unidimensional care are lungimea pasului egală cu unu
  - În cazul unui "random walk" de lungime k, o deviațiw față de valoarea medie mai mare ca $\ln\left(\frac{k}{\delta}\right)\sqrt{k}$

apare cu probabilitatea δ

Distribuția Soliton Robust

- Pentru un compromis se acceptă că din K pachete recepţionate, decodorul LT nu reuşeşte să determine informaţiile originale cu probabilitatea δ
- pentru asigurarea ca probabilitatea de eroare să fie maxim δ, dimensiunea riplului trebuie să fie:

$$\ln\left(\frac{k}{\delta}\right)\sqrt{k}$$

- Iar numărul pachetelor codate necesare pentru decodare este:

$$K = k + O\left(\ln^2\left(\frac{k}{\delta}\right)\sqrt{k}\right)$$

Distribuția Soliton Robust

- Se alege lunngimea riplului dorit ca fiind:

$$R = c \ln\left(\frac{k}{\delta}\right)\sqrt{k} \quad unde\ c > 0$$

- Se defineşte τ(d) ca fiind

$$\tau(d) = \begin{cases} \frac{R}{dk} & pt\ d = 1,\dots,\frac{k}{R}-1 \\ R\frac{\ln\left(\frac{R}{\delta}\right)}{k} & pt\ d = \frac{k}{R} \\ 0 & pt\ d = \frac{k}{R}+1,\dots,k \end{cases}$$

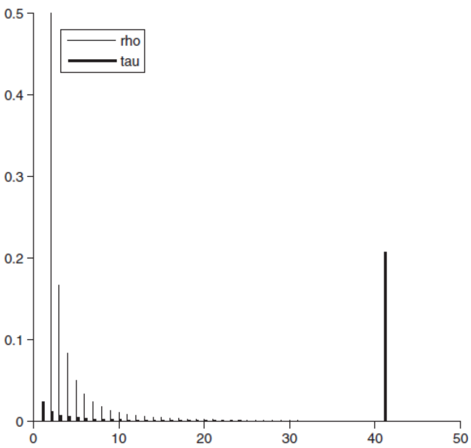
Distribuția Soliton Robust

- Se adună distribuţia ideală p(d) la τ(d), normalizând această sumă cu β se obţine distribuţia robustă μ(d)

$$\beta = \sum_{d=1}^{k} p(d) + \tau(d)$$

$$\mu(d) = \frac{p(d)+\tau(d)}{\beta}$$

- Valoarea medie a gradurilor este c*ln*(k)
- Ovearheadul necesar este proporţional cu K



$$p(d)=\begin{cases}\frac{1}{k}; & pt\ d=1\\ \frac{1}{d(d-1)}; & pt\ d=2,\dots,k\end{cases}$$

$$\tau(d)=\begin{cases}\frac{R}{dk} & pt\ d=1,\dots,\frac{k}{R}-1\\ R\frac{\ln\left(\frac{R}{\delta}\right)}{k} & pt\ d=\frac{k}{R}\\ 0 & pt\ d=\frac{k}{R}+1,\dots,k\end{cases}$$

- Distribuţia soliton ideală şi robustă pentru k=10000, c=0.2, δ=0.05, R=244, k/R=41, iar β≈1.3 [1]

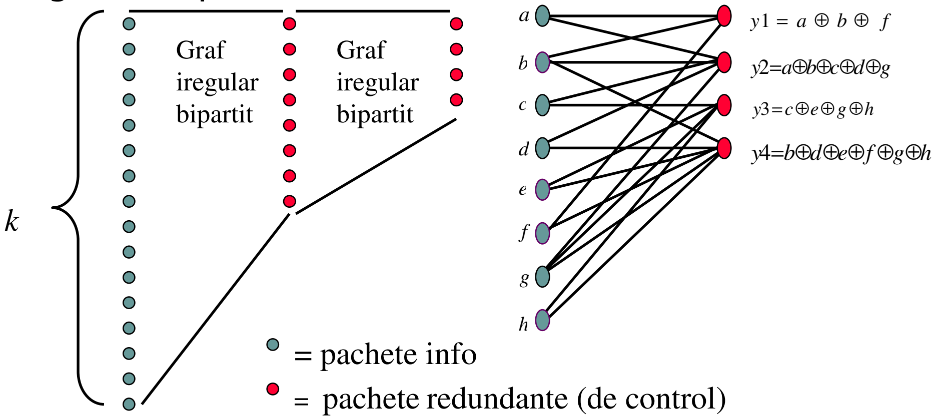
[1] MacKay, D.J.C.: 'Information theory, inference, and learning algorithms' -Cambridge University Press, 2003

