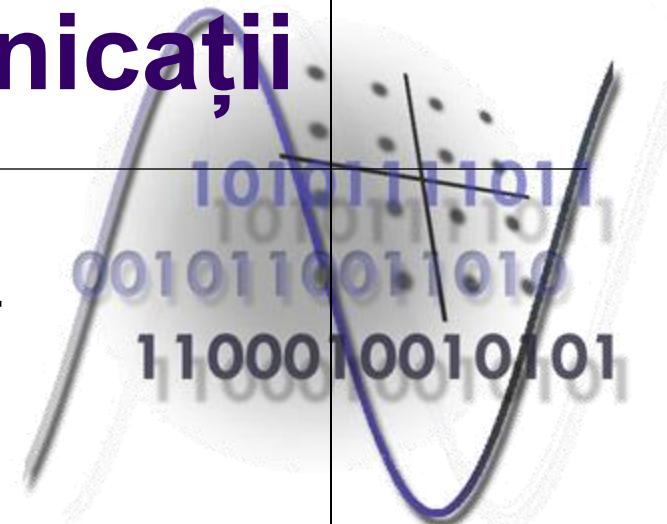
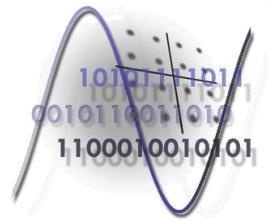


Tehnici avansate de codare și control al fluxului de date în rețelele de telecomunicații

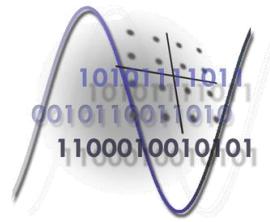
TACCFD Curs 1





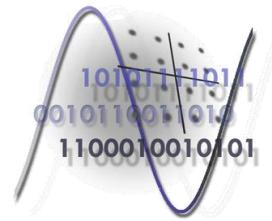
Cuprins

- Tehnici de codare de tip Digital Fountain (DF).
- Coduri de tip Tornado și de Raptor.
- Implementarea conceptului de DF cu ajutorul codurilor cu rată finită.
- Tehnici de codare de tip Network Coding.
- Generarea și optimizarea rețelei de codare Network Coding.
- Comunicații de tip swarm.
- Coduri liniare de tip Network Coding utilizate sisteme de tip swarm.
- Tehnici de codare NC utilizate în rețele wireless de tip “mesh”.
- Tehnici de codare distribuită și NC utilizate în rețele celulare cooperative.
- Tehnici de optimizare bazate pe Teoria Jocurilor
- Tehnici de optimizare bazate pe Algoritmi Genetici



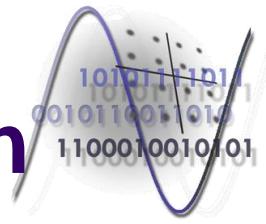
Modul de examinare și atribuire a notei

- Formula de calcul a notei:
 - $N=0,6\text{Examen}+0,4\text{Miniproiect}$



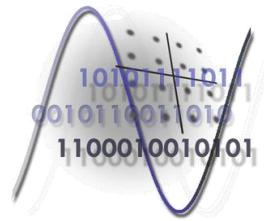
Tehnici de codare de tip Digital Fountain

- Fundamente teoretice
- Algoritmi de codare și decodare



Tehnici de codare de tip Digital Fountain

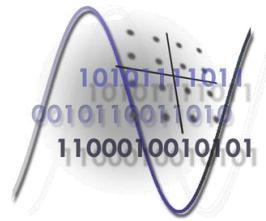
- Conceptul DF
- DF cu coduri corectoare de ștergeri
- Coduri DF lineare, aleatoare
- Coduri LT



Proprietățile unui protocol ideal[1]

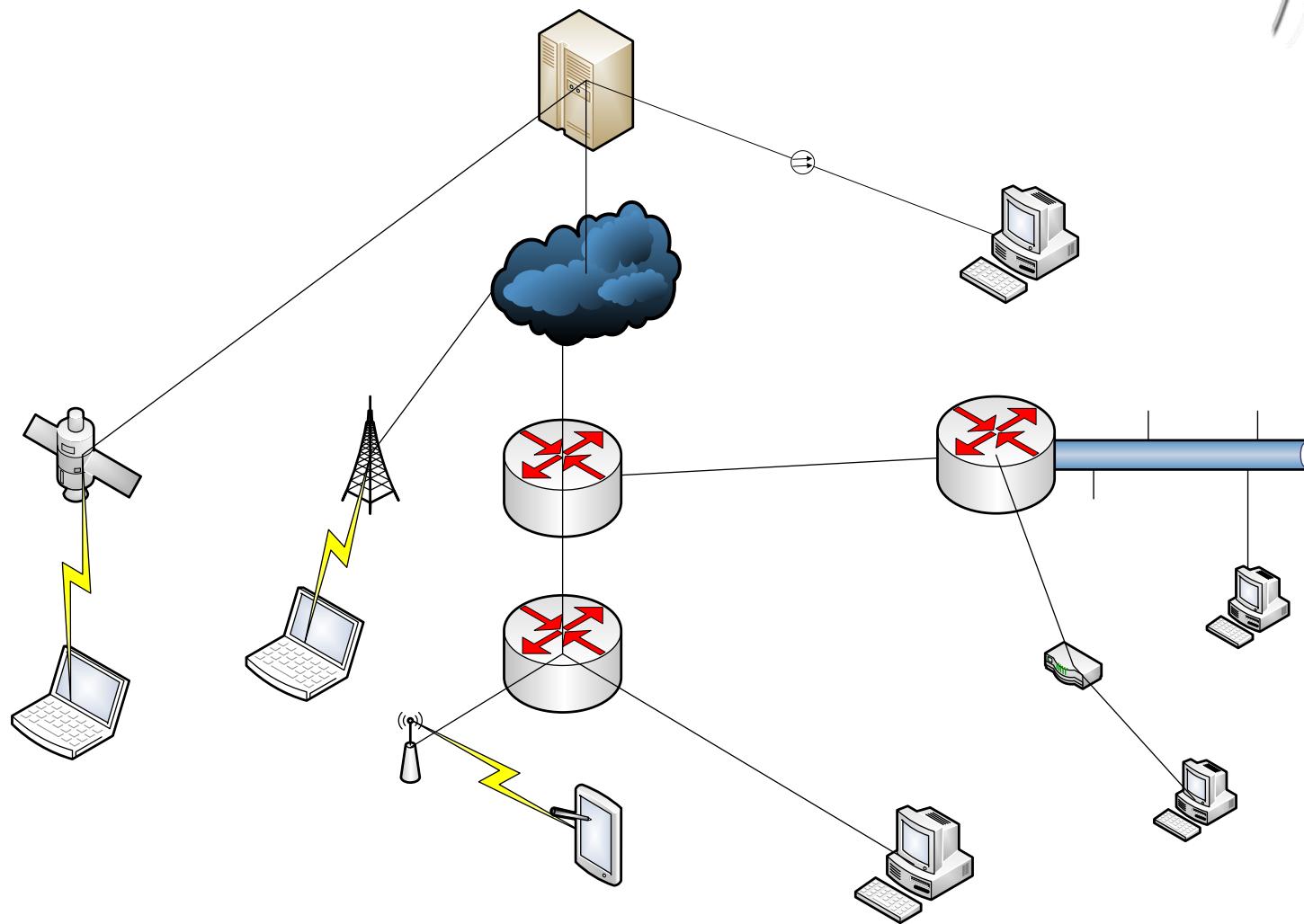
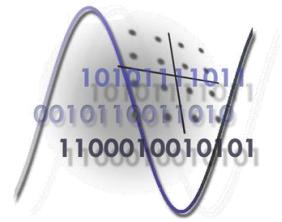
- **Fiabil** – este garantat că datele ajung în întregime
- **Eficient** - atât numărul de pachete necesare pentru reconstrucția informației cât și timpul necesar reconstrucției din pachetele recepționate trebuie să fie minim
- **Tolerant** – protocolul trebuie să suporte o populație de receptori eterogena (deferite debite, rata de pierderi etc.)
- **La cerere** – Utilizatorii pot inițializa sesiunile de comunicări în momente aleatoare, pot întrerupe respectiv relua sesiunile în orice moment de timp

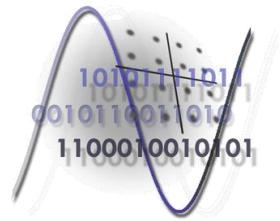
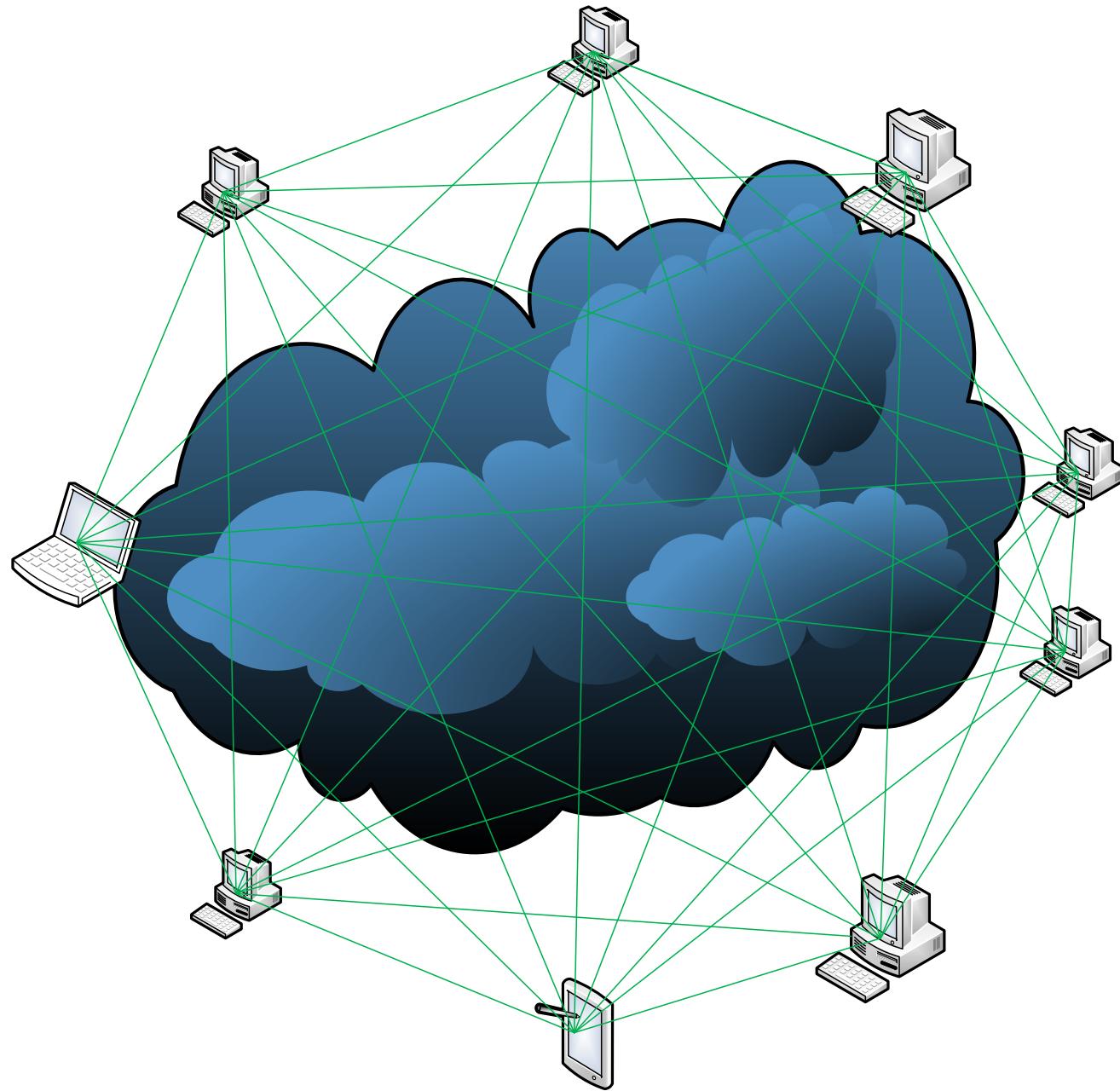
[1] John W.Byers, Michael Luby, Michael Mitzenmacher, Ashutosh Rege; “A Digital Fountain Approach to Reliable Distribution of Bulk Data” -Proceedings of the ACM SIGCOMM '98
22 februarie 2021

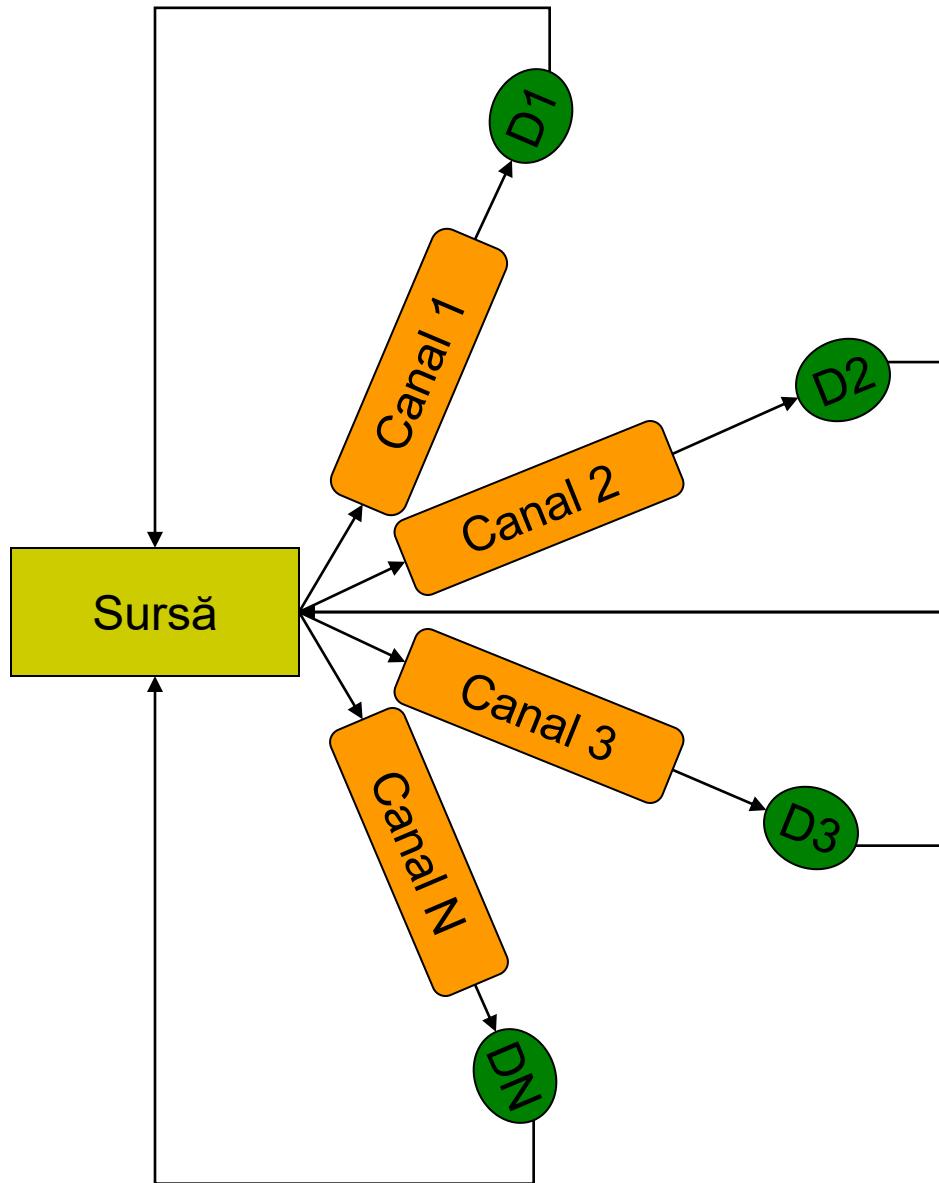
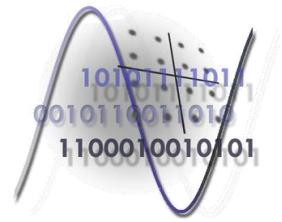


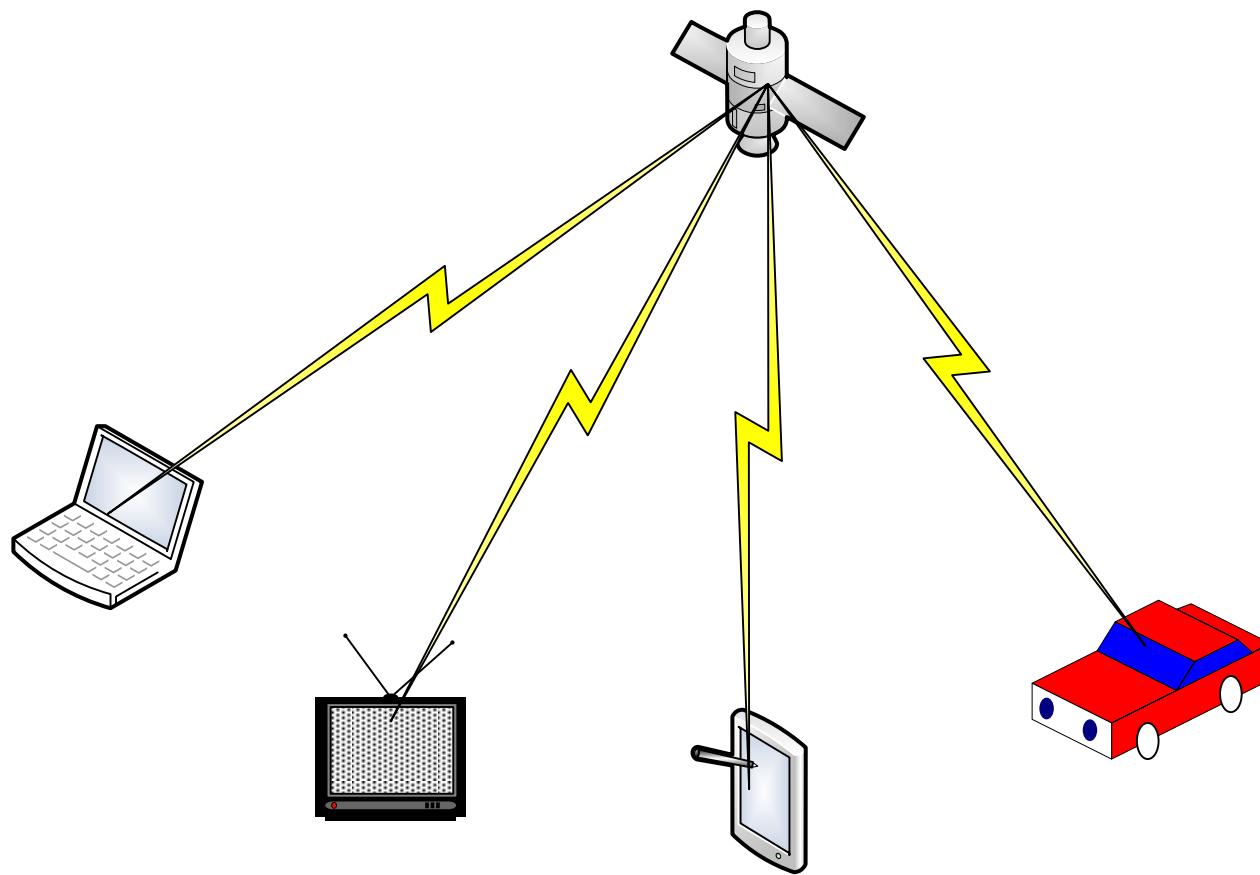
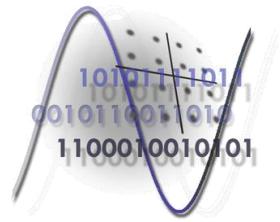
Fiabilitatea pe internet

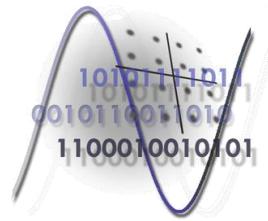
- Pentru asigurarea fiabilități se utilizează
 - ARQ (pe canale cu Feedback)
 - Avantaje:
 - Asigură fiabilitatea transmisiei și la condiții severe de canal
 - Dezavantaje:
 - Resurse utilizate
 - Necesită legătură bidirectională
 - Codarea canalului
 - FEC
 - Nu necesită (teoretic) feedback
 - Poate să corecteze un număr predefinit de erori (depinde de rata)



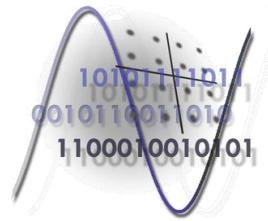








- Numărul cererilor de retransmisie se poate reduce utilizând coduri corectoare de ștergeri (RS)
 - La k pachete de informație se adună $n-k$ pachete de control
 - Codul RS poate corecta până la $n-k$ ștergeri din cuvântul de cod format din n pachete
- Deoarece fiecare legătură are propria probabilitate de pierdere a pachetelor rata codului utilizat se determină pe baza legăturii cu probabilitatea cea mai mare de pierderi
 - Pe legăturile bune se transmit prea multe pachete redundante

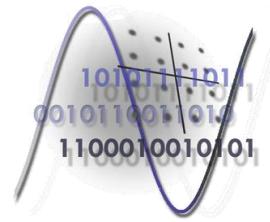


- Avantajele codului RS

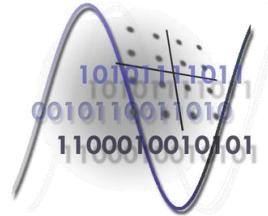
- Poate să corecteze cele mai multe ștergeri la o lungime și o rată dată

- Dezavantajele codului RS

- Lungimea pachetelor foarte mică
- Algoritmi de codare și decodare complicate
- Imposibilitatea adaptării ratei