- În articolul lui Luby (2002) a demonstrat că primele valori a distribuţiei $\tau(d)$ (d mic) asigură pornirea preocesului de decodare, adică asigură ca la începutul decodării riplul să "crească" la dimensiunea dorită
- iar valoarea relativ ridicată a funcţiei $\tau(d)$ pentru d = k/R este inclus pentru a asigura ca fiecare simbol (pachet) informaţional are cel puţin un vecin între simbolurile recepţionate, adică acest "vârf" asigură ca suma suma gradurilor să fie suficient de mare ca cele $K=k+O\left(\ln^2\left(\frac{k}{\delta}\right)\sqrt{k}\right)$ pachete recepţionate să acopere toate cele k simboluri informaţionale

- Deoarece gradul pachetelor nu este constant
  - Timpul de codare/decodare nu este liniară
  - Probabilitatea de pierdere a pachetelor nu este uniformă
- Problema este rezolvată de codurile Raptor introduse de Amin Shokrollahi

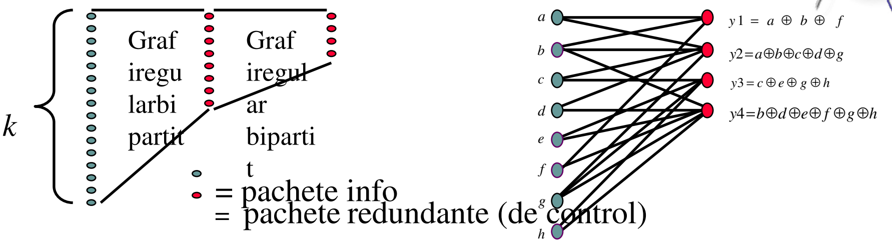
Coduri Tornado

- Codurile Tornado sunt construite pe baza unor grafuri bipartite



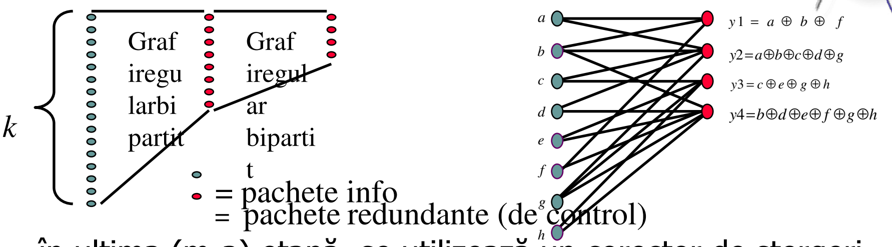
[2] John W.Byers, Michael Luby, Michael Mitzenmacher, Ashutosh Rege; “A Digital Fountain Approach to Reliable Distribution of Bulk Data” -Proceedings of the ACM SIGCOMM '98

Coduri Tornado



- în prima etapă din cele k pachete informaţionale se obţin βk pachete de control (β număr pozitiv subunitar) conform grafului B1
- în a doua etapă pornind de la cele βk pachete de control obţinute în etapa anterioară se obţin alte ββk pachete de control conform grafului B2

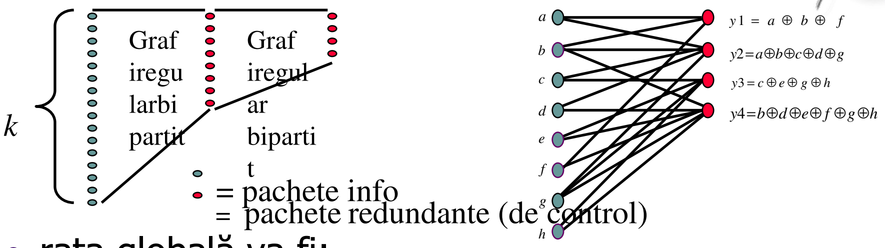
Coduri Tornado



- în ultima (m-a) etapă se utilizează un corector de ştergeri "clasic", C, cu rata 1- β care poate să corecteze in număr de lungime·β ştergeri
- Codul C are β^{m-1}k simboluri (pachete) la intrare, şi generează β β^{m-1}k/(1- β) simboluir de control adiţionale
- Numărul simbolurilor de control generate în cele m etape este

$$\sum_{i=1}^{m-1}\beta^i k + \frac{\beta^m k}{1-\beta} = \frac{k\beta}{1-\beta}$$