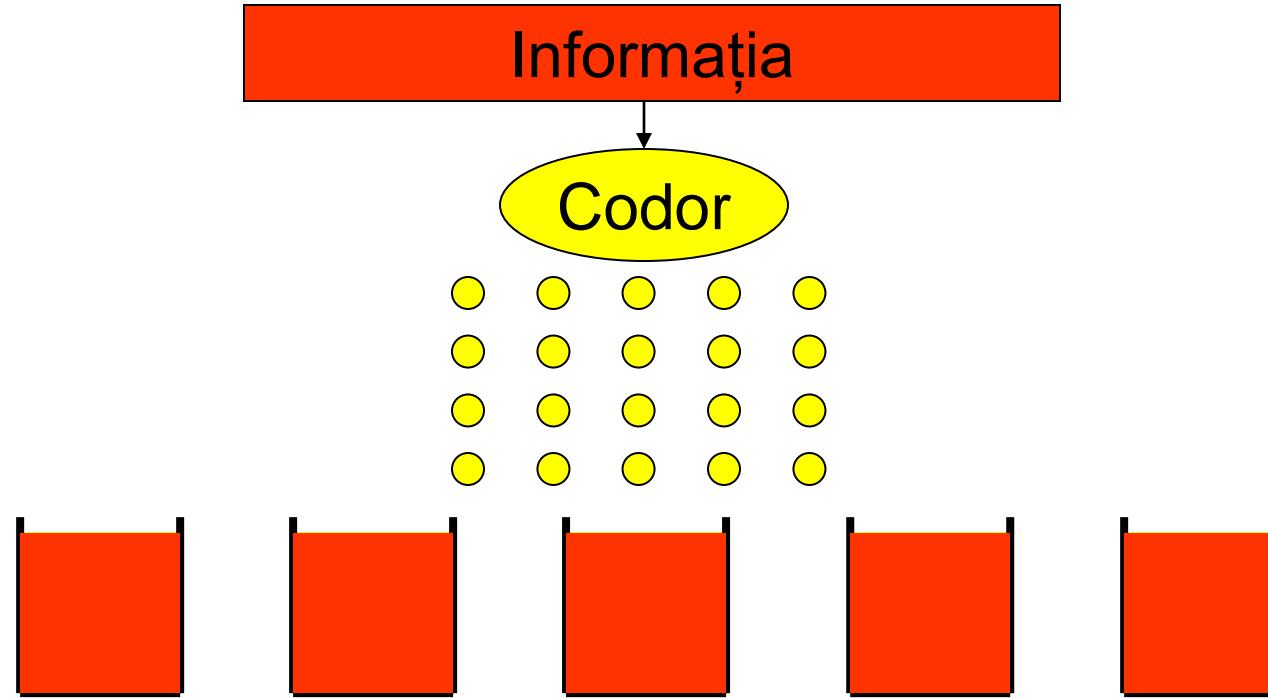
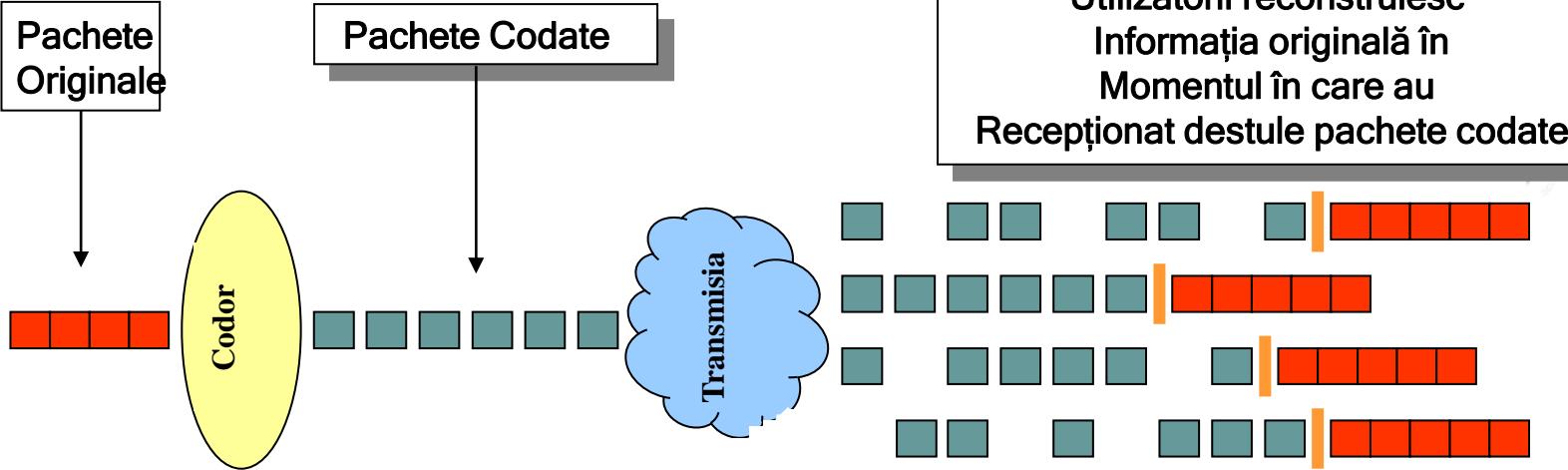


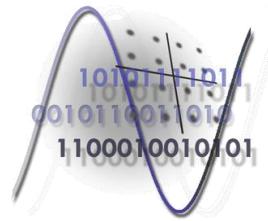
- Restricții (dezavantajele) utilizării codurilor corectoare de ștergeri clasice
 - Receptorul trebuie să cunoască exact poziția pachetului pierdut
 - Pachetele care ajung trebuie să ajungă în ordinea transmiterii, sau trebuie să fie numerotate
 - Lungimea pachetelor trebuie să fie foarte mică
 - Este complicat adaptarea capacitații de corecție la caracteristicile fiecărui legături
- Probleme la implementarea protocolelor eficiente și *la cere*



Conceptul Digital Fountain

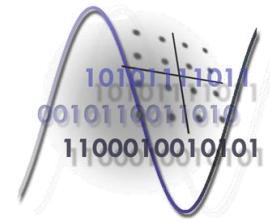
- Posibilitatea generării dintr-un număr k finit de pachete informaționale un număr infinit de pachete codate
- Receptorul din oricare set de $k+\varepsilon$ pachete codate informaționale poate reconstrui cele k pachete informaționale



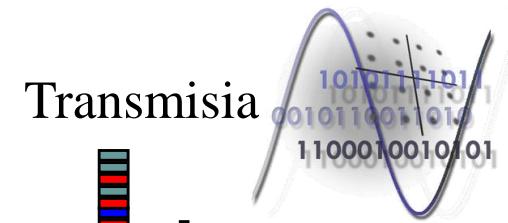
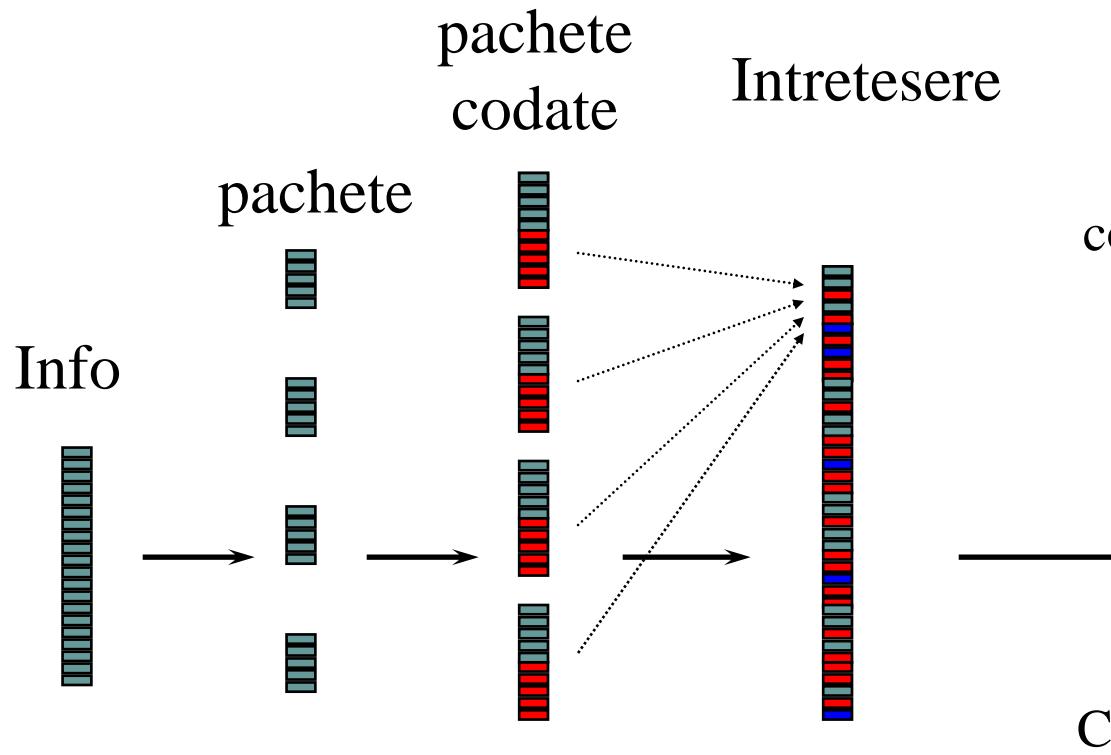


- Nu necesită feedback
- Timpul de recepție pentru fiecare utilizator depinde de calitatea legăturii
- Sursa nu trebuie să gestioneze separat fiecare conexiune
- Rata se adaptează automat la caracteristicile legăturii

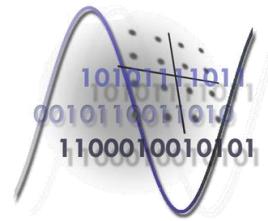
DF cu coduri corectoare de ștergeri



- Utilizând coduri RS
 - Informația originală se împarte în k pachete (simboluri)
 - Se alege un cod cu rata suficient de mică și se generează cele n simboluri codate
 - Sursa transmite repetat simbolurile codate
 - Când un utilizator a recepționat cel puțin k simboluri independente poate să reconstruiască informația originală
 - Fiecare utilizator poate să înceapă receptia pachetelor în momente aleatoare
 - Fiind că pachetele codate se transmit repetitiv, utilizatorii nu trebuie să recepționeze cele k pachete independente pe durata unei cicluri
 - Posibilitatea de a recepționa de mai multe ori același pachet

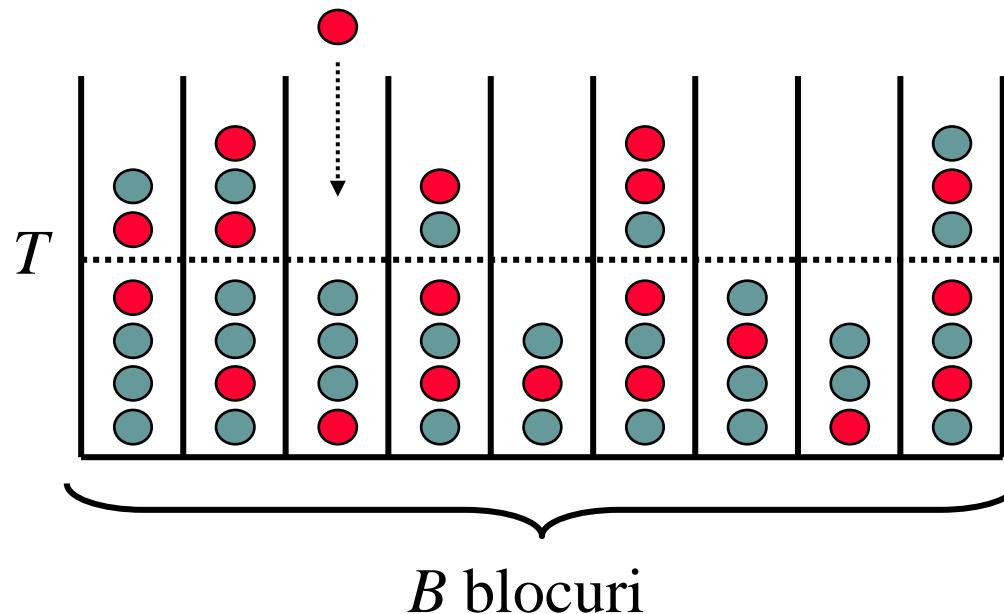


[1] John W.Byers, Michael Luby, Michael Mitzenmacher, Ashutosh Rege; “A Digital Fountain Approach to Reliable Distribution of Bulk Data” -Proceedings of the ACM SIGCOMM '98
22 februarie 2021

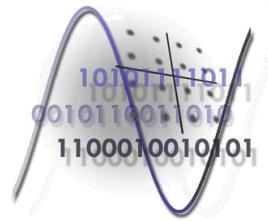


Dezavantajele Întreținerii ciclice

- Așteaptă după ultimele blocuri

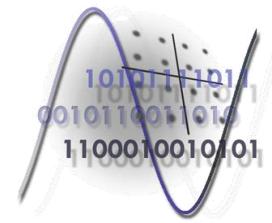


[1] John W.Byers, Michael Luby, Michael Mitzenmacher, Ashutosh Rege; “A Digital Fountain Approach to Reliable Distribution of Bulk Data” -Proceedings of the ACM SIGCOMM '98
22 februarie 2021



Limitările utilizării codurilor RS

- Datorită complexității de implementare a codorului și a decodorului dimensiunea pachetelor în practică nu poate să fie mai mare de 16biți (**operații în $GF(2^x)$**)
- Datorită întrețeserii ciclice poate să crească numărul pachetelor redundante
- Complexitatea decodării depind de n, și k (cod “*puternic*” timp de decodare mare)

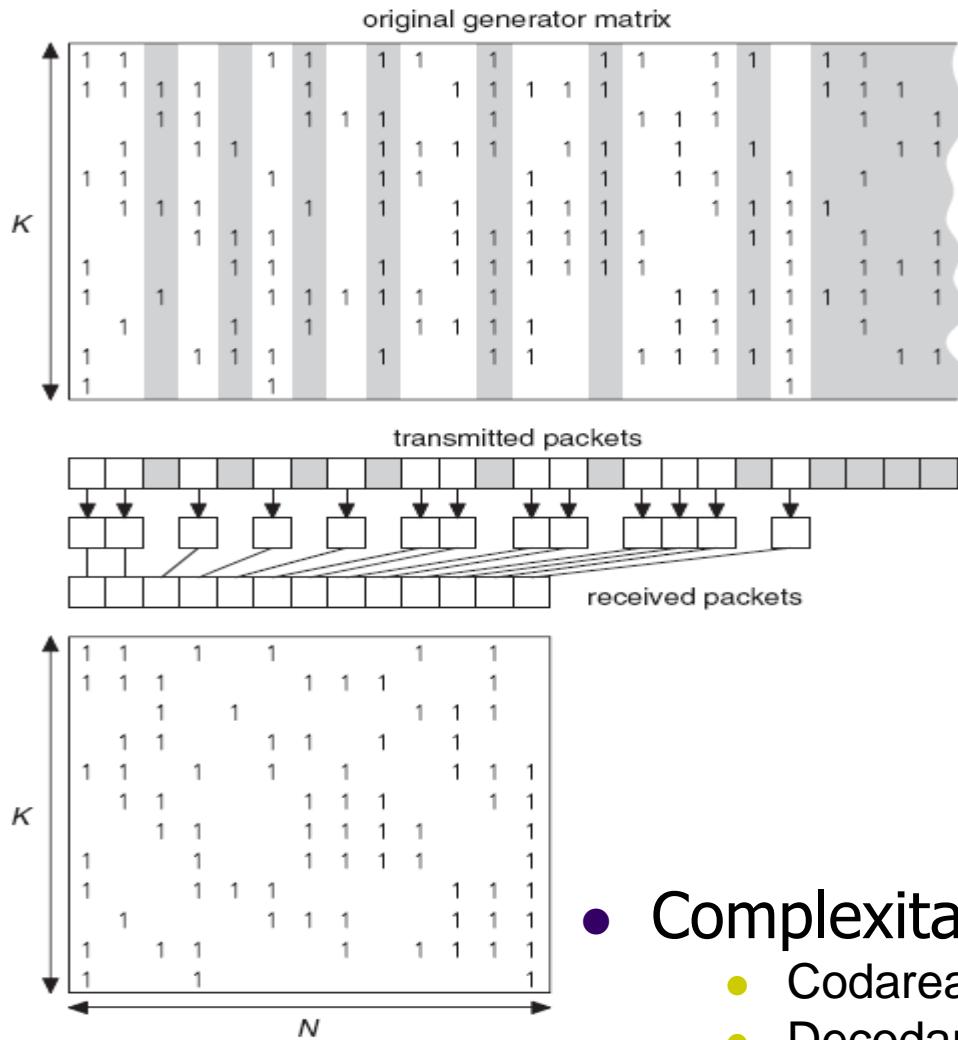


Coduri DF lineare, aleatoare

- Informația este împărțită în K pachete $s_1, s_2, \dots s_k$
- La fiecare perioadă de timp (marcat cu n), codorul generează k biți aleatorii $\{G_{kn}\}$
- Pachetul codat în momentul E_n se obține adunând modulo 2 pachetele pentru care elementele corespunzătoare din G_{kn} este 1, adică:

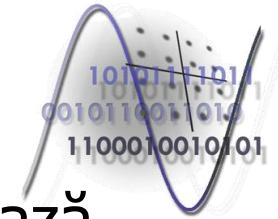
$$E_n = \sum_{k=1}^K s_k G_{kn}$$

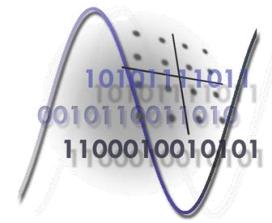
- La fiecare pachet codat se adaugă vectorul G_{kn}



- Receptorul stochează pachetele captureate și construiește matricea de decodare
 - Dacă s-a recepționat $N=K$ pachete probabilitatea ca matricea de decodare să fie inversabilă este 0.289
 - Dacă $N>K$ informația poate fi decodată dacă și numai dacă există în matricea de decodare o sub-matrice cu dimensiuni $K \times K$ inversabilă

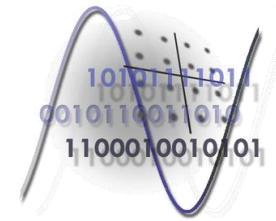
Complexitatea implementării:
 $O(k^2)$
Complexitatea $O(k^3)$
Boggs – Commun. Vol. 152, No 6, Dec 2005.





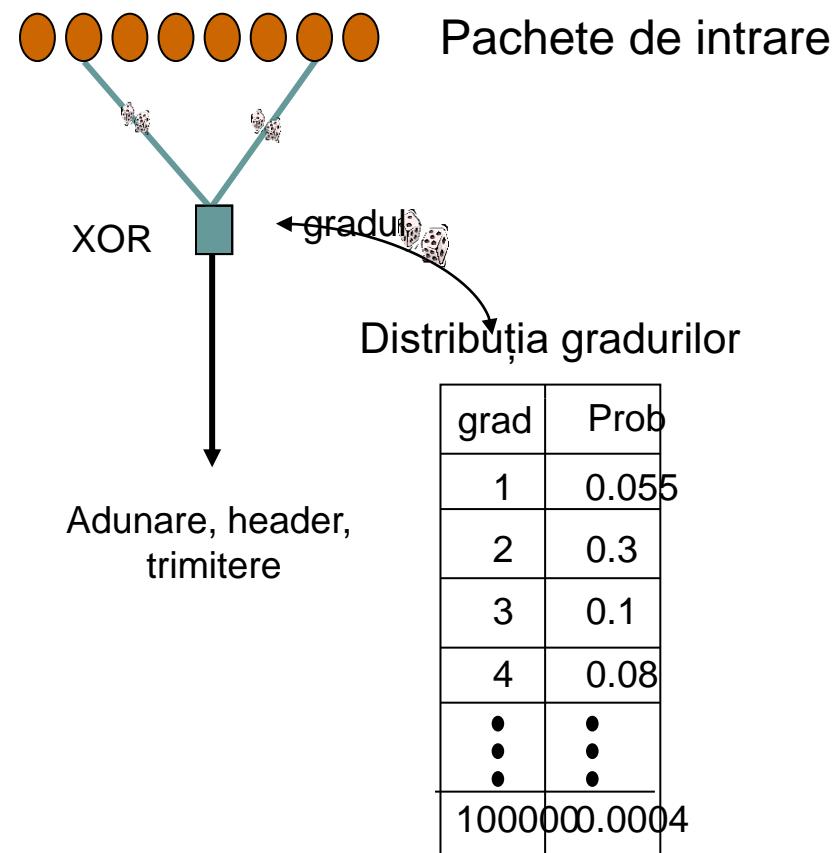
Codurile Luby-Transform (LT)

- Codurile LT reprezintă prima realizare practică a codurilor rateless; nu trebuie definită o rată fixă înainte de codare
- Se poate construi un flux infinit de pachete codate din pachetele informaționale



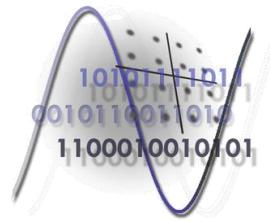
Principiul de codarea LT [3]

- Pentru pachetul codat care urmează să fie generat se alege aleator un grad d , conform unei distribuții predefinite
- Se aleg aleator d pachete din multimea pachetelor informaționale
- Pachetul codat se obține adunând modulo doi pachete selecționate la pasul anterior

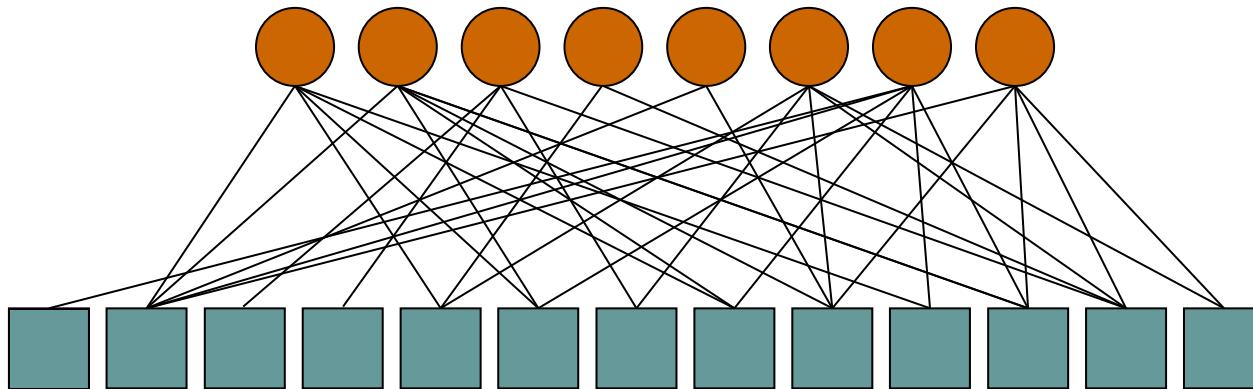


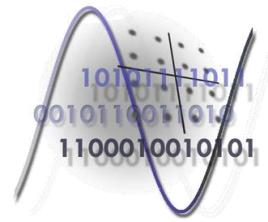
[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
22 februarie 2021 TACCFD - Curs 1



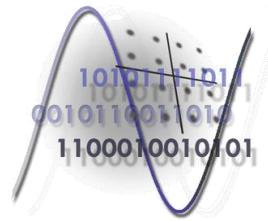
Graful asociat codului



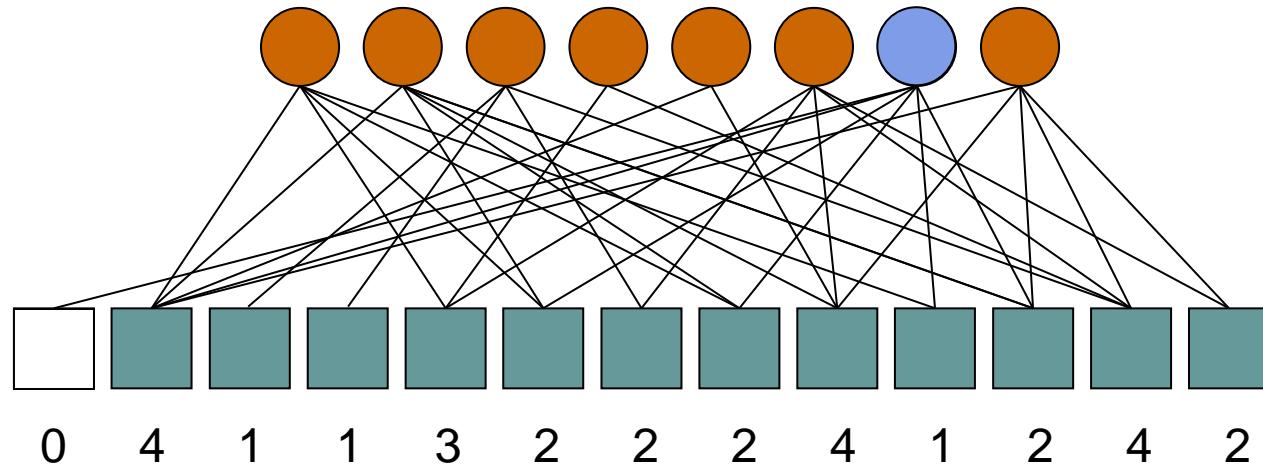


Principiul de decodare

- Regula de decodare
 - Dacă există minim un simbol codat care are un singur vecin, atunci valoarea vecinului respectiv este o copie a simbolului codat.
 - Valoarea simbolului de intrare recuperat va fi adunat modulo 2 cu toate simbolurile codate rămase care au ca și vecin acel simbol de intrare
 - gradul simbolurilor codate, la care a fost adunat simbolul informațional recuperat este redus cu unu, și acest simbol de intrare este eliminat din lista vecinilor

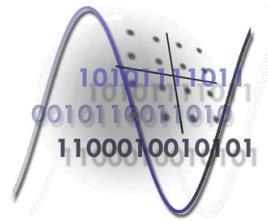


Decodare LT

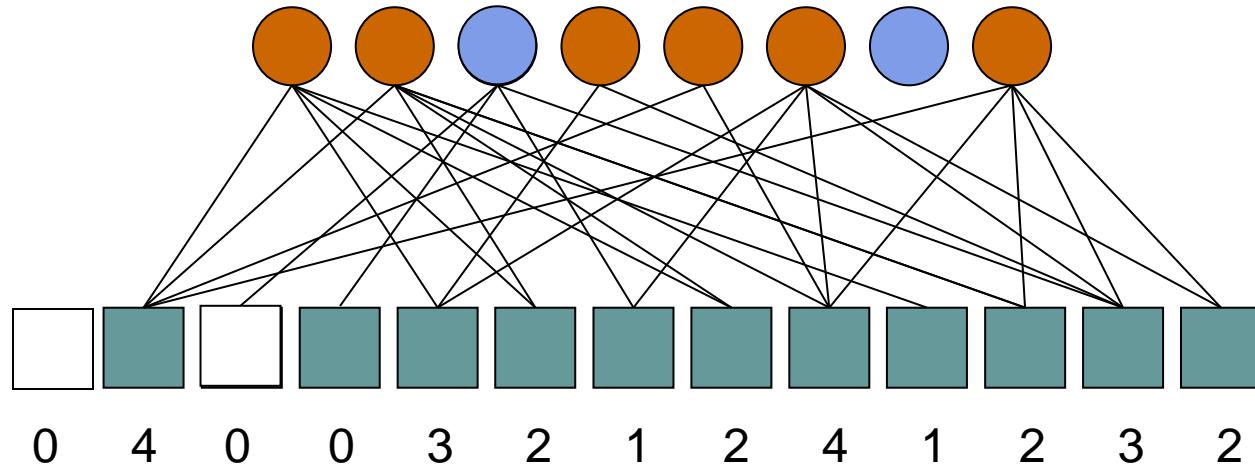


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
22 februarie 2021
TACCFD - Curs 1

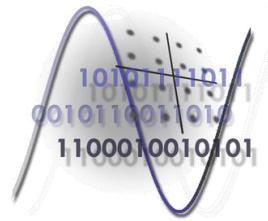


Decodare LT

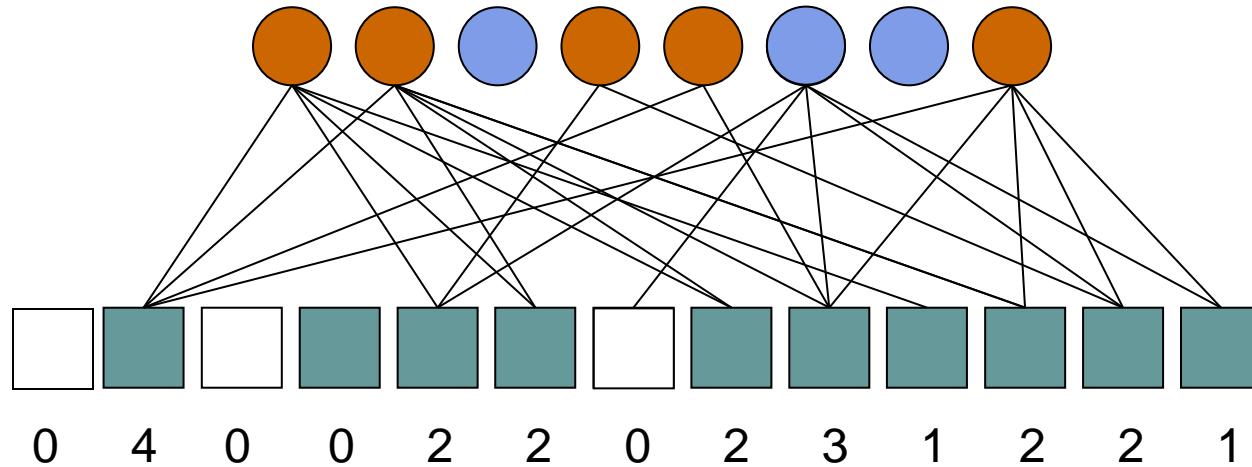


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
22 februarie 2021 *TACCFD - Curs 1*

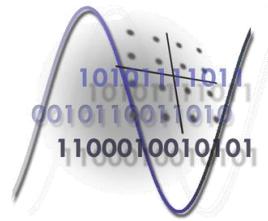


Decodare LT

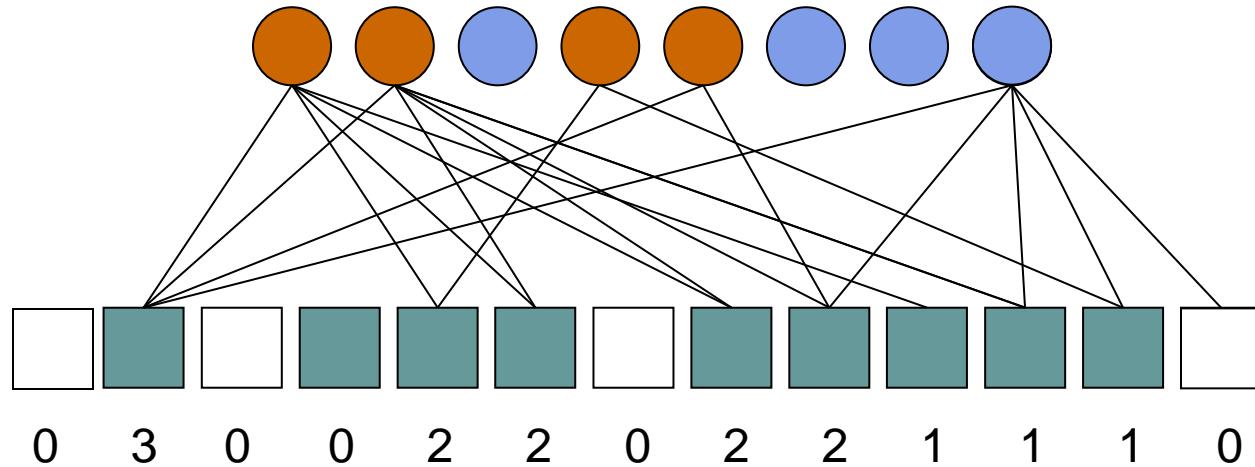


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
22 februarie 2021 *TACCFD - Curs 1*

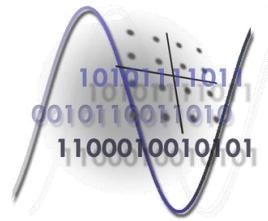


Decodare LT

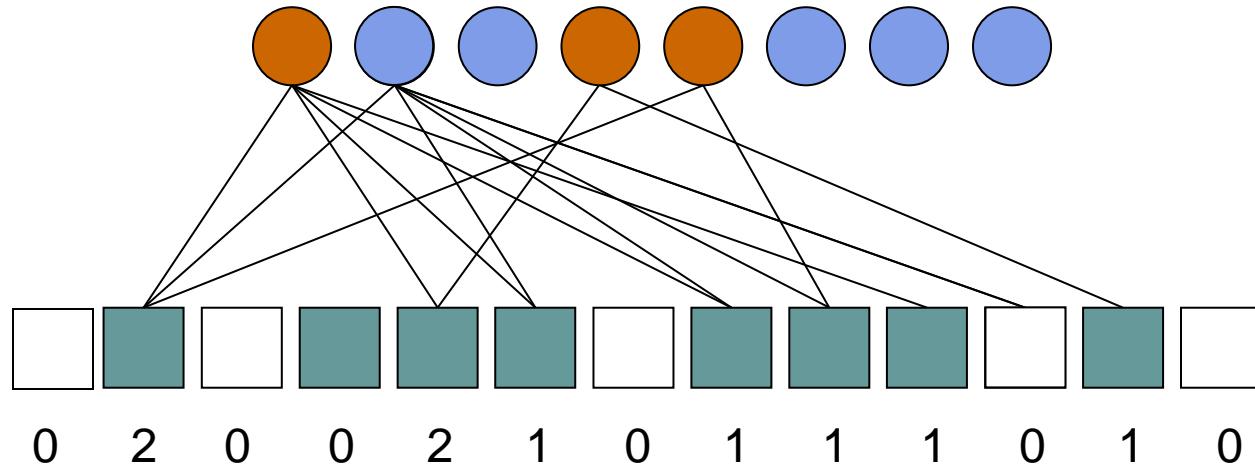


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
22 februarie 2021 *TACCFD - Curs 1*

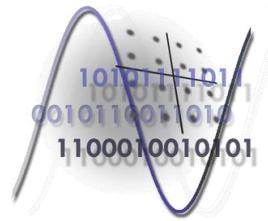


Decodare LT

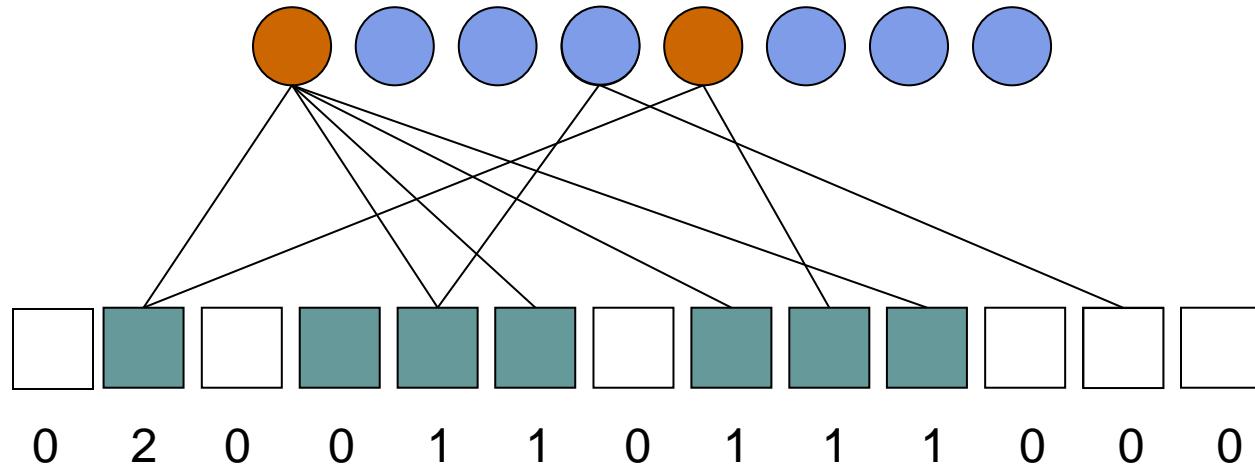


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
22 februarie 2021 *TACCFD - Curs 1*

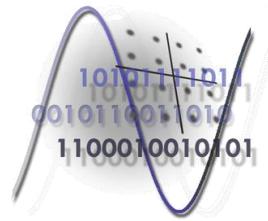


Decodare LT

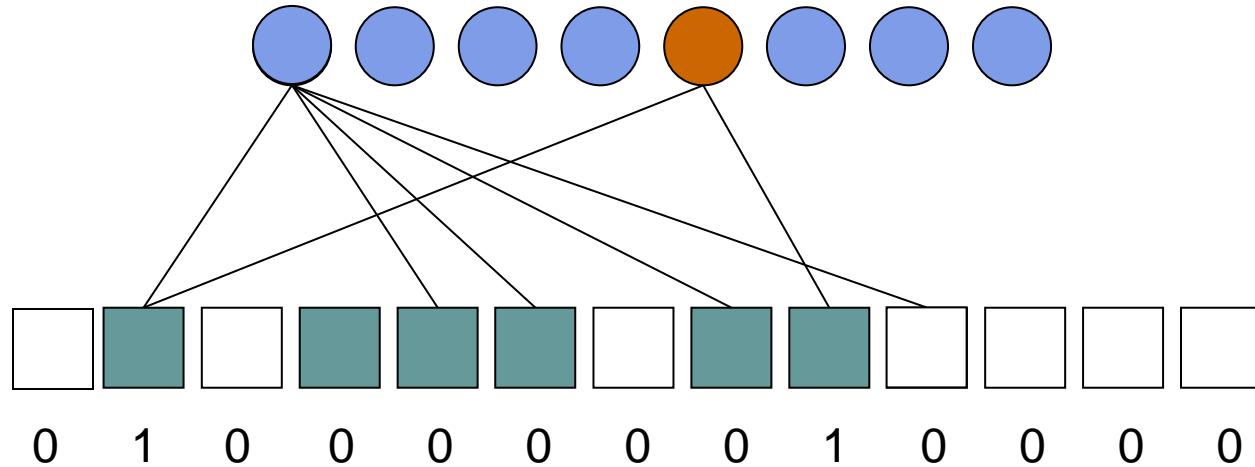


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
22 februarie 2021 *TACCFD - Curs 1*

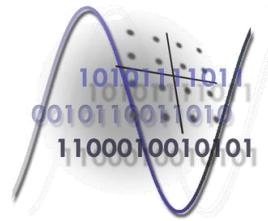


Decodare LT

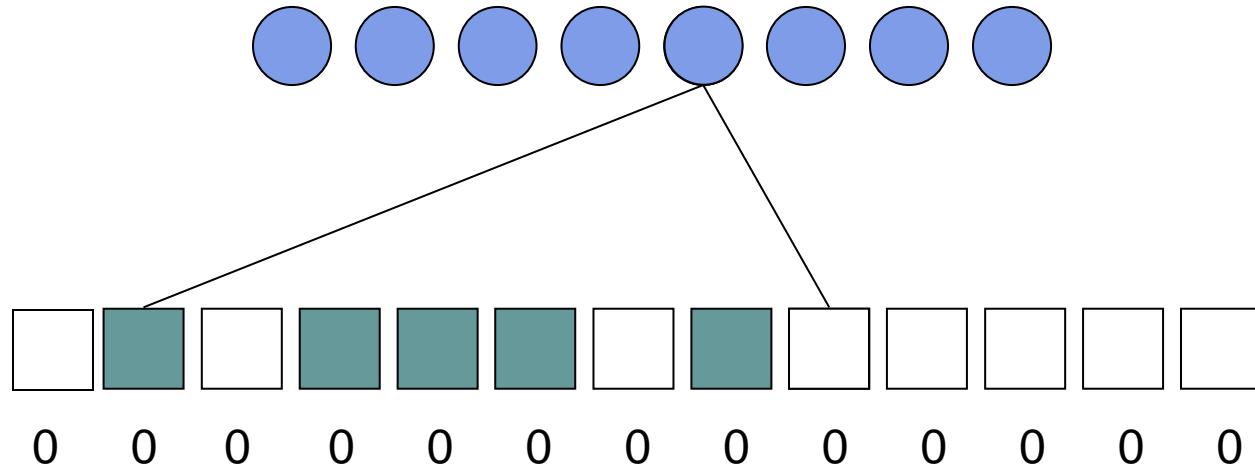


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
22 februarie 2021 *TACCFD - Curs 1*



Decodare LT

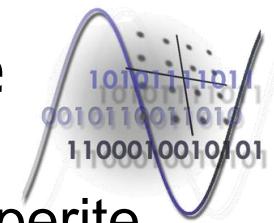


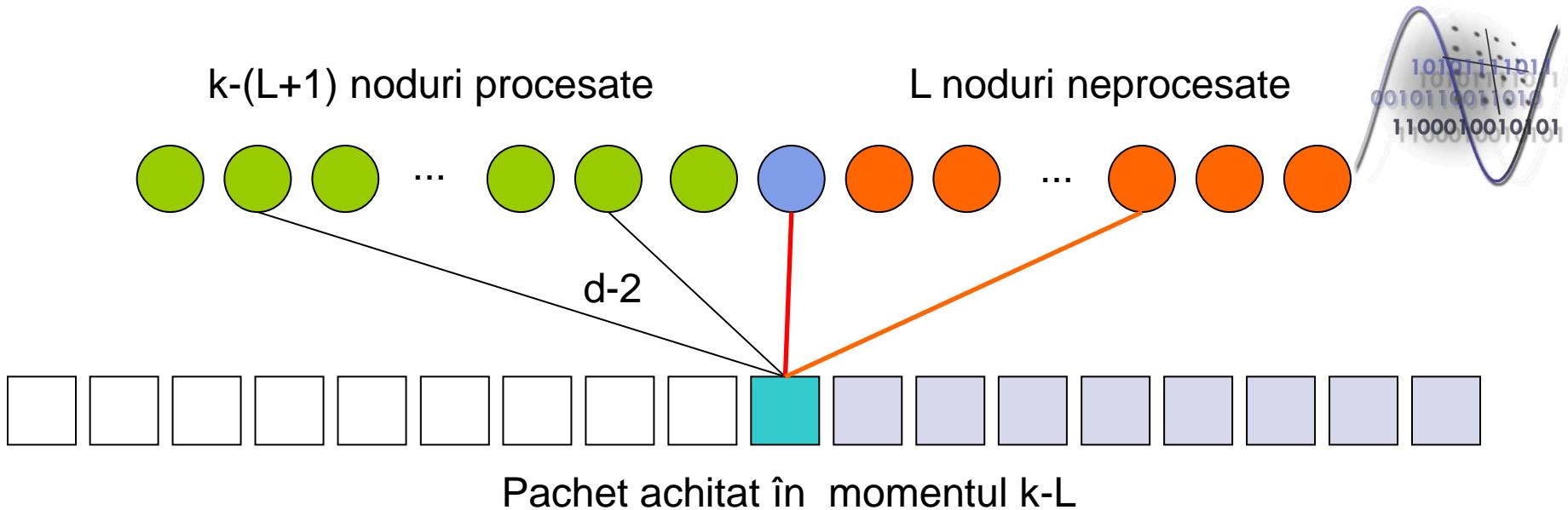
[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
22 februarie 2021 *TACCFD - Curs 1*

- Pt studierea distribuțiilor se reformulează procesul de decodare:

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





- Probabilitatea $q(d,L)$ ca un pachet cu gradul inițial d să fie achitat când mai sunt L pachete informative neprocesate este:

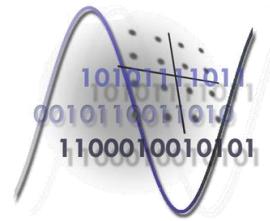
$$q(1,k)=1$$

pentru $d = 2, \dots, k$ și $L = k - d + 1, \dots, 1$

$$q(d,L) = \frac{\binom{k-(L+1)}{d-2}}{\binom{k-1}{d}} L = \frac{d(d-1) \cdot L \prod_{j=0}^{d-3} k - (L+1) - j}{\prod_{j=0}^{d-1} k - j}$$

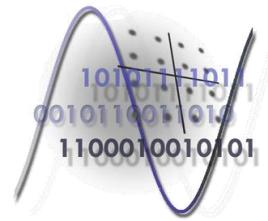
pentru celelalte valori d și L

$$q(d,L) = 0;$$



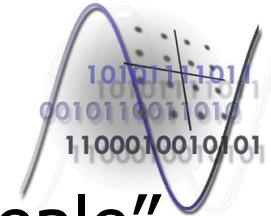
- Probabilitatea $r(d,L)$ este probabilitatea ca un simbol codat să aibă gradul d , și să fie achitat când mai sunt L simboluri de intrare neprocesate
 - $r(d,L) = p(d)q(d,L)$
- Probabilitatea $r(L)$ ca un pachet să fie achitat când mai sunt L pachete informative neprocesate este:

$$r(L) = \sum_d r(d,L) = \sum_{d=2}^k p(d) \frac{d(d-1) \cdot L \prod_{j=0}^{d-3} k - (L+1) - j}{\prod_{j=0}^{d-1} k - j}$$



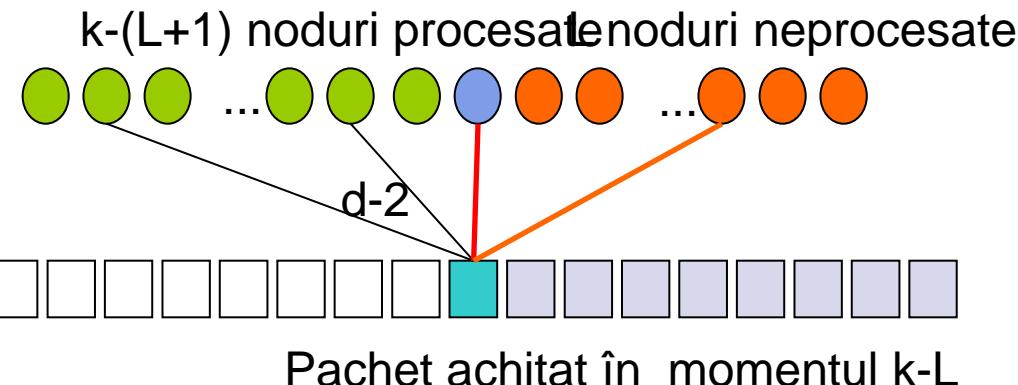
Distribuția *Soliton* ideală

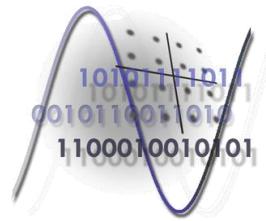
- Cerințele impuse unei distribuții de graduri sunt:
 - Un număr mediu de simboluri codate cât mai mic posibil pentru a asigura succesul procesului LT.
 - Gradul mediu al simbolurilor cât mai mic posibil. Gradul mediu definește numărul operațiilor de simbol necesare pentru generarea unui simbol codat, iar $k^*(\text{grad mediu})$ este numărul operațiilor necesare pentru recuperarea completă a datelor.



Distribuția *Soliton* ideală

- O proprietate elementară a unei distribuții "ideale" este ca la procesul de decodare, la procesarea unui simbol informational la riplu să fie adăugat un simbol acoperit.
- Asta asigură că dimensiunea riplului să nu fie niciodată prea mică sau prea mare.
- $r(L)$ –este probabilitatea ca la riplu să fie adunat un singur simbol la procesarea simbolului $k-(L+1)$





Distribuția *Soliton* ideală

- Tânărând cont că:

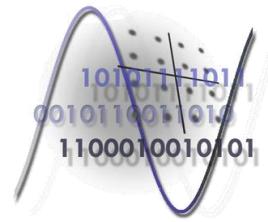
$$r(L) = \sum_d r(d, L) = \sum_{d=2}^k p(d) \frac{d(d-1) \cdot L \prod_{j=0}^{d-3} k - (L+1) - j}{\prod_{j=0}^{d-1} k - j}$$

și

$$\sum_{d=2}^k \frac{L \prod_{j=0}^{d-3} k - (L+1) - j}{\prod_{j=0}^{d-1} k - j} = 1 \quad pt\ orice\ L > 1$$

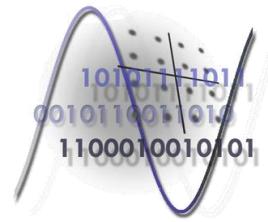
Rezultă distribuția Soliton ideală:

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



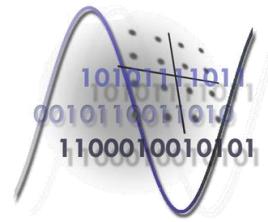
Distribuția *Soliton* ideală

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



Distribuția Soliton Robust

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



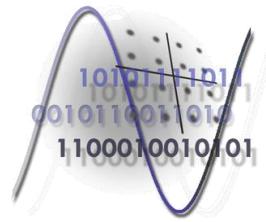
Distribuția Soliton Robust

- Pentru un compromis se acceptă că din K pachete receptionate, decodorul LT nu reușește să determine informațiile 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:

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



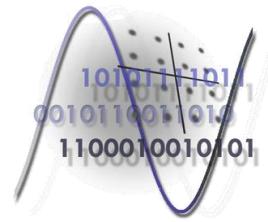
Distribuția Soliton Robust

- Se alege lungimea riplului dorit ca fiind:

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

- Se definește $\tau(d)$ ca fiind

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



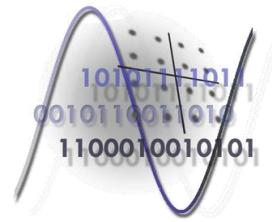
Distribuția Soliton Robust

- Se adună distribuția ideală $p(d)$ la $\tau(d)$, normalizând această sumă cu β se obține distribuția robustă $\mu(d)$

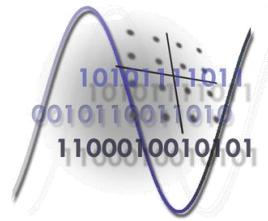
$$\beta = \sum_{d=1}^k p(d) + \tau(d)$$

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

- Valoarea medie a gradurilor este $c/n(k)$
- Overageheadul necesar este proporțional cu K



- 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



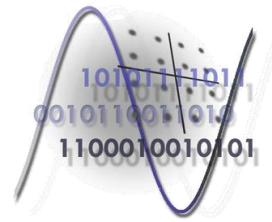
Probleme legate de DF

- Implementarea oricărei aplicații, se poate face numai cu acordul DF inc.