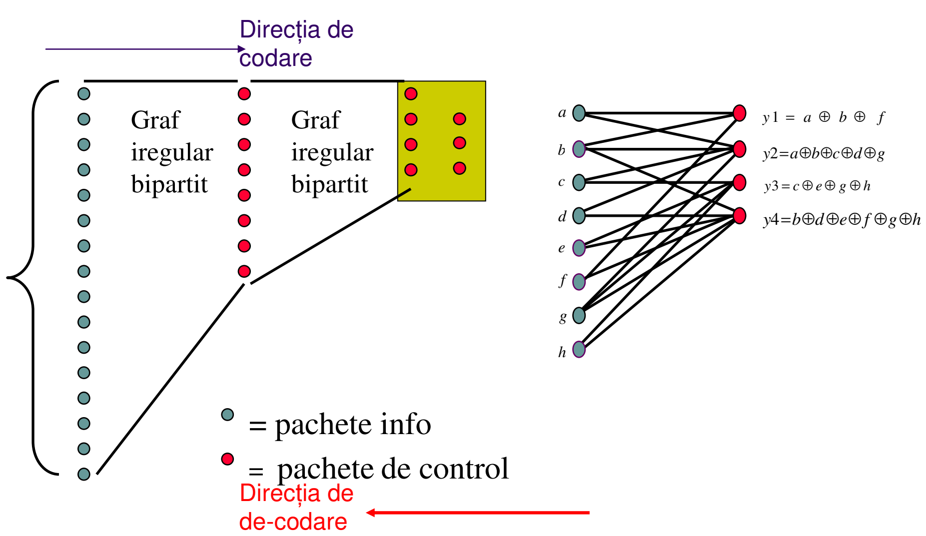
Coduri Tornado



- rata globală va fi:

$$R = \frac{k}{k + \frac{k\beta}{1-\beta}} = \frac{k}{\frac{k - k\beta + k\beta}{1-\beta}} = 1-\beta$$

- Codul global $C(B_1,B_2,....B_{m-1},C)$ este un corector de ştergeri cu rata 1-β, care poate să corecteze cu probabilitate mare orice până la lungime·β ştergeri



Coduri Tornado

- Procesul de codare constă în sumarea modulo 2 a pachetelor conform grafului "generator"
- Pachetele recepţionate sunt "substituite" în ecuaţiile de control
- Prin înlocuirea variabilelor în ecuaţii de control, se pot reconstitui pachetele pierdute, utilizând adunări modulo 2
- De regulă primele k pachete sosite generează un număr redus de "rezolvări", dar când numărul pachetelor receptionate este mai mare de k, un pachet recepţionat generaează o avalanşă de "rezolvări" permiţând reconstrucţia pachetelor informaţionale

- Deoarece nodurile din graf au un număr redus de vecini (ecuaţii cu puţini termeni) operţiunea de codare şi decodare nu necesită multe operaţii
- Numărul de operaţii pentru obţinerea pachetelor redundante depinde numai de gradul nodului respectiv
- Complexitatea decodării depinde tot de gradul nodurilor şi de "poziţia" pachetelor recepţionate în graf
- Codurile Tornado sunt tot coduri cu lungime finită, deci pentru utilizarea lor ca DF pachetele codate se transmit întreţesute ciclic

- Comparaţie între timpii de codare şi decodare

Timpul de codare, pachete 1K		
Size	Reed-Solomon	Tornado
250 K	4.6 sec.	0.11 sec.
500 K	19 sec.	0.18 sec.
1 MB	93 sec.	0.29 sec.
2 MB	442 sec.	0.57 sec.
4 MB	30 min.	1.01 sec.
8 MB	2 hrs.	1.99 sec.
16 MB	8 hrs.	3.93 sec.

Timpul de decodare		
Size	Reed-Solomon	Tornado
250 K	2.06 sec.	0.18 sec.
500 K	8.4 sec.	0.24 sec.
1 MB	40.5 sec.	0.31 sec.
2 MB	199 sec.	0.44 sec.
4 MB	13 min.	0.74 sec.
8 MB	1 hr.	1.28 sec.
16 MB	4 hrs.	2.27 sec.

[1] John W.Byers, Michael Luby, Michael Mitzenmacher, Ashutosh Rege; "A Digital Fountain Approach to Reliable Distribution of Bulk Data" -Proceedings of the ACM SIGCOMM '98