Codurile Luby- Transform (LT)

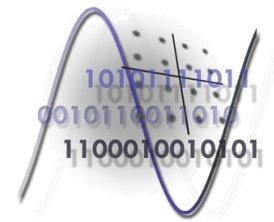
TACCFDRT Curs 2





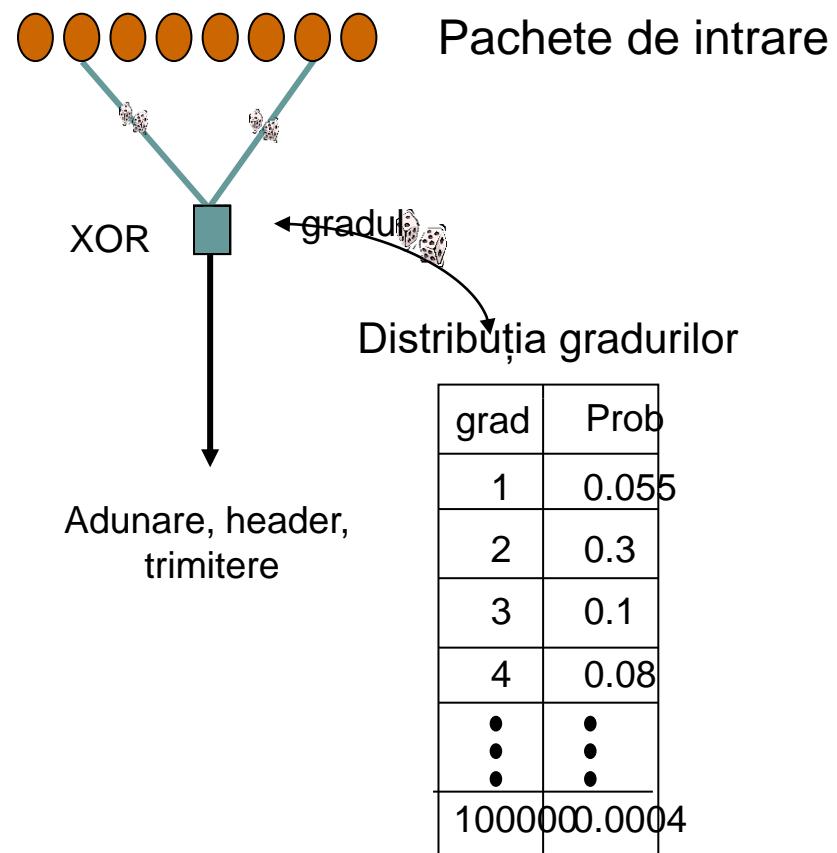
Codurile Luby-Transform (LT)

- Codurile LT reprezintă prima realizare practică a codurilor rateless; nu trebuie definită o rată fixă înainte de codare
- Se poate construi un flux infinit de pachete codate din pachetele informative



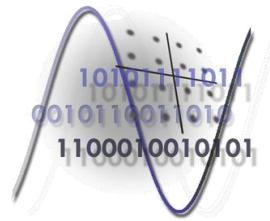
Principiul de codarea LT [3]

- Pentru pachetul codat care urmează să fie generat se alege aleator un grad d , conform unei distribuții predefinite
- Se aleg aleator d pachete din multimea pachetelor informaționale
- Pachetul codat se obține adunând modulo doi pachete selecționate la pasul anterior

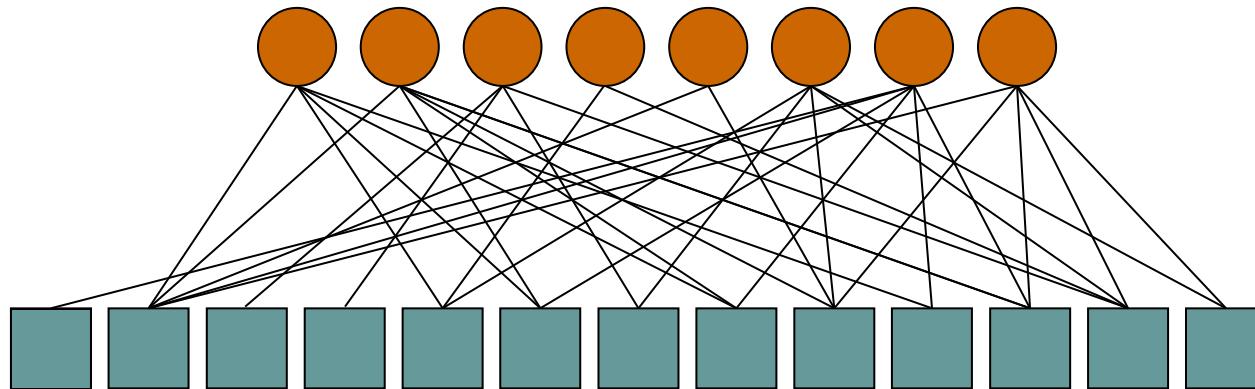


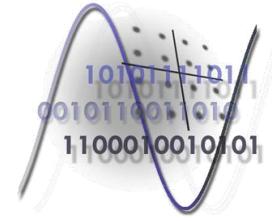
[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
8 Martie 2024
TACCFDRT - Curs 2



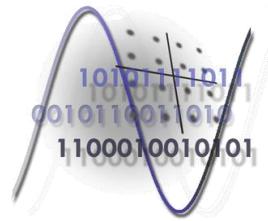
Graful asociat codului



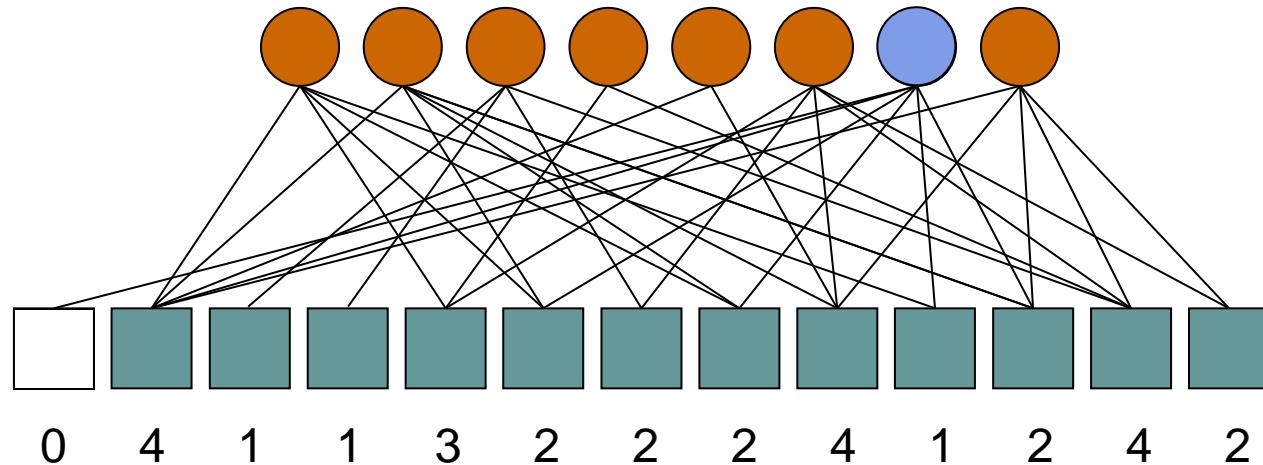


Principiul de decodare

- Regula de decodare
 - Dacă există minim un simbol codat care are un singur vecin, atunci valoarea vecinului respectiv este o copie a simbolului codat.
 - Valoarea simbolului de intrare recuperat va fi adunat modulo 2 cu toate simbolurile codate rămase care au ca și vecin acel simbol de intrare
 - gradul simbolurilor codate, la care a fost adunat simbolul informațional recuperat este redus cu unu, și acest simbol de intrare este eliminat din lista vecinilor

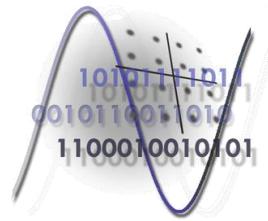


Decodare LT

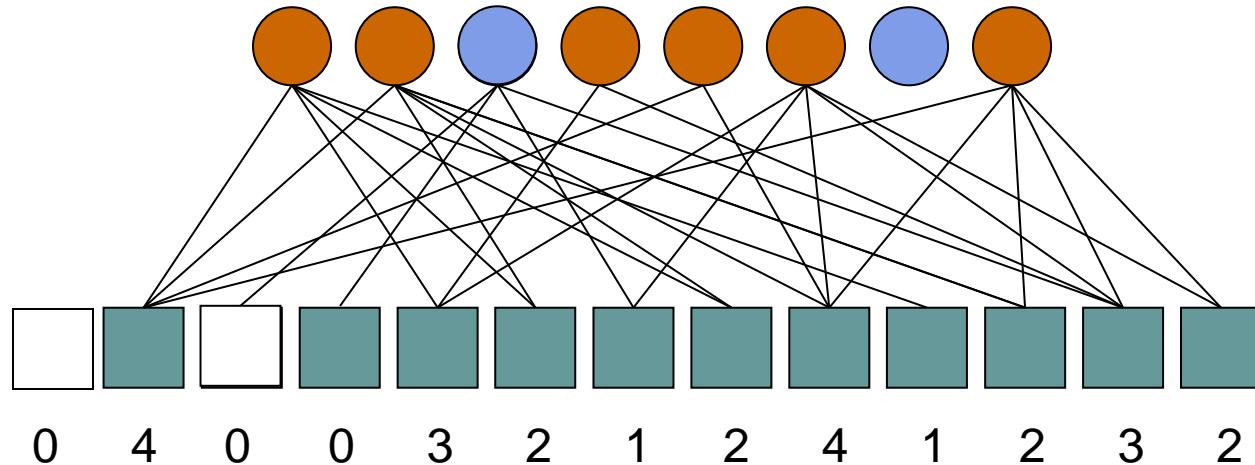


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
8 Mai 2024
TACCFDRT - Curs 2

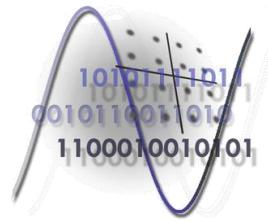


Decodare LT

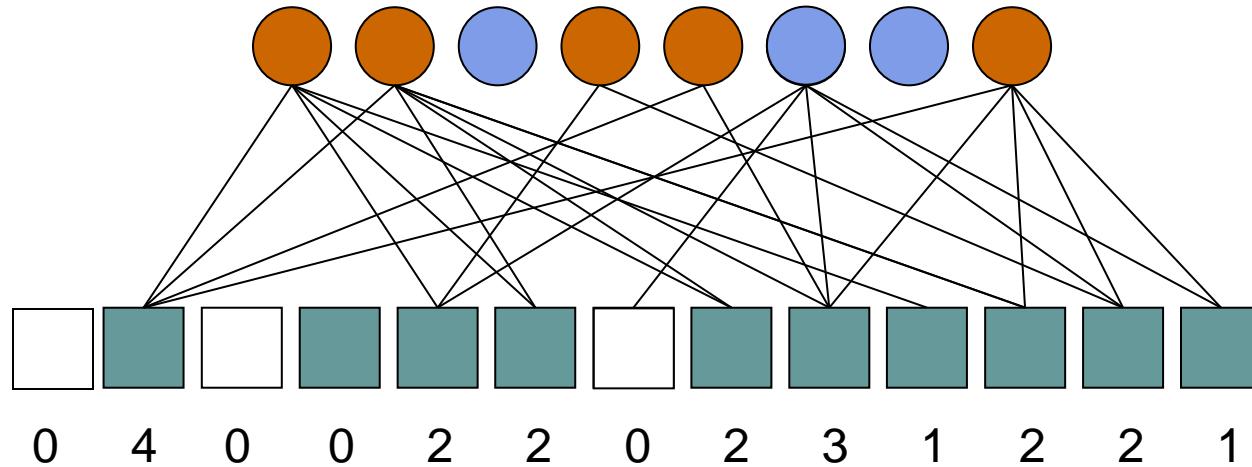


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
8 Martie 2024
TACCFDRT - Curs 2

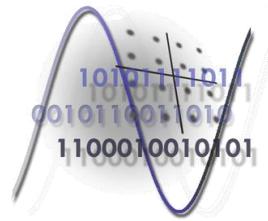


Decodare LT

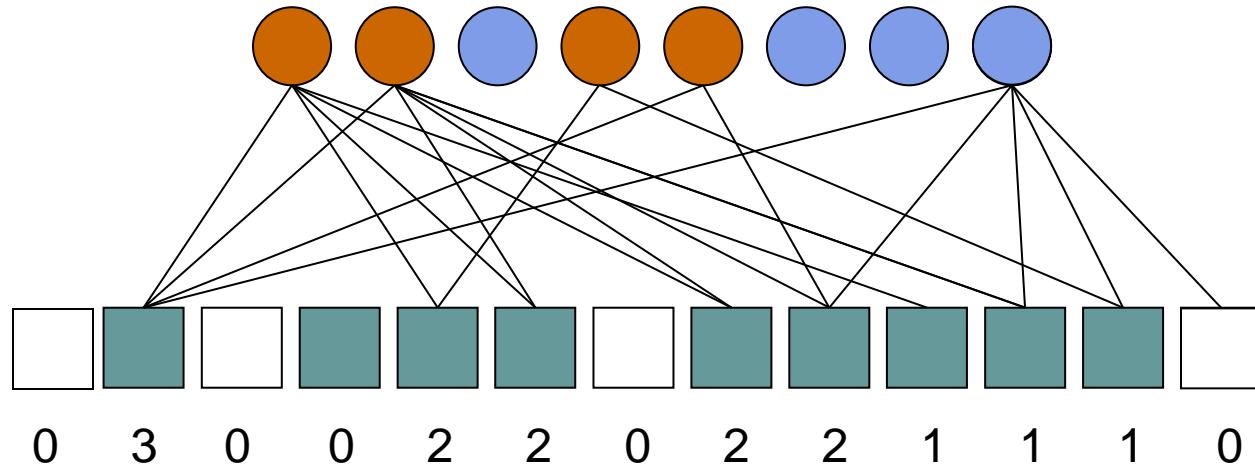


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
8 Marie 2024
TACCFDRT - Curs 2

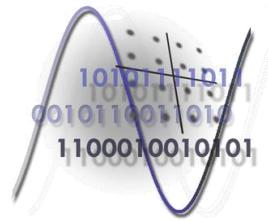


Decodare LT

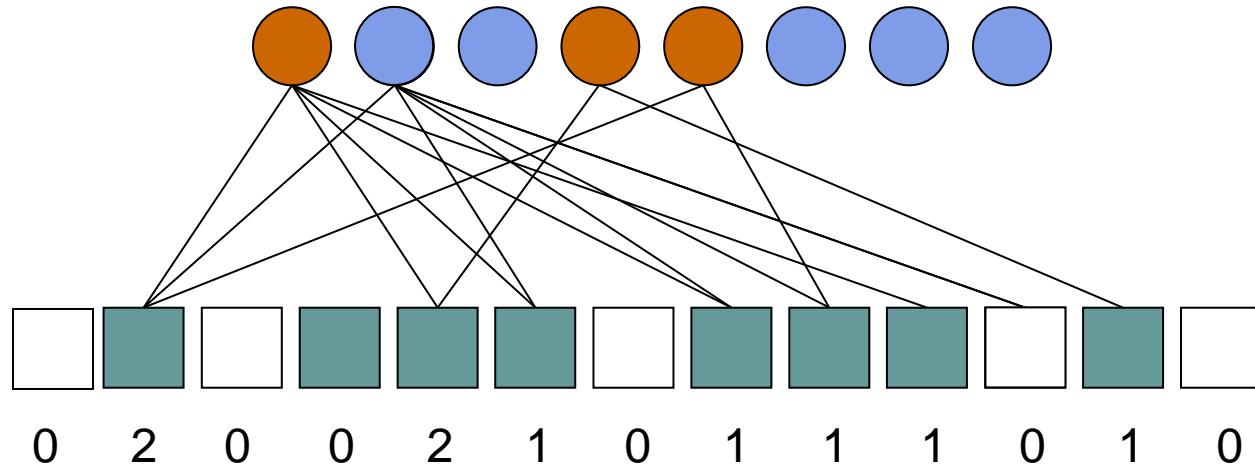


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
8 Martie 2024
TACCFDRT - Curs 2

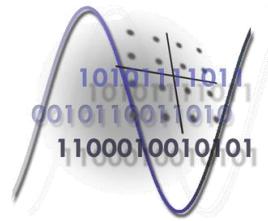


Decodare LT

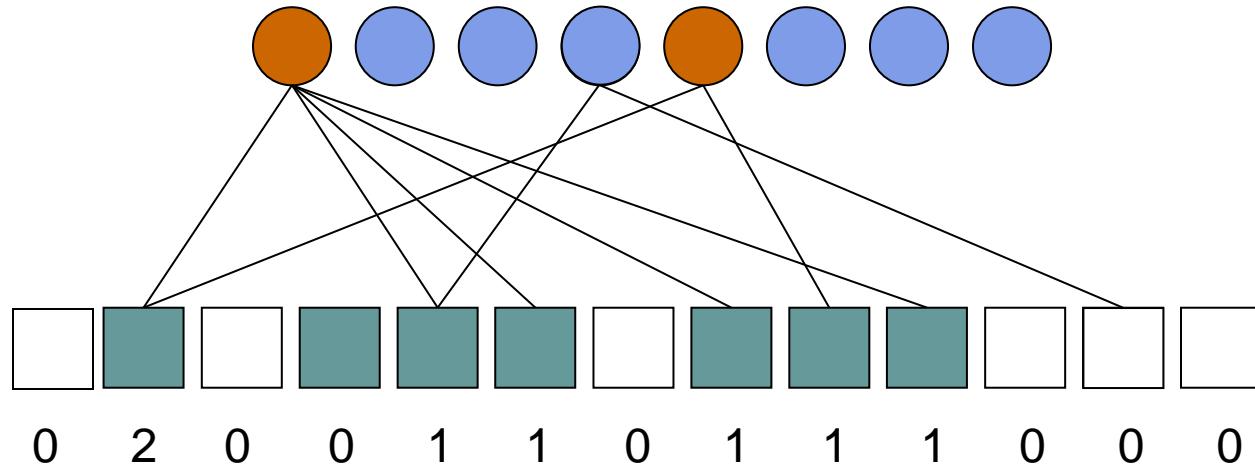


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
8 Martie 2024
TACCFDRT - Curs 2

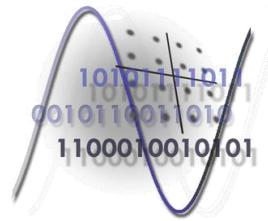


Decodare LT

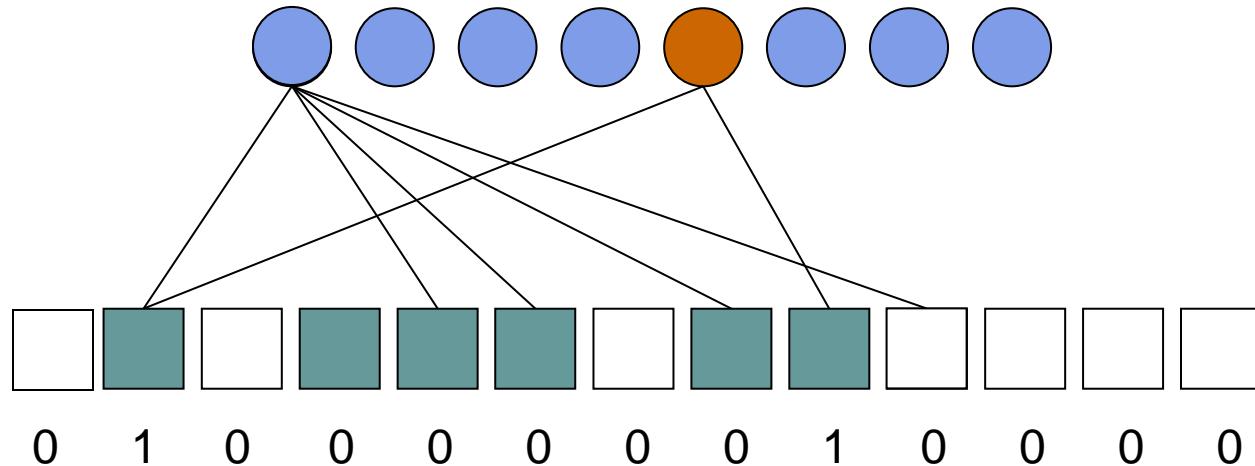


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
8 Martie 2024
TACCFDRT - Curs 2

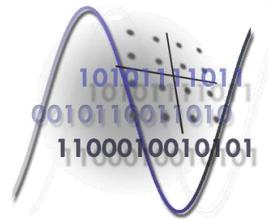


Decodare LT

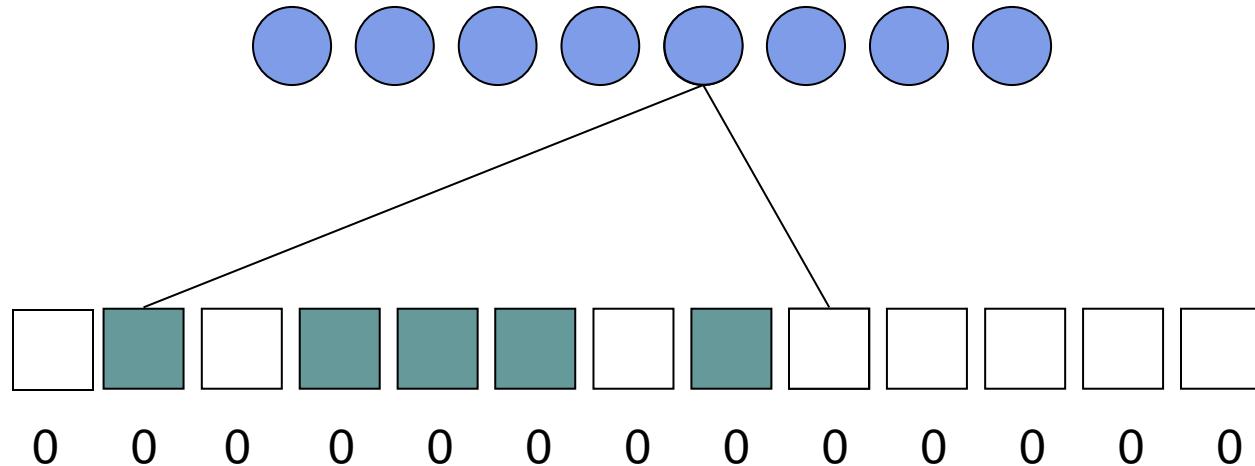


[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
8 Martie 2024
TACCFDRT - Curs 2



Decodare LT

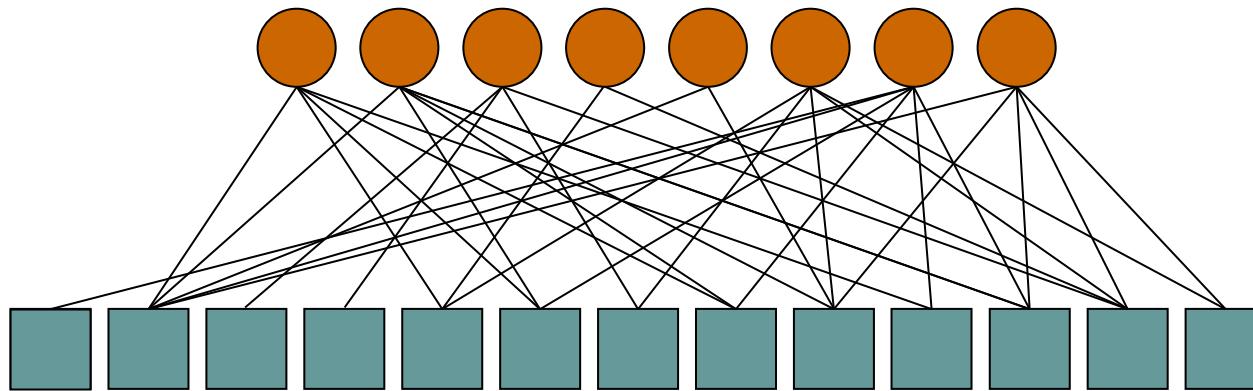
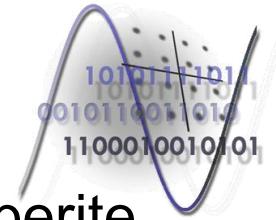


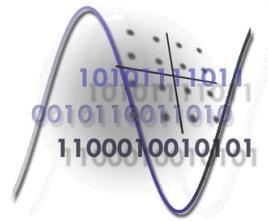
[3] Michael Luby; “LT Codes” -*Digital Fountain, Inc.*, Fremont, 2002

[4] Amin Shokrollahi; “Fountain Codes” EPFL and Digital Fountain, Inc.
8 Martie 2024
TACCFDRT - Curs 2

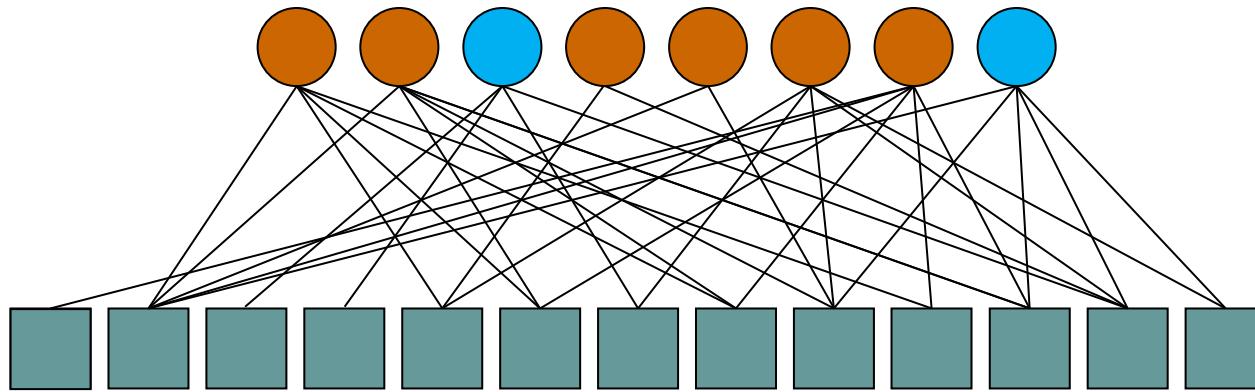
- Pt studierea distribuțiilor se reformulează procesul de decodare:

- La început toate simbolurile informaționale sunt descoperite

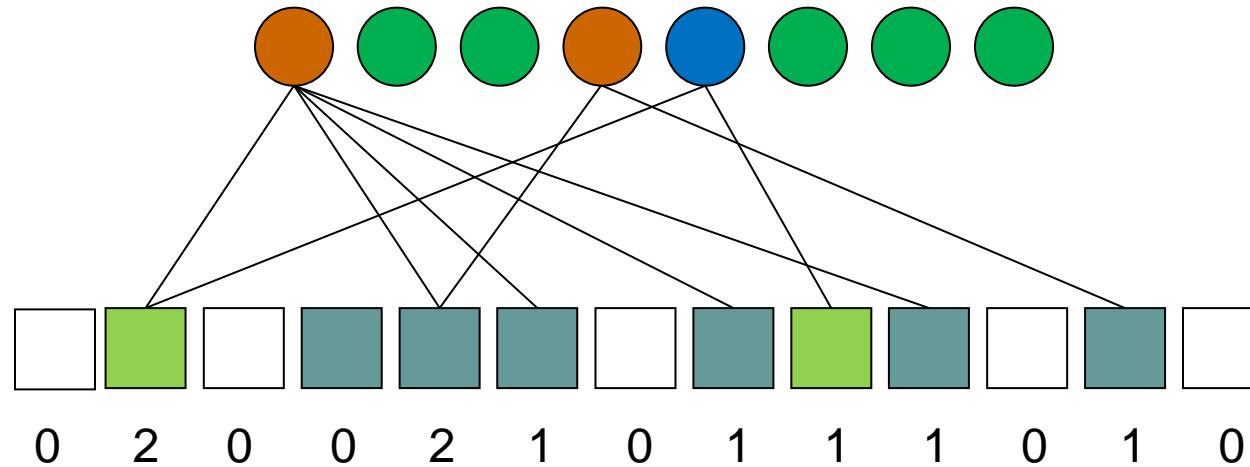




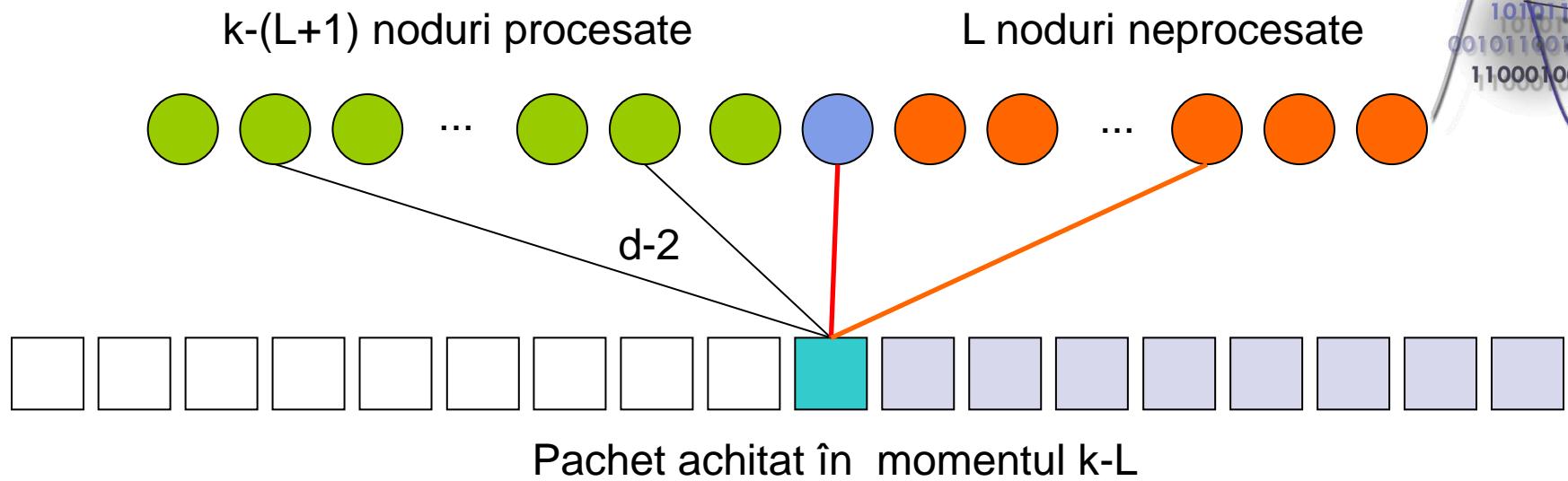
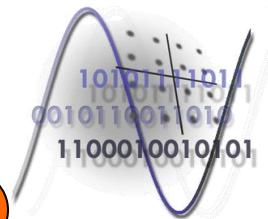
- În prima etapă toate simbolurile codate care sunt formate dintr-un singur simbol informațional, “acoperă” singurul lor vecin. Multimea formată din simbolurile acoperite, care încă nu au fost procesate, se numește “riplu”



- În pașii următor este luat câte un simbol informațional din riplu, este adunat la simbolurile codate la care este vecin și se reduce gradul acestor simboluri.
- Dacă un simbol codat astfel va avea grad 1, acest simbol va acoperi vecinul său, iar acest simbol codat astfel va fi **“achitat”**. Dacă acest vecin acoperit nu a fost acoperit mai devreme, de un alt simbol codat, atunci dimensiunea riplului crește.



- Procesul se termină când riplul se golește
 - Procesul de decodare este cu succes dacă la golirea riplului nu mai sunt simboluri de intrare neacoperite.



- Probabilitatea $q(d,L)$ ca un pachet cu gradul inițial d să fie achitat când mai sunt L pachete informaționale neprocesate este:

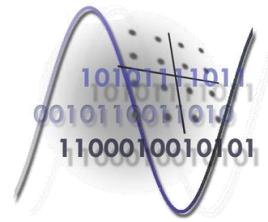
$$q(1,k)=1$$

pentru $d = 2, \dots, k$ și $L = k-d+1, \dots, 1$

$$q(d,L) = \frac{\binom{k-(L+1)}{d-2} \cdot \binom{L}{1}}{\binom{k-1}{d}} = \frac{d(d-1) \cdot L \prod_{j=0}^{d-3} k - (L+1) - j}{\prod_{j=0}^{d-1} k - j}$$

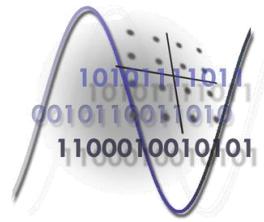
pentru celelalte valori d și L

$$q(d,L) = 0;$$



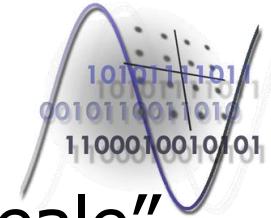
- Probabilitatea $r(d,L)$ este probabilitatea ca un simbol codat să aibă gradul d , și să fie achitat când mai sunt L simboluri de intrare neprocesate
 - $r(d,L) = p(d)q(d,L)$
- Probabilitatea $r(L)$ ca un pachet să fie achitat când mai sunt L pachete informative neprocesate este:

$$r(L) = \sum_d r(d,L) = \sum_{d=2}^k p(d) \frac{d(d-1) \cdot L \prod_{j=0}^{d-3} k - (L+1) - j}{\prod_{j=0}^{d-1} k - j}$$



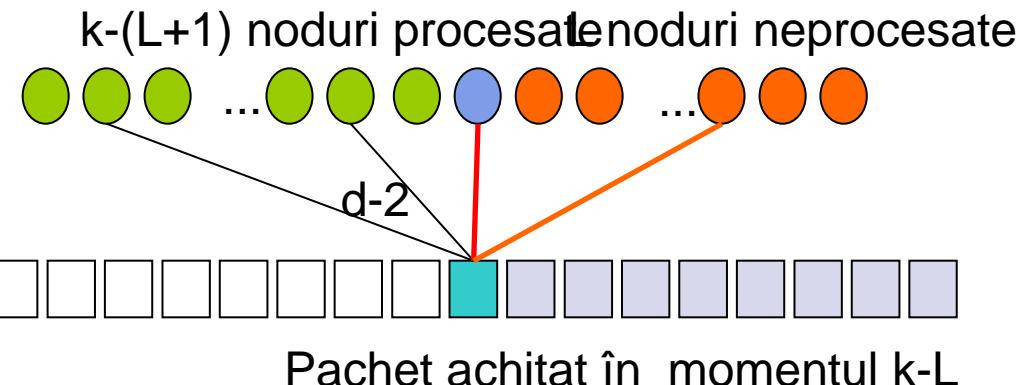
Distribuția *Soliton* ideală

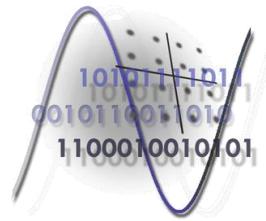
- Cerințele impuse unei distribuții de graduri sunt:
 - Un număr mediu de simboluri codate cât mai mic posibil pentru a asigura succesul procesului LT.
 - Gradul mediu al simbolurilor cât mai mic posibil. Gradul mediu definește numărul operațiilor de simbol necesare pentru generarea unui simbol codat, iar $k^*(\text{grad mediu})$ este numărul operațiilor necesare pentru recuperarea completă a datelor.



Distribuția *Soliton* ideală

- O proprietate elementară a unei distribuții "ideale" este ca la procesul de decodare, la procesarea unui simbol informational la riplu să fie adăugat un simbol acoperit.
- Asta asigură că dimensiunea riplului să nu fie niciodată prea mică sau prea mare.
- $r(L)$ –este probabilitatea ca la riplu să fie adunat un singur simbol la procesarea simbolului $k-(L+1)$





Distribuția *Soliton* ideală

- Tânărând cont că:

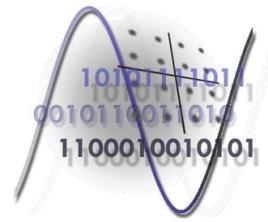
$$r(L) = \sum_d r(d, L) = \sum_{d=2}^k p(d) \frac{d(d-1) \cdot L \prod_{j=0}^{d-3} k - (L+1) - j}{\prod_{j=0}^{d-1} k - j}$$

și

$$\sum_{d=2}^k \frac{L \prod_{j=0}^{d-3} k - (L+1) - j}{\prod_{j=0}^{d-1} k - j} = 1 \quad pt\ orice\ L > 1$$

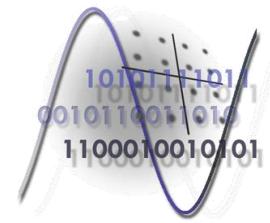
Rezultă distribuția Soliton ideală:

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



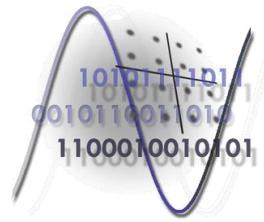
Distribuția *Soliton* ideală

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



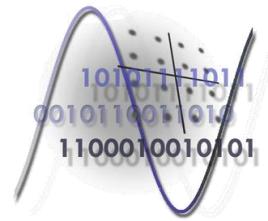
Distribuția Soliton Robust

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



Distribuția Soliton Robust

- Cât trebuie să fie suma gradurilor pachetelor receptionate ca fiecare pachet informațional să fie acoperit cu probabilitate $1-\delta$?
 - Problema este echivalentă cu problema “clasică” coșuri și mingi:
 - avem k coșuri
 - se aruncă K mingi, fiecare minge intră într-unul dintre coșuri
 - Câte mingi trebuie aruncate ca probabilitatea ca în fiecare coș să intre cel puțin o minge să fie $1-\delta$



Distribuția Soliton Robust

- Câte mingi trebuie aruncate ca probabilitatea ca în fiecare coș să intre cel puțin o minge să fie $1-\delta$
- Probabilitatea ca o minge să intre în coșul i este $1/k$
- Probabilitatea ca după N încercări să existe un coș gol este:

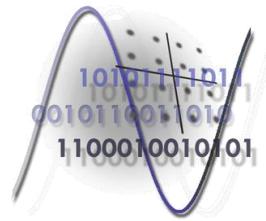
$$k \left(1 - \frac{1}{k}\right)^N$$

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

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

- Logaritmând ecuația obținem:

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



Distribuția Soliton Robust

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

- Dar....

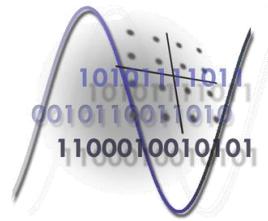
$$N \ln\left(1 - \frac{1}{k}\right)^{\frac{k}{k}} = \ln\left(\frac{\delta}{k}\right)$$

- 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:

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



Distribuția Soliton Robust

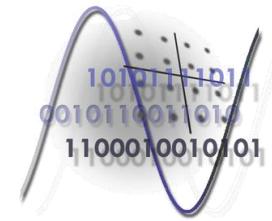
$$N \frac{1}{k} \ln\left(\frac{1}{e}\right) = \ln\left(\frac{\delta}{k}\right)$$

- Dar....

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

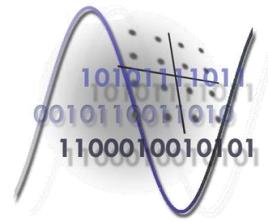
- Adică

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



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 deviație față de valoarea medie mai mare ca $\ln\left(\frac{k}{\delta}\right)\sqrt{k}$ apare cu probabilitatea δ



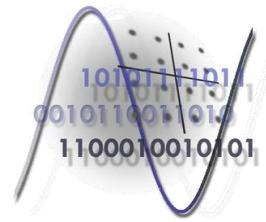
Distribuția Soliton Robust

- Pentru un compromis se acceptă că din K pachete recepționate, decodorul LT nu reușește să determine informațiile 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:

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



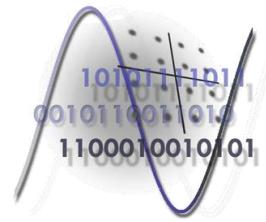
Distribuția Soliton Robust

- Se alege lungimea riplului dorit ca fiind:

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

- Se definește $\tau(d)$ ca fiind

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



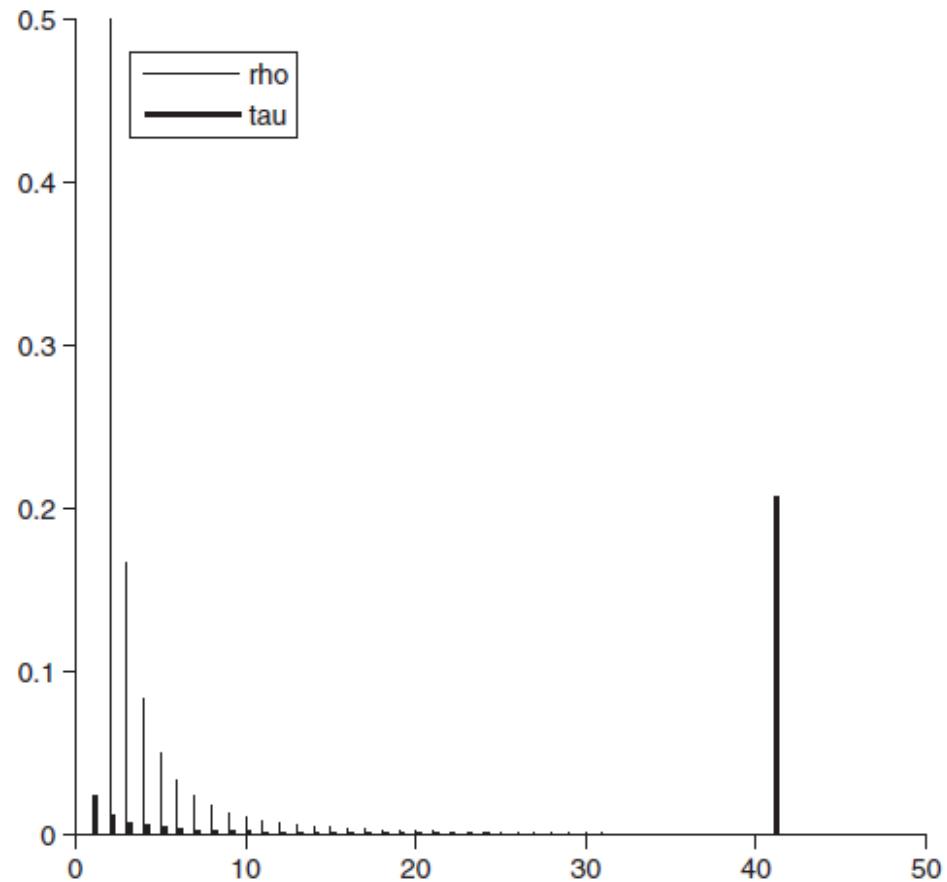
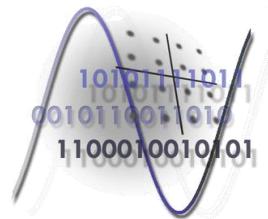
Distribuția Soliton Robust

- Se adună distribuția ideală $p(d)$ la $\tau(d)$, normalizând această sumă cu β se obține distribuția robustă $\mu(d)$

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

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

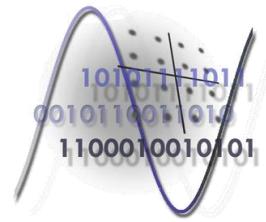
- Valoarea medie a gradurilor este $c/n(k)$
- Overageheadul necesar este proporțional cu K



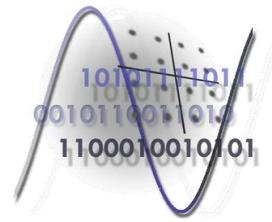
$$p(d) = \begin{cases} \frac{1}{k}; & \text{pt } d = 1 \\ \frac{1}{d(d-1)}; & \text{pt } d = 2, \dots, k \end{cases}$$

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

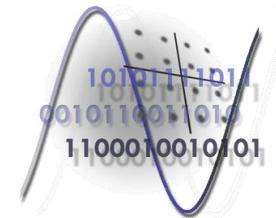
- Distribuția soliton ideală și robustă pentru $k=10000$, $c=0.2$, $\delta=0.05$, $R=244$, $k/R=41$, iar $\beta \approx 1.3$ [1]



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

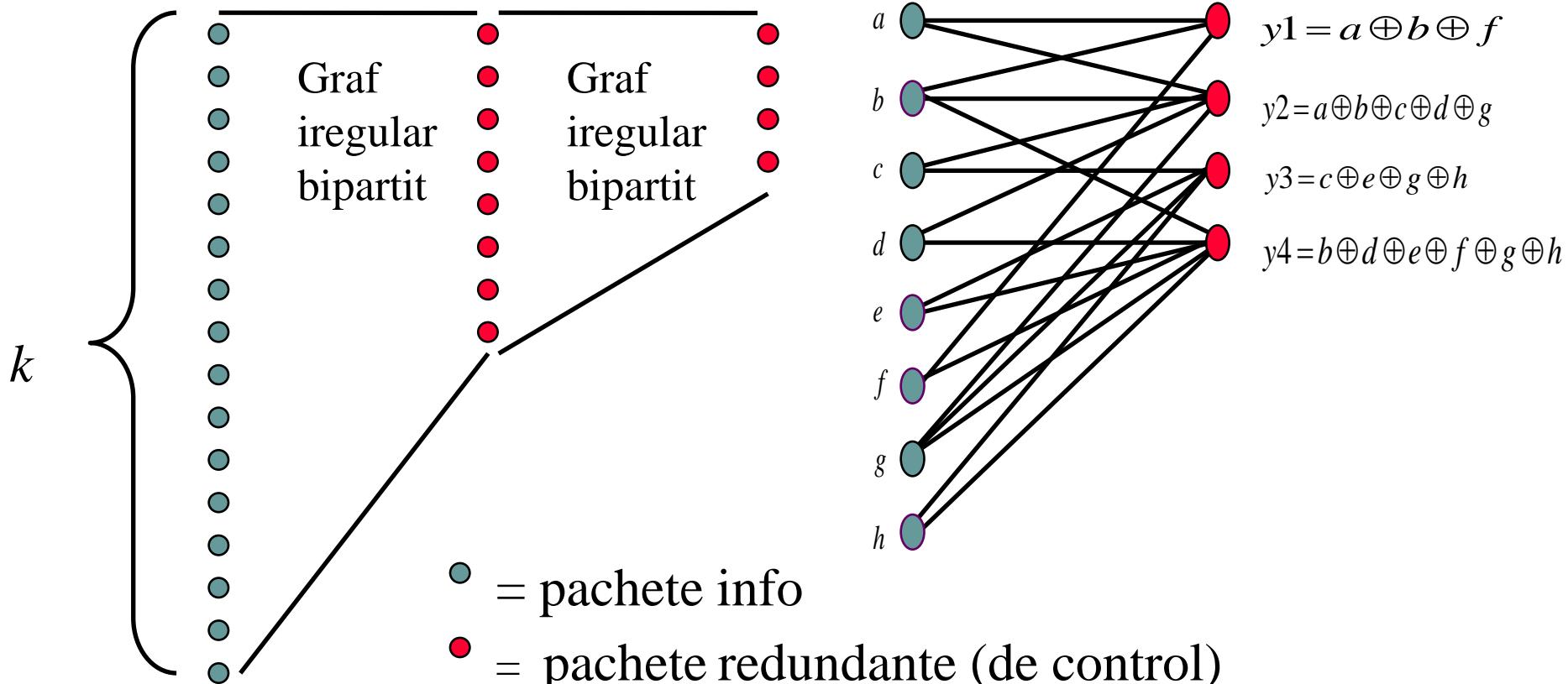


- Deoarece gradul pachetelor nu este constant
 - Timpul de codare/decodare nu este liniară
 - Probabilitatea de pierdere a pachetelor nu este uniformă
- Problema este rezolvată de codurile Raptor introduse de Amin Shokrollahi