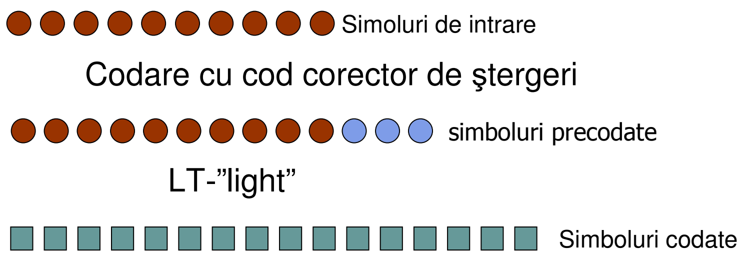
Tornado vs LT

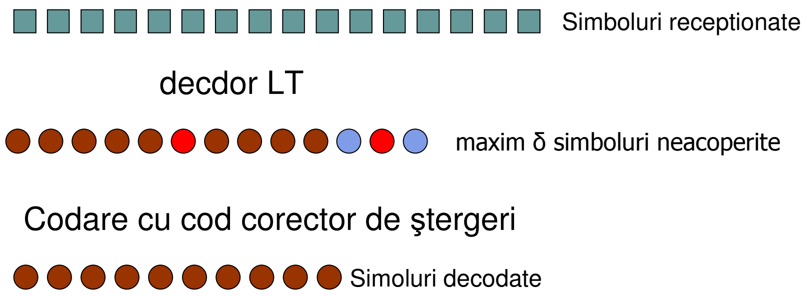
- Atât codurile RS cât şi codurile Tornado sunt coduri sistematice, în schim codurile LT sunt nesistematice
- memoria necesară pentru codarea decodarea codurilor tornado este mult mai mare decăt în cazul codurilor LT
- codurile LT sunt coduri *rateless* iar codurile tornado au rata finită
- Codurile Tornado sunt generate pe baza grafurilor cu grad maxim constant, în schimb codurile LT au gtafuri cu densitate logaritmică

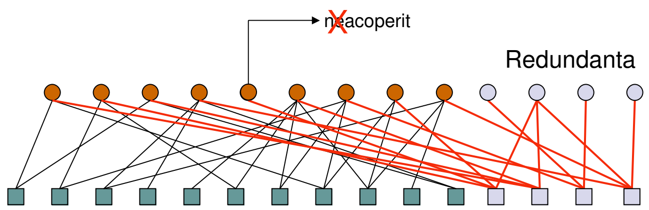
Coduri Raptor(RAPide TORnado)

- Primul cod DF cu timp de codare şi decodare liniară
- au fost inventate in 2000/2001 publicate in 2004
- Se acceptă ca o fracţiune maxim δ din simbolurile de intrare a unui cod LT să nu fie acoperite la terminarea procesului de decodare LT
- Aceste simboluri "pierdute" se pot recupera utilizând un cod corector de ştergeri classic
  - Se cunoaşte numărul maxim de ştergeri posibile
- Implică o precodare cu un cod corector de ştergeri classic

[5] A. Shokrollahi,„Raptor Codes",http://www.cs.huji.ac.il/labs/danss/p2p/resources/raptor.pdf



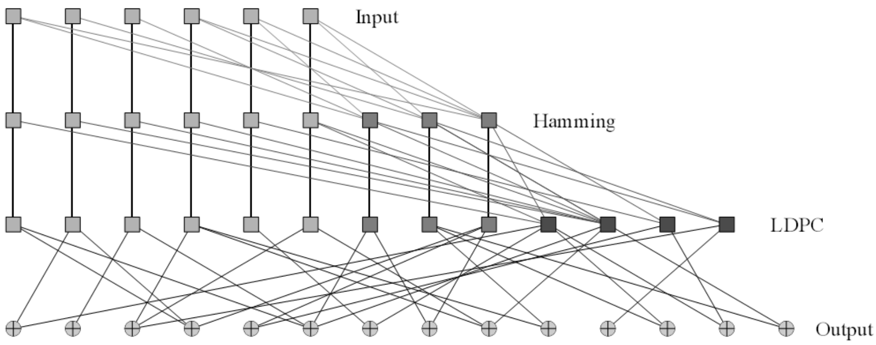




- Dacă precodorul este ales în mod corespunzător, atunci se poate utiliza un cod LT cu gradul mediu constant, care asigură timp de codare liniară
- Un cod Raptor (k,C, Ω(d))este definit de numărul de pachete informaţionale k, codul corector de ştergeri C şi distribuţia gradurilor al codului LT Ω(d)

Coduri Raptor

- Parametrii principali de performanţă ai codului Raptor sunt definite după cum urmează:
  - Spațiul de memorie: Codurile Raptor necesită spațiu de stocare pentru simbolurile intermediare, Consumul de spațiu al codurilor Raptor este k/*R*, unde *R* este rata pre-codului.
  - Overhead: Overheadul este o funcție a algoritmului de decodare folosit, şi este definit ca numărul de simboluri de ieşire pe care trebuie să aibă decodorul pentru a recupera cu probabilitate mare simbolurile de intrare. Un overhead de $1+\varepsilon$ înseamnă că trebuie recepționate $k(1+\varepsilon)$ simboluri de ieşire pentru a asigura o decodare cu success cu o probabilitate mare.
  - Costul: Costul procesului de codare şi de decodare.



Probleme legate de DF

- Implementarea oricărei aplicaţii, se poate face numai cu acordul DF inc.
- "…but networking people do not want to deal with developing codes." [1]
- datorită drepturilor de autor aplicaţiile cu DF sunt foarte puţin cunoscute

[1] Michael Mitzenmacher-Digital Fountains: Applications and Related Issues