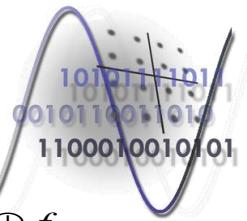


Coduri Tornado

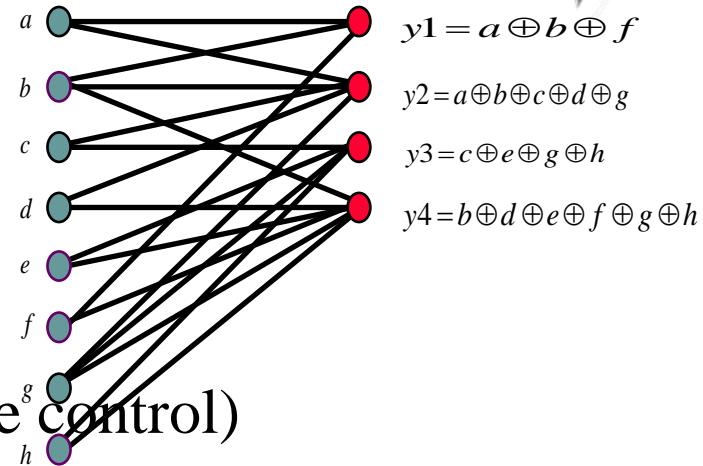
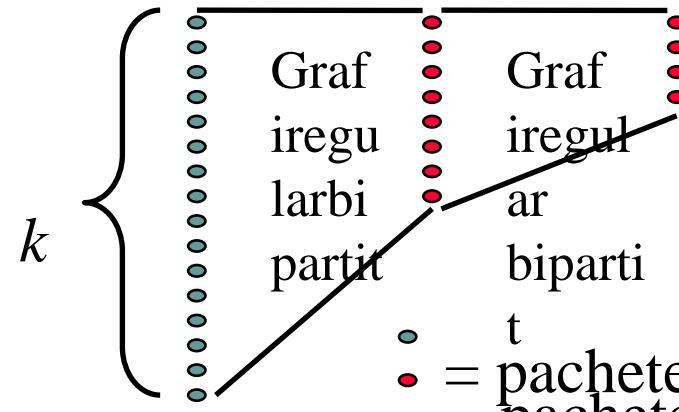
- Codurile Tornado sunt construite pe baza unor grafuri bipartite



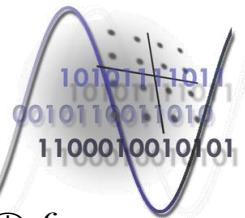
[2] John W.Byers, Michael Luby, Michael Mitzenmacher, Ashutosh Rege; "A Digital Fountain Approach to Reliable Distribution of Bulk Data" -Proceedings of the ACM SIGCOMM '98



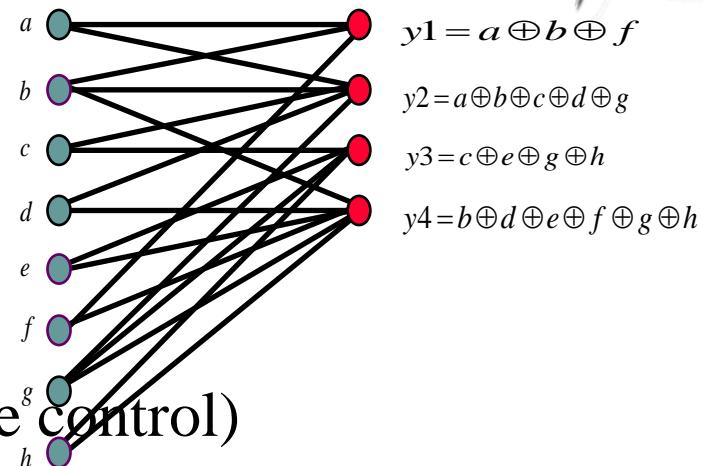
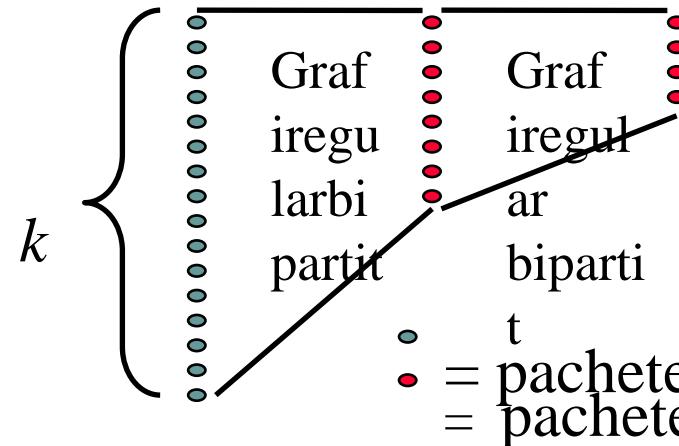
Coduri Tornado



- În prima etapă din cele k pachete informaționale se obțin βk pachete de control (β număr pozitiv subunitar) conform grafului B1
- În a doua etapă pornind de la cele βk pachete de control obținute în etapa anterioară se obțin alte $\beta\beta k$ pachete de control conform grafului B2

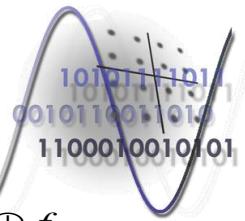


Coduri Tornado

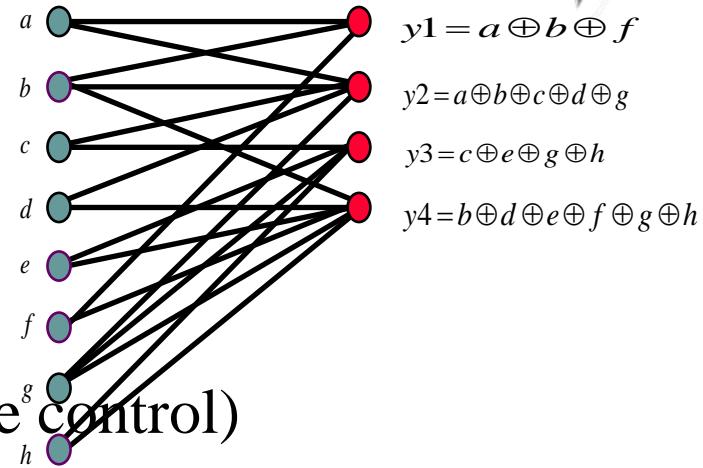
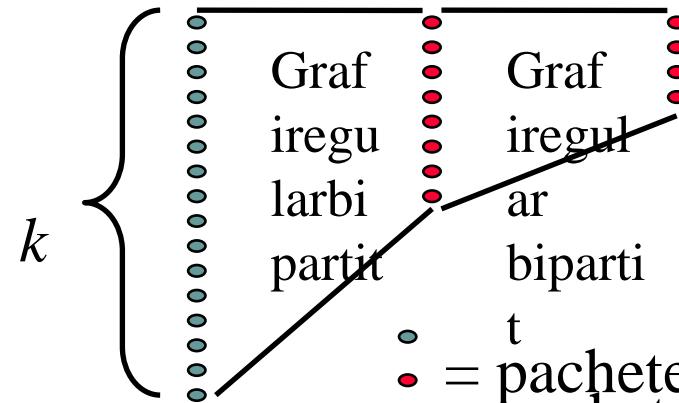


- În ultima (m -a) etapă se utilizează un cod corector de ștergeri "clasic", C, cu rata $1 - \beta$ care poate să corecteze β ștergeri
- Codul C are $\beta^{m-1}k$ simboluri (pachete) la intrare, și generează $\beta \beta^{m-1}k/(1 - \beta)$ simboluri de control adiționale
- Numărul simbolurilor de control generate în cele m etape este

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



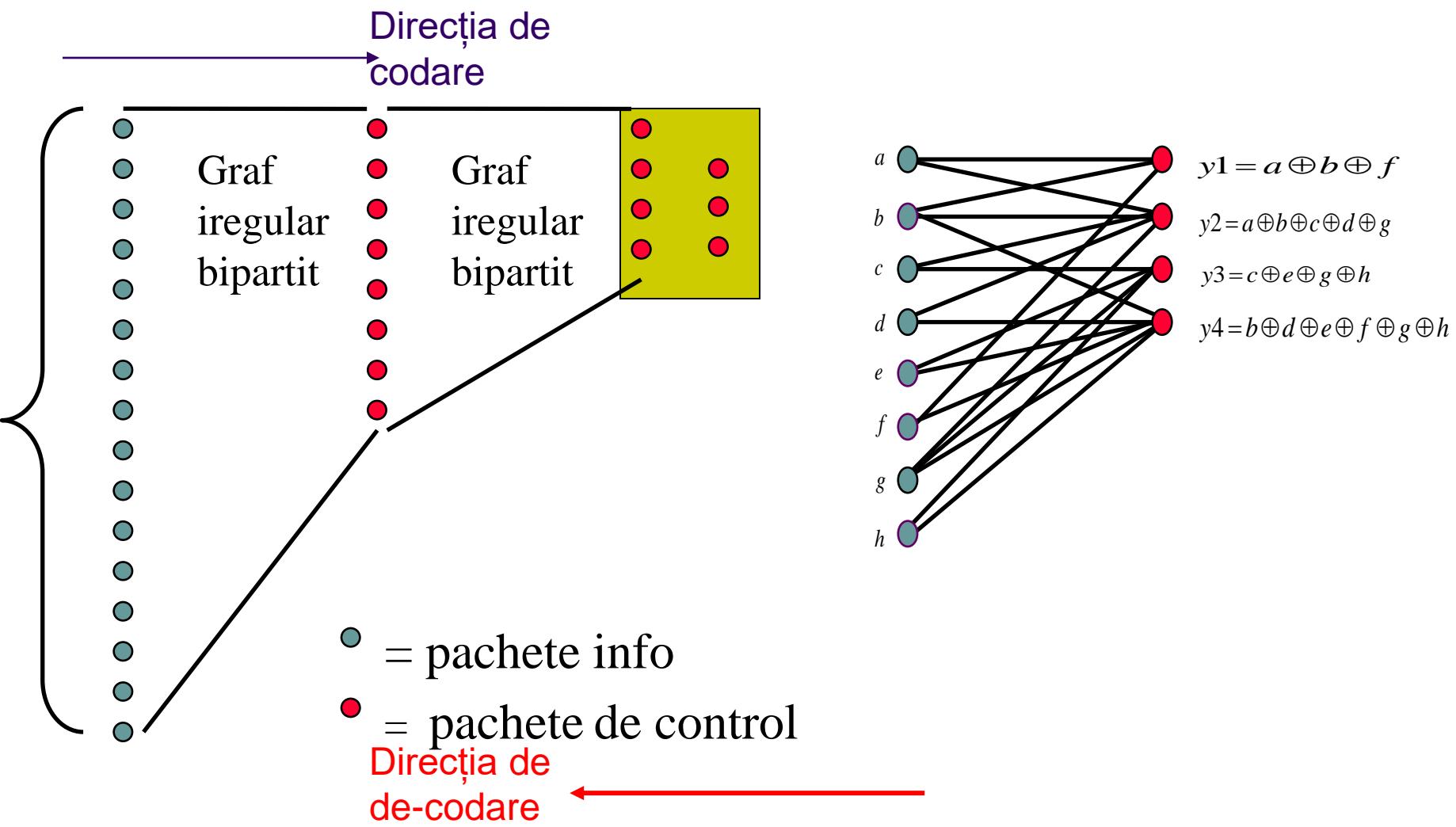
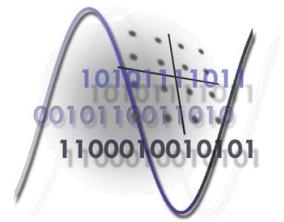
Coduri Tornado

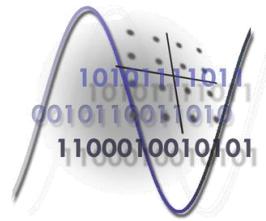


- rata globală va fi:

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

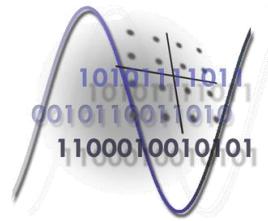
- Codul global $C(B_1, B_2, \dots, B_{m-1}, C)$ este un corector de ștergeri cu rata $1 - \beta$, care poate să corecteze cu probabilitate mare orice ștergere până la lungime $\cdot \beta$



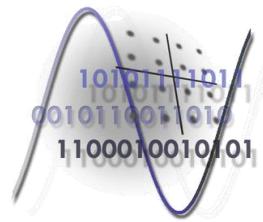


Coduri Tornado

- Procesul de codare constă în sumarea modulo 2 a pachetelor conform grafului “generator”
- Pachetele recepționate sunt “substituite” în ecuațiile de control
- Prin înlocuirea variabilelor în ecuații 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 recepționate este mai mare de k , un pachet recepționat generează o avalanșă de “rezolvări” permitând reconstrucția pachetelor informative



- Deoarece nodurile din graf au un număr redus de vecini (ecuații cu putini termeni) operațiunea de codare și decodare nu necesită multe operații
- Numărul de operații pentru obținerea pachetelor redundante depinde numai de gradul nodului respectiv
- Complexitatea decodării depinde tot de gradul nodurilor și de “poziția” pachetelor recepționate în graf
- Codurile Tornado sunt tot coduri cu lungime finită, deci pentru utilizarea lor ca DF pachetele codate se transmit întrețesute ciclic

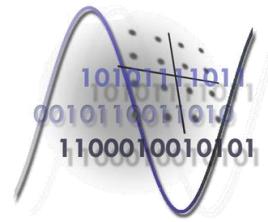


• Comparație între timpii de codare și decodare

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.

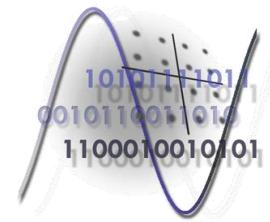
Timpul de decodare		
Size	Reed-Solomon	Tornado
250 K	2.06 sec.	0.18 sec.
500 K	8.4 sec.	0.24 sec.
1 MB	40.5 sec.	0.31 sec.
2 MB	199 sec.	0.44 sec.
4 MB	13 min.	0.74 sec.
8 MB	1 hr.	1.28 sec.
16 MB	4 hrs.	2.27 sec.

[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



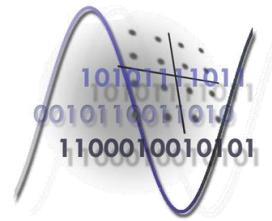
Tornado vs LT

- Atât codurile RS cât și codurile Tornado sunt coduri sistematice, în schimb codurile LT sunt nesistematice
- memoria necesară pentru codarea decodarea codurilor tornado este mult mai mare decât 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 grafuri cu densitate logaritmică



Coduri Raptor(RAPide TORnado)

- Primul cod DF cu timp de codare și decodare liniară
- au fost inventate în 2000/2001 publicate în 2004
- Se acceptă ca o fracțiune maximă 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 ștergeri clasic
 - Se cunoaște numărul maxim de ștergeri posibile
- Implică o precodare cu un cod corector de ștergeri clasic



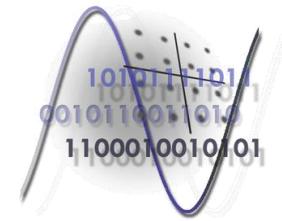
● ● ● ● ● ● ● ● ● Simboluri de intrare

Codare cu cod corector de ștergeri

● ● ● ● ● ● ● ● ● ● simboluri precodate

LT-”light”

■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ Simboluri codate



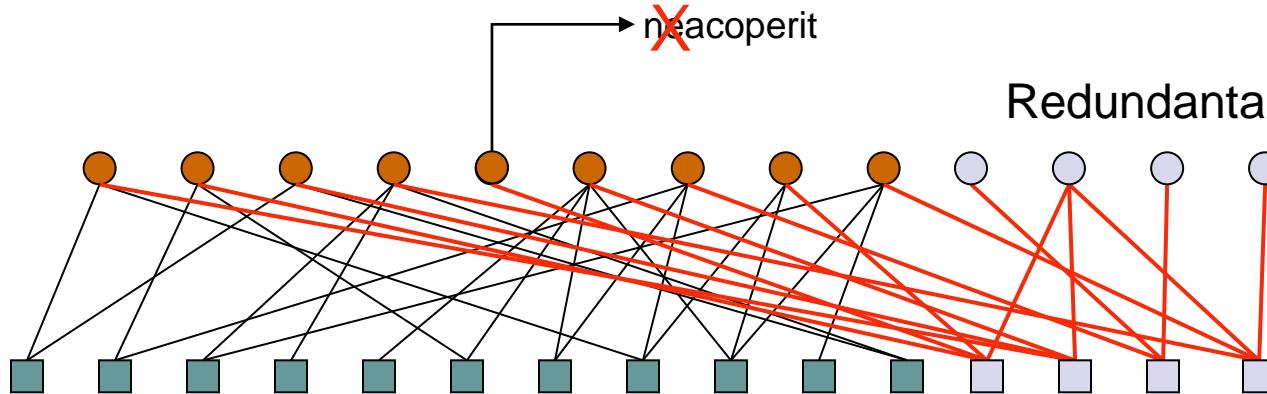
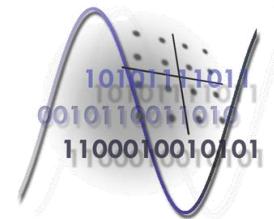
■ Simboluri recepționate

decoder LT

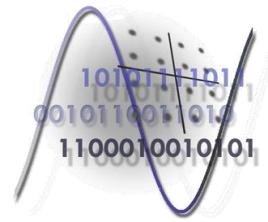
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● maxim δ simboluri neacoperite

Codare cu cod corector de ștergeri

● ● ● ● ● ● ● ● ● ● Simboluri decodeate

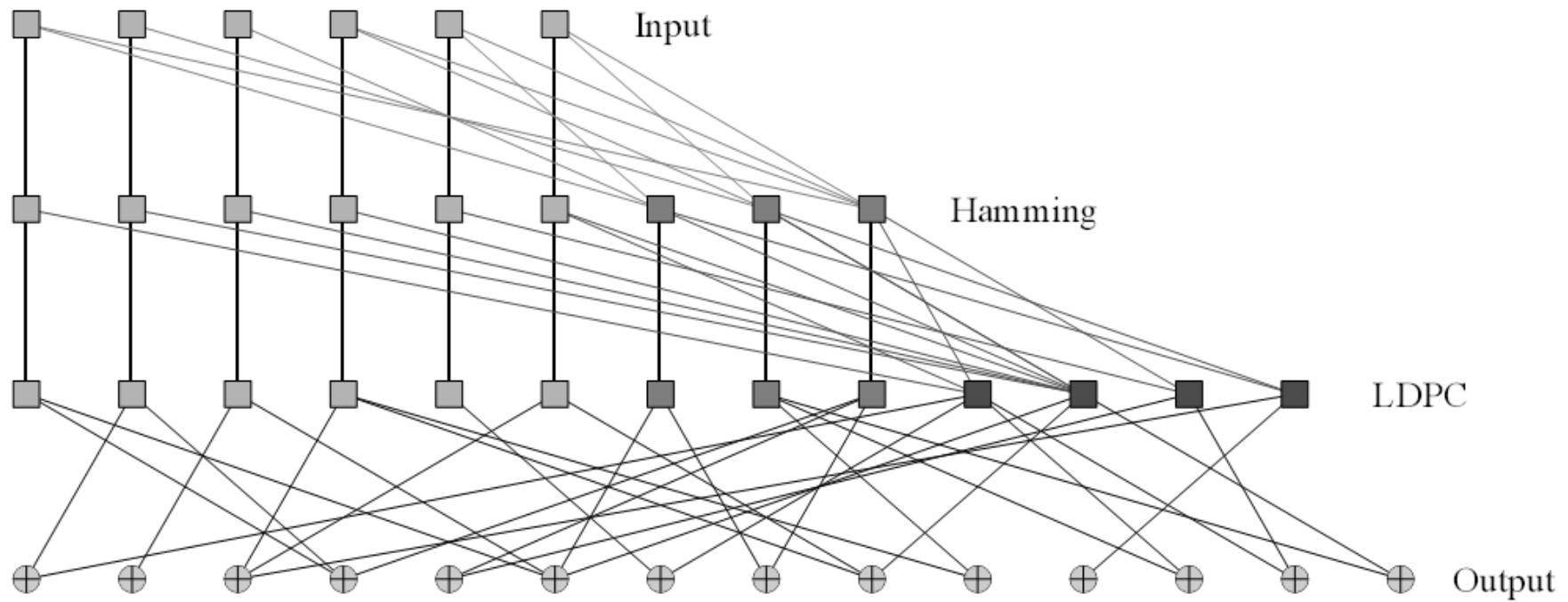
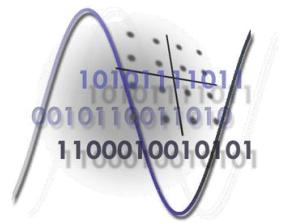


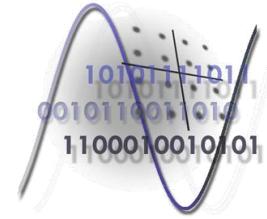
- Dacă precodorul este ales în mod corespunzător, atunci se poate utiliza un cod LT cu gradul mediu constant, care asigură timp de codare liniară
- Un cod Raptor $(k, C, \Omega(d))$ este definit de numărul de pachete informaționale k , codul corector de ștergeri C și distribuția gradurilor al codului LT $\Omega(d)$



Coduri Raptor

- Parametrii principali de performanță 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, și este definit ca numărul de simboluri de ieșire pe care trebuie să aibă decodorul pentru a recupera cu probabilitate mare simbolurile de intrare. Un overhead de $1+\varepsilon$ însemenă că trebuie recepționate $k(1+\varepsilon)$ simboluri de ieșire pentru a asigura o decodare cu succes cu o probabilitate mare.
 - Costul: Costul procesului de codare și de decodare.





Probleme legate de DF

- Implementarea oricărei aplicații, se poate face numai cu acordul DF inc.
- “...but networking people do not want to deal with developing codes.” [1]
- datorită drepturilor de autor aplicațiile cu DF sunt foarte puțin cunoscute

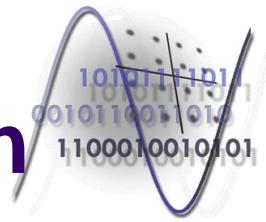
[1] Michael Mitzenmacher-Digital Fountains: Applications and Related Issues

Implementarea concepului de DF cu ajutorul codurilor cu rată finită.

Utilizarea codurilor LDPC
pentru implementarea
conceptului DF

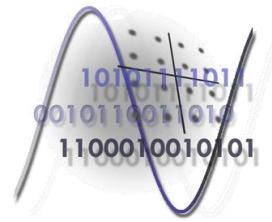
TACCFDRT Curs 3



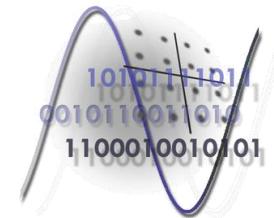


Tehnici de codare de tip Digital Fountain

- Coduri Tornado
- Coduri Raptor
 - Coduri LDPC
- Tipuri de coduri Raptor
- Utilizarea codurilor DF

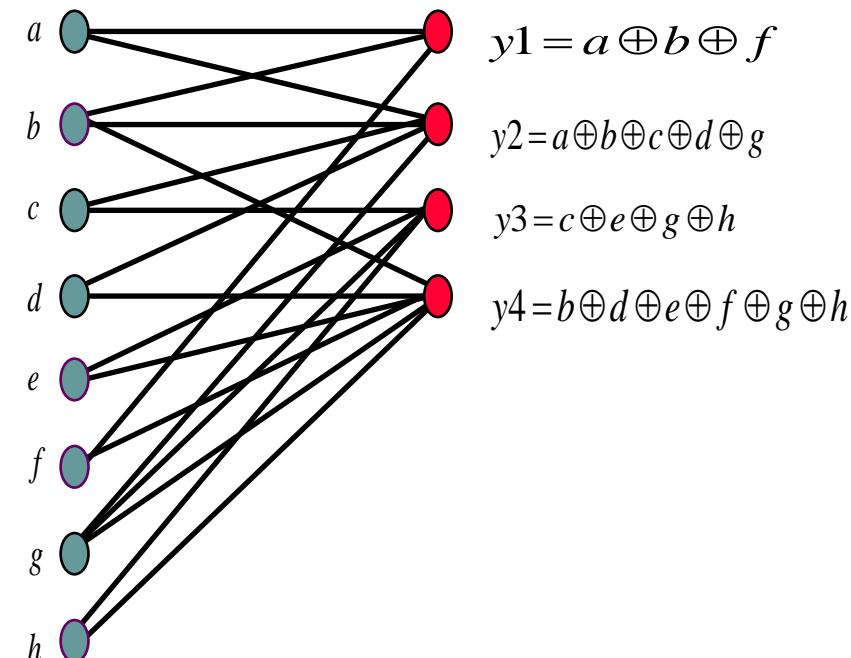
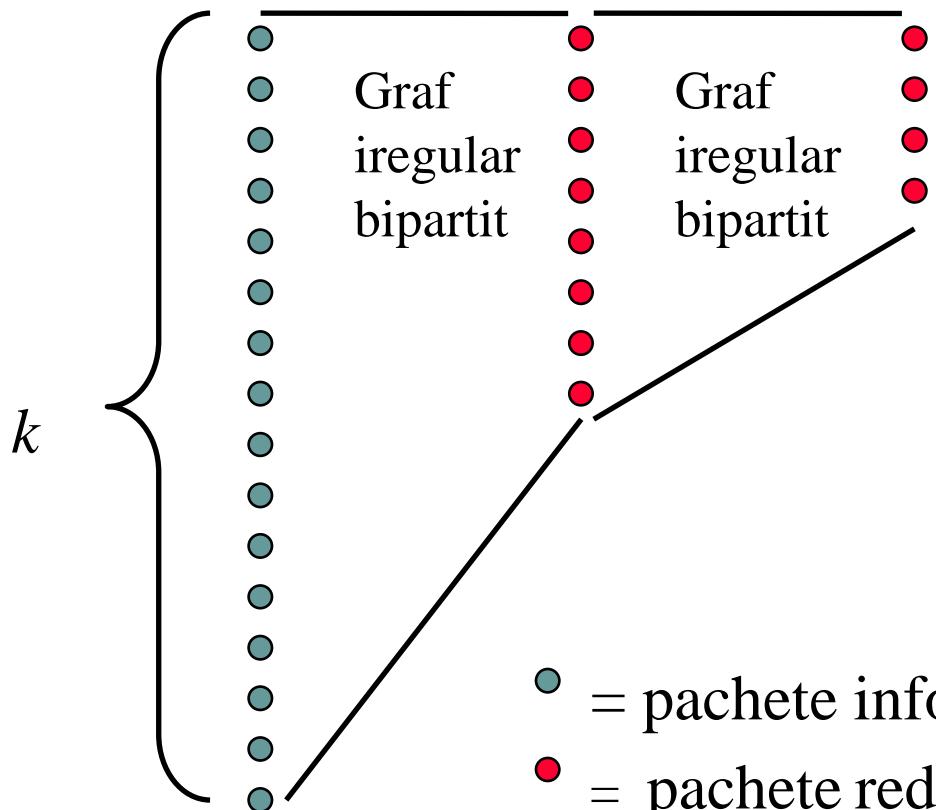


- Deoarece gradul pachetelor nu este constant
 - Timpul de codare/decodare nu este liniară
 - Probabilitatea de pierdere a pachetelor nu este uniformă
- Problema este rezolvată de codurile Raptor introduse de Amin Shokrollahi

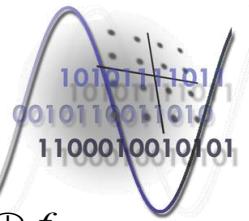


Coduri Tornado

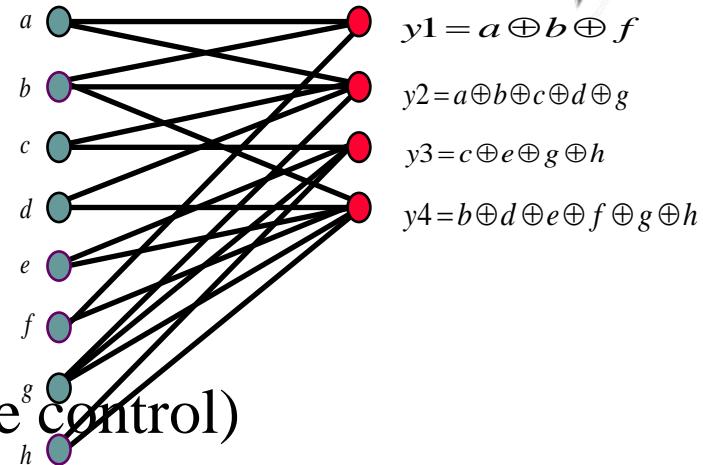
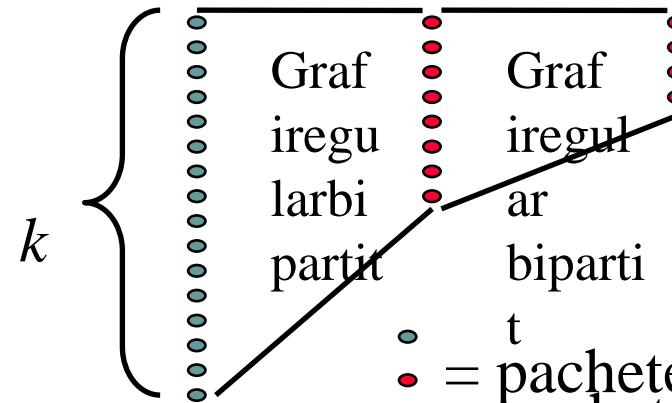
- Codurile Tornado sunt construite pe baza unor grafuri bipartite



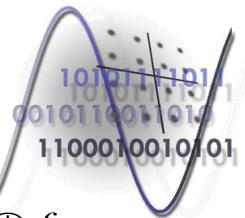
[2] John W.Byers, Michael Luby, Michael Mitzenmacher, Ashutosh Rege; "A Digital Fountain Approach to Reliable Distribution of Bulk Data" -Proceedings of the ACM SIGCOMM '98



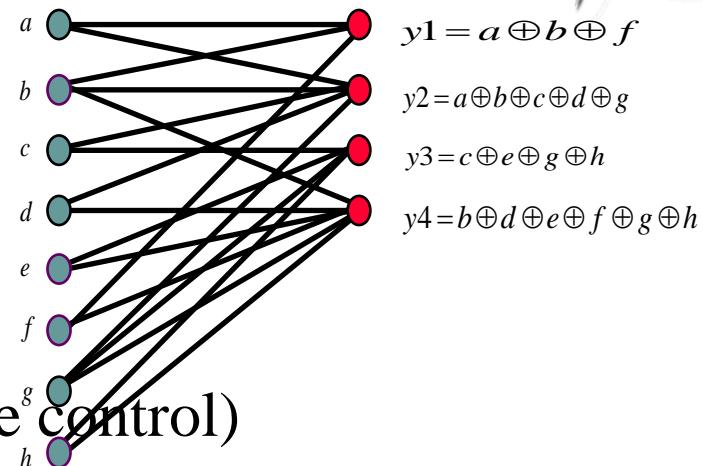
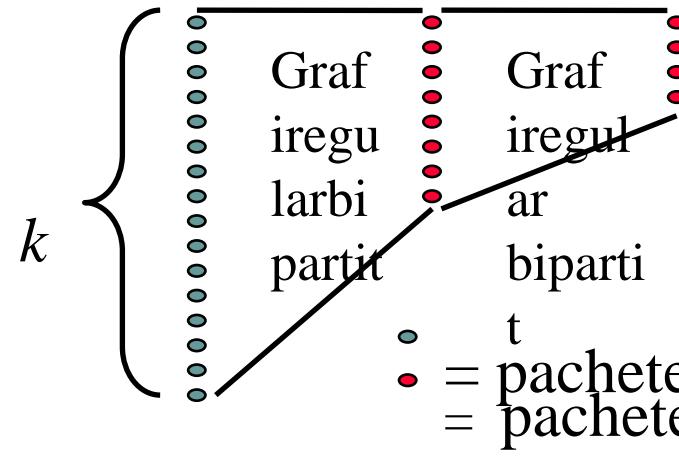
Coduri Tornado



- În prima etapă din cele k pachete informaționale se obțin βk pachete de control (β număr pozitiv subunitar) conform grafului B1
- În a doua etapă pornind de la cele βk pachete de control obținute în etapa anterioară se obțin alte $\beta\beta k$ pachete de control conform grafului B2

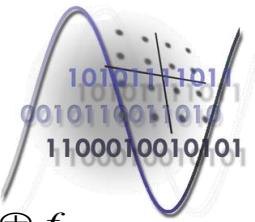


Coduri Tornado

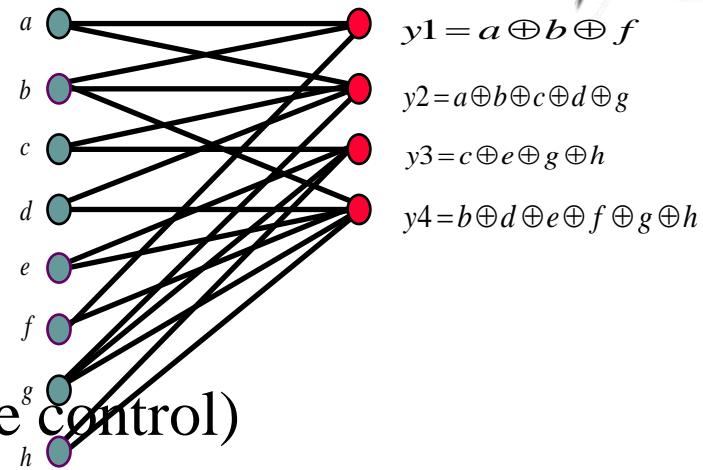
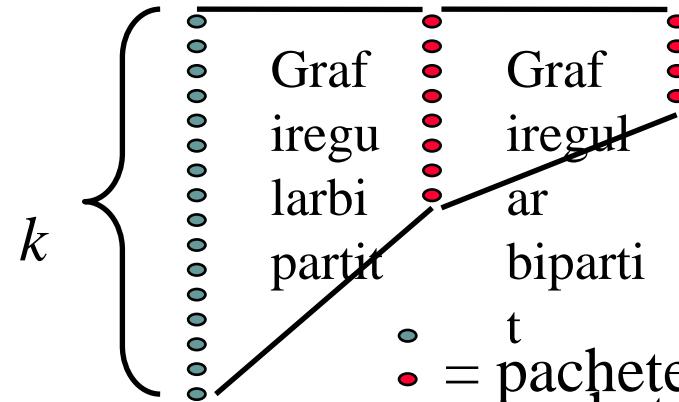


- În ultima (m -a) etapă se utilizează un cod corector de ștergeri "clasic", C, cu rata $1 - \beta$ care poate să corecteze β ștergeri
- Codul C are $\beta^{m-1}k$ simboluri (pachete) la intrare, și generează $\beta \beta^{m-1}k/(1 - \beta)$ simboluri de control adiționale
- Numărul simbolurilor de control generate în cele m etape este

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



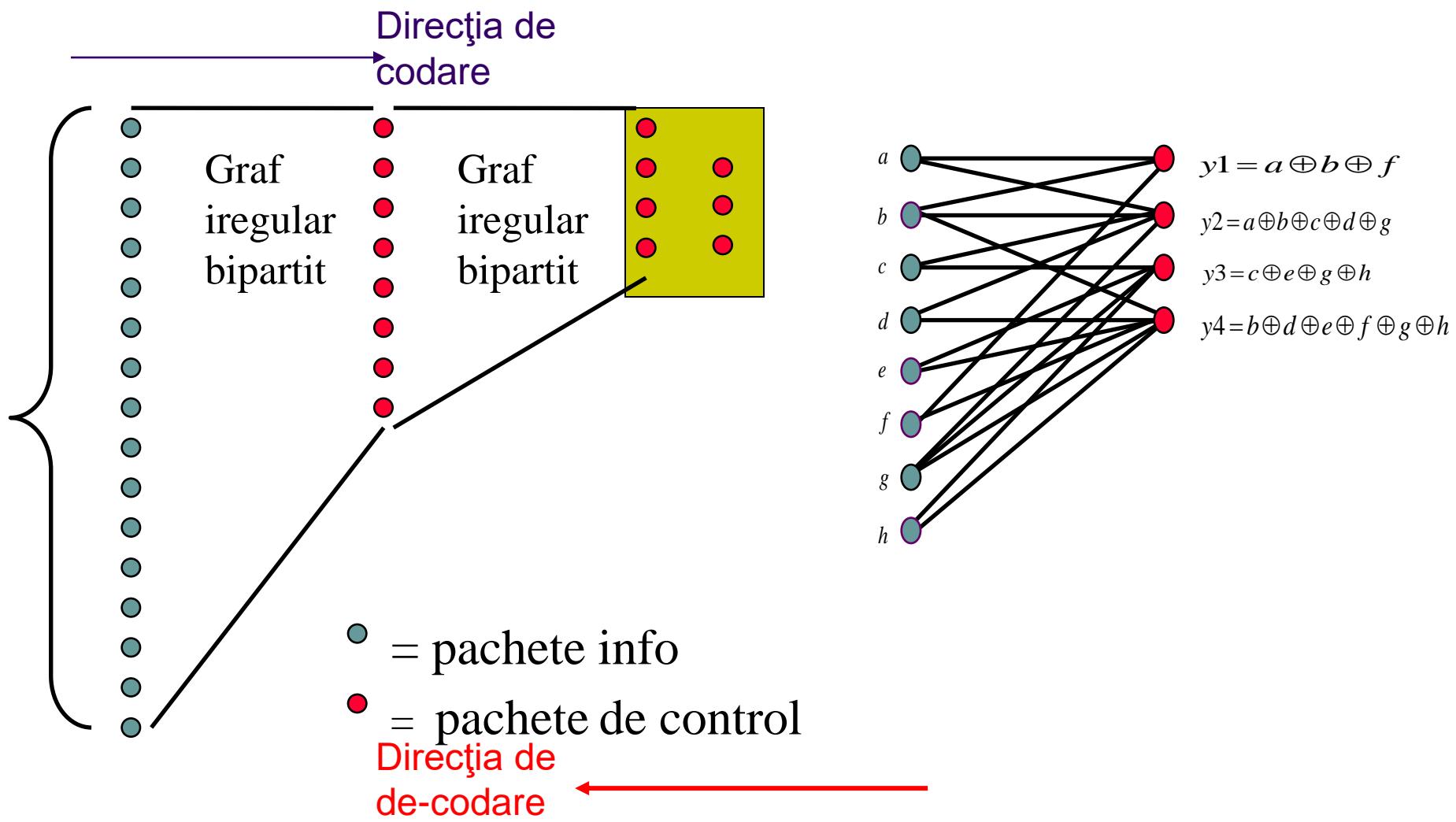
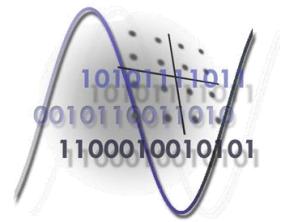
Coduri Tornado

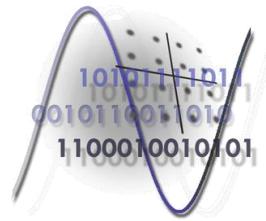


- rata globală va fi:

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

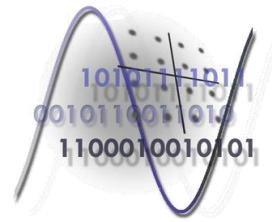
- Codul global $C(B_1, B_2, \dots, B_{m-1}, C)$ este un corector de ștergeri cu rata $1 - \beta$, care poate să corecteze cu probabilitate mare orice ștergere până la lungime $\cdot \beta$



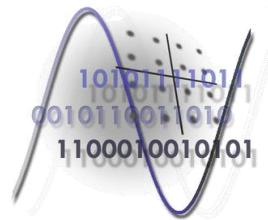


Coduri Tornado

- Procesul de codare constă în sumarea modulo 2 a pachetelor conform grafului “generator”
- Pachetele recepționate sunt “substituite” în ecuațiile de control
- Prin înlocuirea variabilelor în ecuații 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 receptionat generează o avalanșă de “rezolvări” permitând reconstrucția pachetelor informative



- Deoarece nodurile din graf au un număr redus de vecini (ecuații cu puțini termeni) operațiunea de codare și decodare nu necesită multe operații
- Numărul de operații pentru obținerea pachetelor redundante depinde numai de gradul nodului respectiv
- Complexitatea decodării depinde tot de gradul nodurilor și de “poziția” pachetelor recepționate în graf
- Codurile Tornado sunt tot coduri cu lungime finită, deci pentru utilizarea lor ca DF pachetele codate se transmit întrețesute ciclic

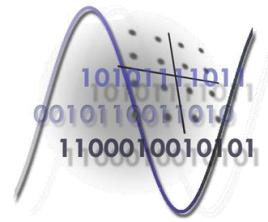


• Comparație între timpii de codare și decodare

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.

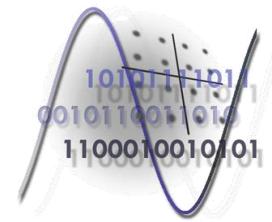
Timpul de decodare		
Size	Reed-Solomon	Tornado
250 K	2.06 sec.	0.18 sec.
500 K	8.4 sec.	0.24 sec.
1 MB	40.5 sec.	0.31 sec.
2 MB	199 sec.	0.44 sec.
4 MB	13 min.	0.74 sec.
8 MB	1 hr.	1.28 sec.
16 MB	4 hrs.	2.27 sec.

[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



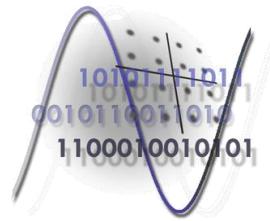
Tornado vs LT

- Atât codurile RS cât și codurile Tornado sunt coduri sistematice, în schimb codurile LT sunt nesistematice
- memoria necesară pentru codarea decodarea codurilor tornado este mult mai mare decât 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 grafuri cu densitate logaritmică



Coduri Raptor(RAPide TORnado)

- Primul cod DF cu timp de codare și decodare liniară
- au fost inventate în 2000/2001 publicate în 2004
- Se acceptă ca o fracțiune maximă 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 ștergeri classic
 - Se cunoaște numărul maxim de ștergeri posibile
- Implică o precodare cu un cod corector de ștergeri classic



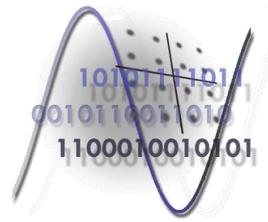
● ● ● ● ● ● ● ● ● Simoluri de intrare

Codare cu cod corector de ștergeri

● ● ● ● ● ● ● ● ● ● simboluri precodate

LT-”light”

■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ Simboluri codate



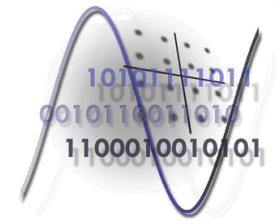
■ Simboluri receptionate

decodor LT

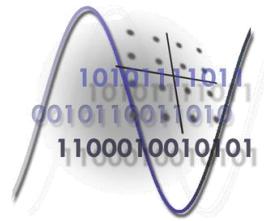
● ● ● ● ● ● ● ● ● ● ● ● ● maxim δ simboluri neacoperite

Codare cu cod corector de ștergeri

● ● ● ● ● ● ● ● ● ● Simboluri decodeate

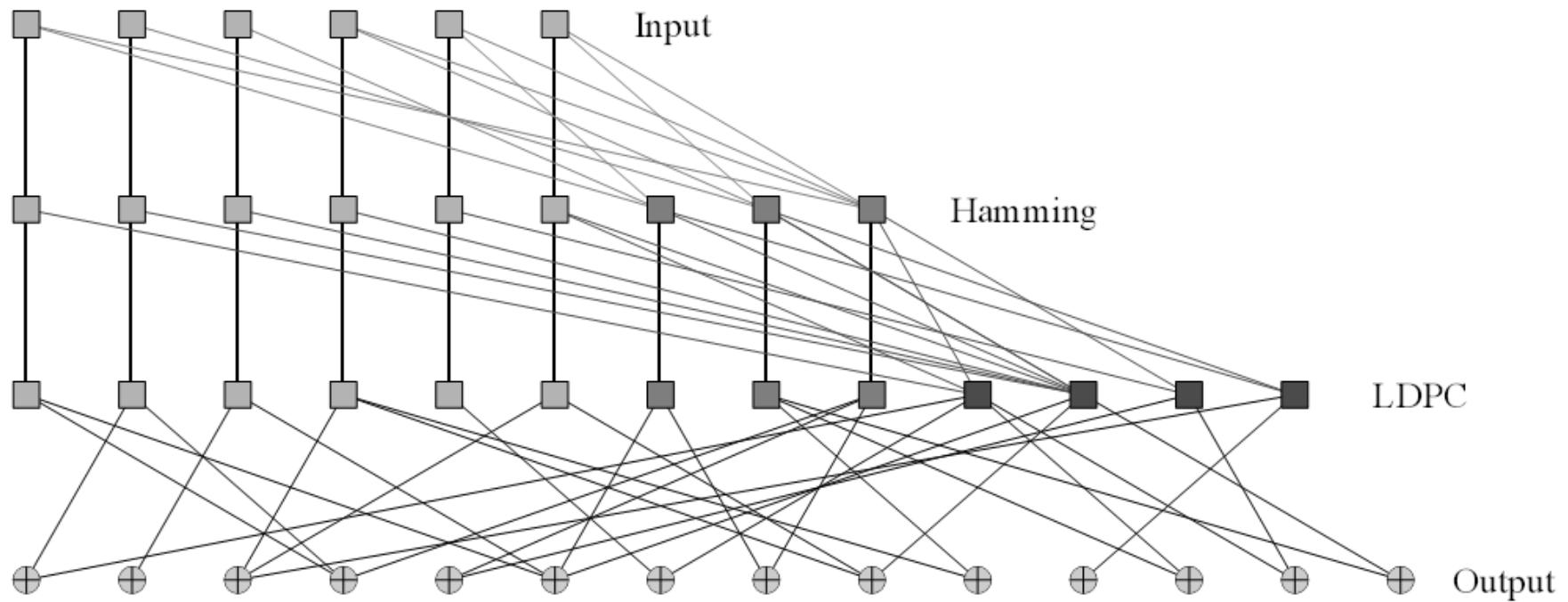
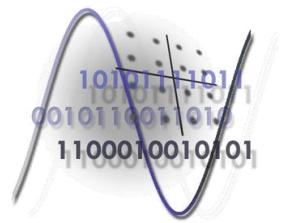


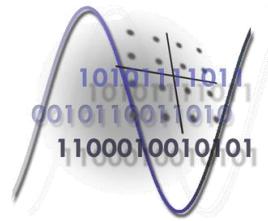
- Dacă precodorul este ales în mod corespunzător, atunci se poate utiliza un cod LT cu gradul mediu constant, care asigură timp de codare liniară
- Un cod Raptor ($k, C, \Omega(d)$) este definit de numărul de pachete informaționale k , codul corector de ștergeri C și distribuția gradurilor al codului LT $\Omega(d)$



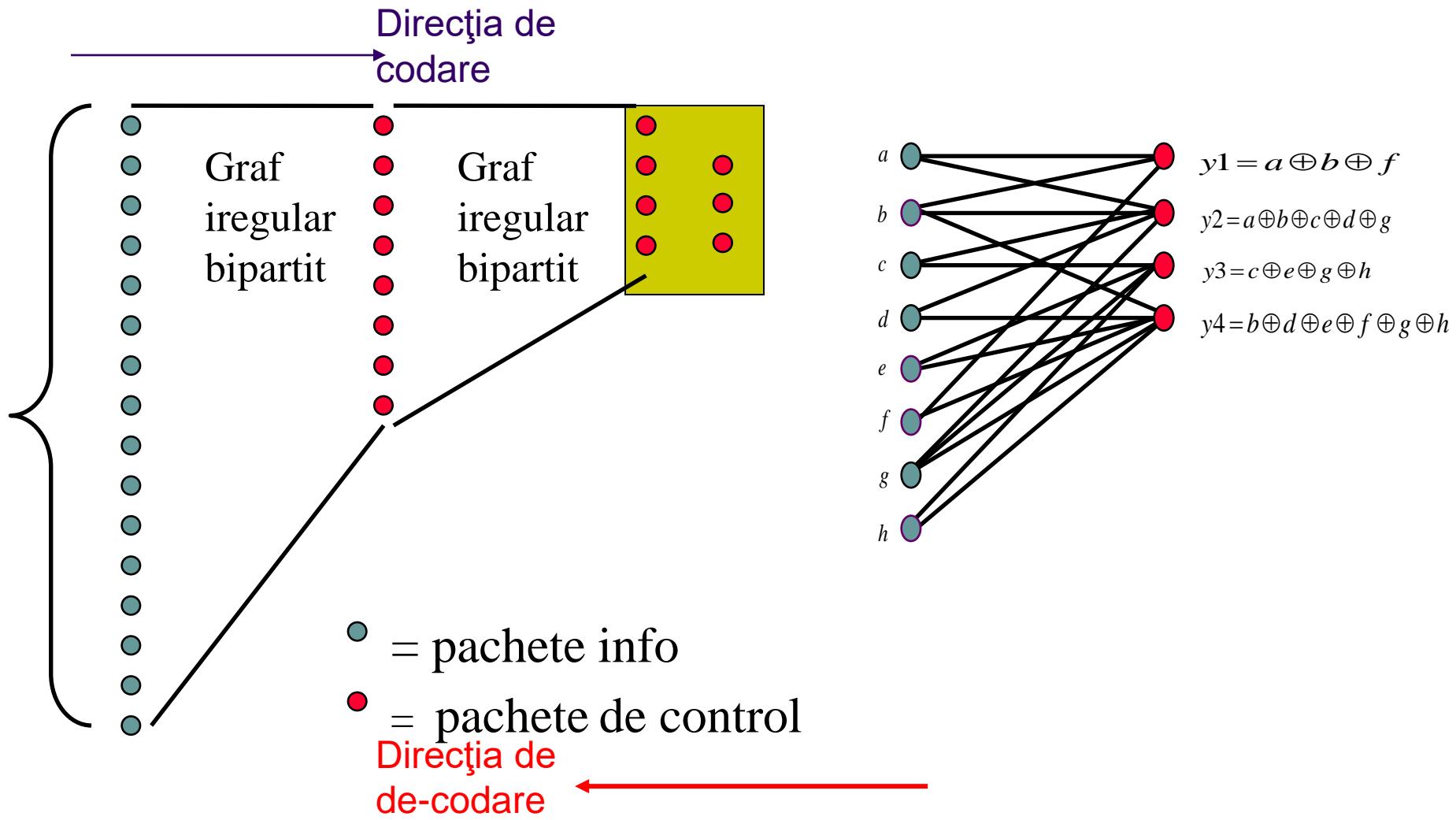
Coduri Raptor

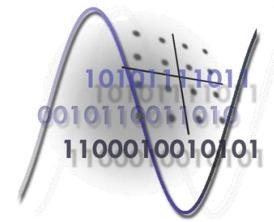
- Parametrii principali de performanță ai codului Raptor sunt definite după cum urmează:
 - Spațiul de memorie: Codurile Raptor necesită spațiu de stocare pentru simbolurile intermediare, Consumul de spațiu al codurilor Raptor este k/R , unde R este rata pre-codului.
 - Overhead: Overheadul este o funcție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





Coduri Tornado





Coduri Raptor(RAPide TORnado)

● ● ● ● ● ● ● ● ● Simoluri de intrare

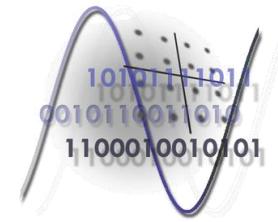
Codare cu cod corector de ștergeri

● ● ● ● ● ● ● ● ● ● simboluri precodate

LT-”light”

■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ Simboluri codate

Coduri Raptor(RAPide TORnado)



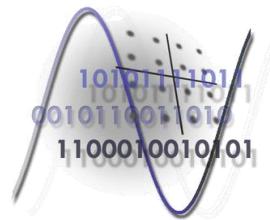
 Simboluri receptionate

decor LT

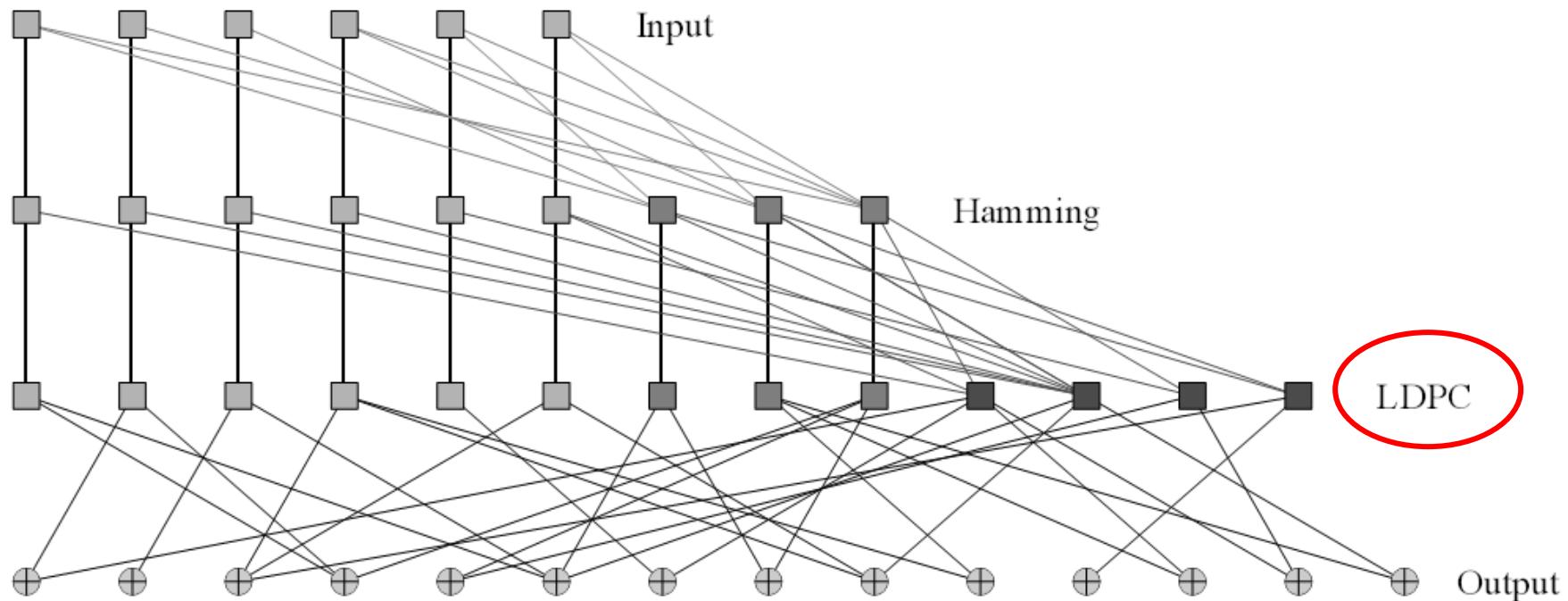
 maxim δ simboluri neacoperite

Codare cu cod corector de ștergeri

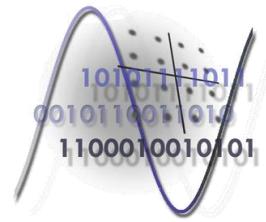
 Simoluri decodate



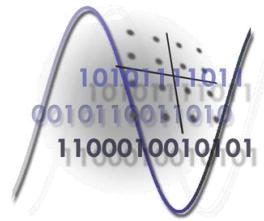
Coduri Raptor(RAPide TORnado)



Codurile LDPC – Low Density Parity Check Codes

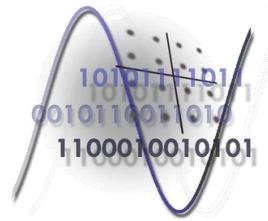


- Codurile LDPC, introduse în 1963 de Robert G. Gallager, reprezintă o familie de coduri bloc liniare
- se obțin pornind de la o matrice „rară” de control a parității H , care conține un număr mic de elemente nenule și un număr mare de elemente de 0 („sparse matrix”)
- Codurile LDPC pot fi privite ca și coduri bloc liniare, descrise de matricea de control a parității H de dimensiune $M \times N$ care satisface ecuația: $Hx^T = 0$ pentru toate cuvintele de cod x



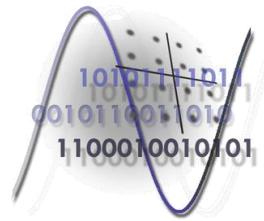
Codurile LDPC

- codurile LDPC au următoarele proprietăți specifice:
 - dimensiunile M și N ale matricii H sunt mult mai mari față de dimensiunile unei matrici de control specifică unui cod Hamming;
 - matricile H sunt definite, prin construcție, într-o formă nesistematică și conțin un număr de elemente de 1 mult mai mic decât numărul de elemente de 0;
 - matricea de control a parității ce definește un cod LDPC $A(j, k)$ are exact j elemente de 1 pe fiecare coloană și exact k elemente de 1 pe fiecare linie.



• Tipuri de coduri LDPC

- construite aleator (random)
- construite pe baza unor structuri regulate („array-based”)
- pe baza unor construcții combinatoriale
- pe baza unor construcții geometrice
- construite pe baza grafurilor
- quasi-ciclice

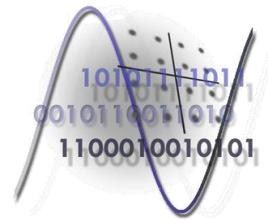


Codarea codurilor LDPC

- Considerând cuvântul de cod $v = [c_0, \dots, c_{M-1}, i_0, \dots, i_{(N-M)-1}]$, biții de control c_m se pot determina, în funcție de biții informaționali i_l , prin rezolvarea sistemului de M ecuații liniare:

$$Hv^T = 0$$

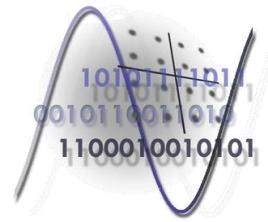
- Această abordare are două dezavantaje majore
 - pentru valori mari al parametrului j , M ia valori mari necesitând un mare volum de calcule, ceea ce conduce la creșterea timpului și/sau resurselor hardware necesare procesării.
 - are nevoie de toți biții informaționali i_l , $l = 0, \dots, (N-M)-1$, în același timp; această cerință induce o întârziere suplimentară de un cuvânt de cod în sistemul de transmisie



Codarea codurilor LDPC

- Matricea H , $(M \times N)$, este împărțită în două matrici D și E , reținând primele M coloane în matricea D și restul de $(N-M)$ coloane în matricea E .
- Cele două matrici au dimensiunile:
 - $D: (M \times M)$
 - $E: ((N-M) \times M)$
- Matricea D este pătrată și are determinant nenul, deci ecuația codării poate fi rescrisă sub forma

$$[H] \cdot [v]^t = [0] \Leftrightarrow [D] \cdot \begin{bmatrix} c_0 \\ \vdots \\ c_{M-1} \end{bmatrix} + [E] \cdot \begin{bmatrix} i_0 \\ \vdots \\ i_{(N-M)-1} \end{bmatrix} = [0] \Rightarrow \begin{bmatrix} c_0 \\ \vdots \\ c_{M-1} \end{bmatrix} = [F] \cdot \begin{bmatrix} i_0 \\ \vdots \\ i_{(N-M)-1} \end{bmatrix}$$

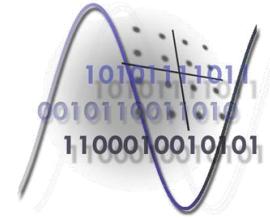


Codarea codurilor LDPC

- Prin dezvoltarea matricii de codare F , ($M \times (N-M)$), dacă notăm cu $[f_i]$ coloanele sale (fiecare din ele un vector de M biți) ecuația de codare poate fi exprimată sub forma:

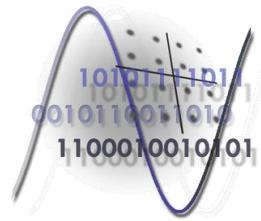
$$[f_0] \cdot i_0 + \dots [f_{(N-M)-1}] \cdot i_{(N-M)-1} = [C]$$

- Matricea F , calculată off-line, este stocată cloană cu cloană în memorie; fiecare bit de informație se înmulțește cu coloana cu același index, iar vectorii-produs rezultați sunt acumulați, astfel obținându-se vectorul biților de control $[C]$



Decodarea codurilor LDPC

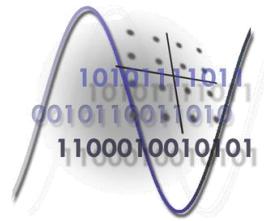
- Algoritmul de decodare MP este un algoritm iterativ care utilizează ca date de intrare probabilitățile *a posteriori* ale bițiilor cuvântului de cod, furnizate de funcție de demapare soft.
- La fiecare iterație, algoritmul modifică valorile acestor probabilități, în funcție de probabilitățile *a posteriori* ale celorlăți biți care intră în aceleasi ecuații de control cu bitul dat. Aceste actualizări sunt făcute iterativ pentru fiecare bit



Decodarea codurilor LDPC

- Matricea de control a unui cod LDPC (5,3,4)

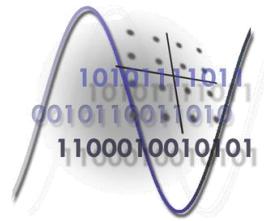
$$H = \left[\begin{array}{cccccccccccccccccccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ \hline 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ H = & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array} \right]$$



Decodarea codurilor LDPC

- sistemul de ecuații de control a parității, construit pe baza relaiei:

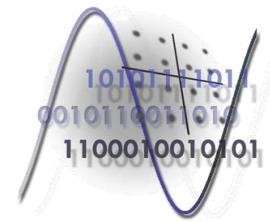
$$\begin{bmatrix}
 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
 \hline
 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
 \hline
 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
 \hline
 m_1 & & & & & & & & & & & & & & & & & & & \\
 m_2 & & & & & & & & & & & & & & & & & & & \\
 m_3 & & & & & & & & & & & & & & & & & & & \\
 m_4 & & & & & & & & & & & & & & & & & & & \\
 m_5 & & & & & & & & & & & & & & & & & & &
 \end{bmatrix} = [0]$$



Decodarea codurilor LDPC

- Pentru a obține biții c_i asociați cuvântului binar $m = \{m_1, m_2, \dots, m_K\}$ trebuie rezolvat sistemul următor:

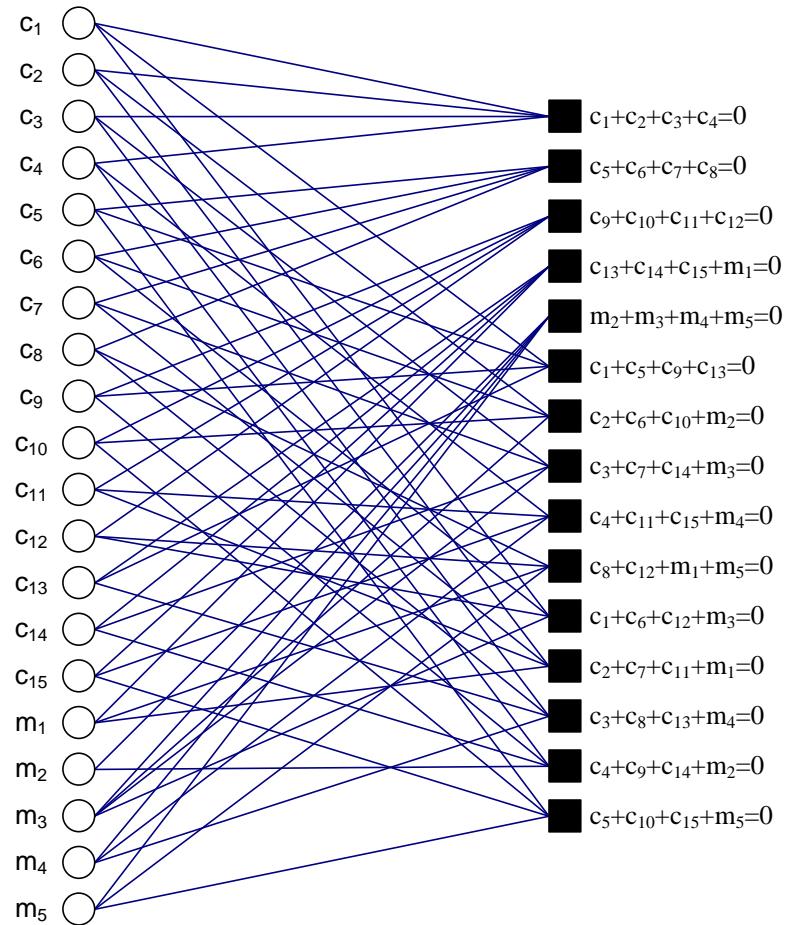
$$\left\{ \begin{array}{l} c_1 + c_2 + c_3 + c_4 = 0 \\ c_4 + c_6 + c_7 + c_8 = 0 \\ c_9 + c_{10} + c_{11} + c_{12} = 0 \\ c_{13} + c_{14} + c_{15} + m_1 = 0 \\ m_2 + m_3 + m_4 + m_5 = 0 \\ c_1 + c_5 + c_9 + c_{13} = 0 \\ c_2 + c_6 + c_{10} + m_2 = 0 \\ c_3 + c_7 + c_{14} + m_3 = 0 \\ c_4 + c_{11} + c_{15} + m_4 = 0 \\ c_8 + c_{12} + m_1 + m_5 = 0 \\ c_1 + c_6 + c_{12} + m_3 = 0 \\ c_2 + c_7 + c_{11} + m_1 = 0 \\ c_3 + c_8 + c_{13} + m_4 = 0 \\ c_4 + c_9 + c_{14} + m_2 = 0 \\ c_5 + c_{10} + c_{15} + m_5 = 0 \end{array} \right.$$

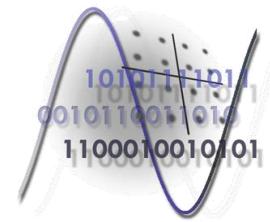


Decodarea codurilor LDPC

- Graful bipartit asociat codului LDPC (5,3,4) este:

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ H = & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline \end{bmatrix}$$

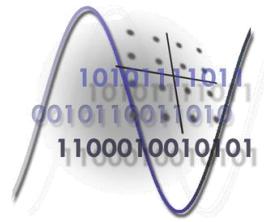




Decodarea codurilor LDPC

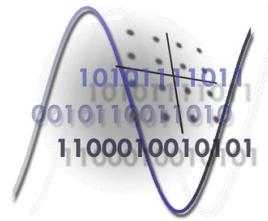
- **Pasul 0: inițializare**

nodurile de bit sunt inițializate cu probabilitățile *aposteriorii* inițiale determinate de blocul de demapare pe baza semnalului recepționat, P_j^0 (probabilitatea ca bitul j din cuvântul recepționat să fie 0) și P_j^1 (probabilitatea ca bitul j din cuvântul recepționat să fie 1)

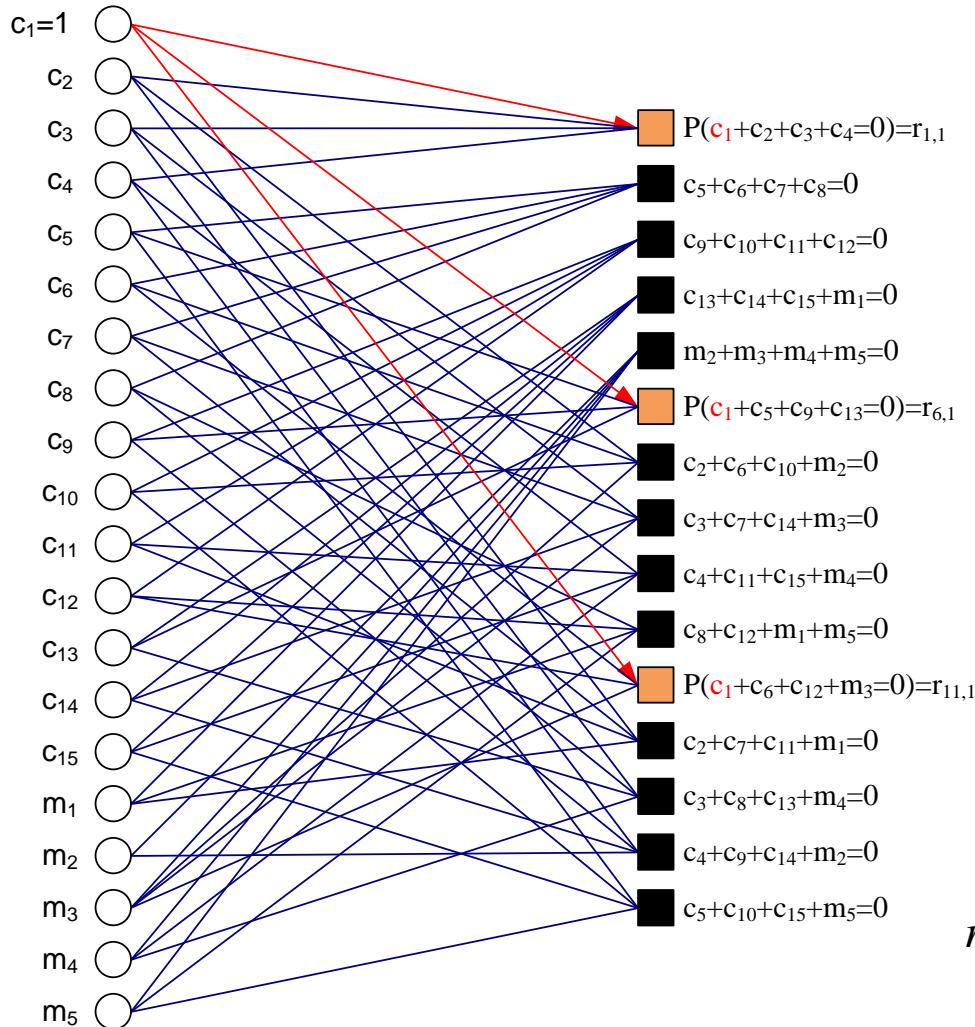


Decodarea codurilor LDPC

- **Pasul 1 actualizarea nodurilor de control**
- se determină probabilitățile $r_{m,n}^x$, care reprezintă probabilitatea ca ecuația de control m să fie satisfăcută dacă bitul n are valoarea x



Decodarea codurilor LDPC

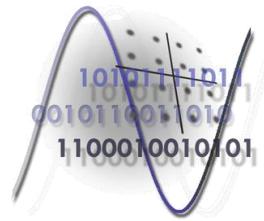


Probabilitatea ca prima ecuație să fie satisfăcută, dacă $c_1 = 1$ este:

$$\begin{aligned}
 & P(c_1 + c_2 + c_3 + c_4 = 0 | c_1 = 1) \\
 & = P(c_2 + c_3 + c_4 = 1) = \\
 & = P_2^1 \cdot P_3^0 \cdot P_4^0 + P_2^0 \cdot P_3^1 \cdot P_4^0 + \\
 & + P_2^0 \cdot P_3^0 \cdot P_4^1 + P_2^1 \cdot P_3^1 \cdot P_4^1
 \end{aligned}$$

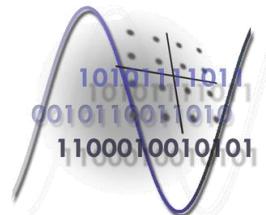
Se poate arăta că probabilitatea ca din k biți independenti un număr impar de biți să fi nenul este:

$$r_{m,n}^1 = \frac{1 - \prod_{\substack{l=1 \\ l \neq n}}^{k_m} (1 - 2q_{l,m}^1)}{2} = \frac{1}{2} \left(1 - \prod_{\substack{l=1 \\ l \neq n}}^{k_m} (q_{l,m}^0 - q_{l,m}^1) \right)$$

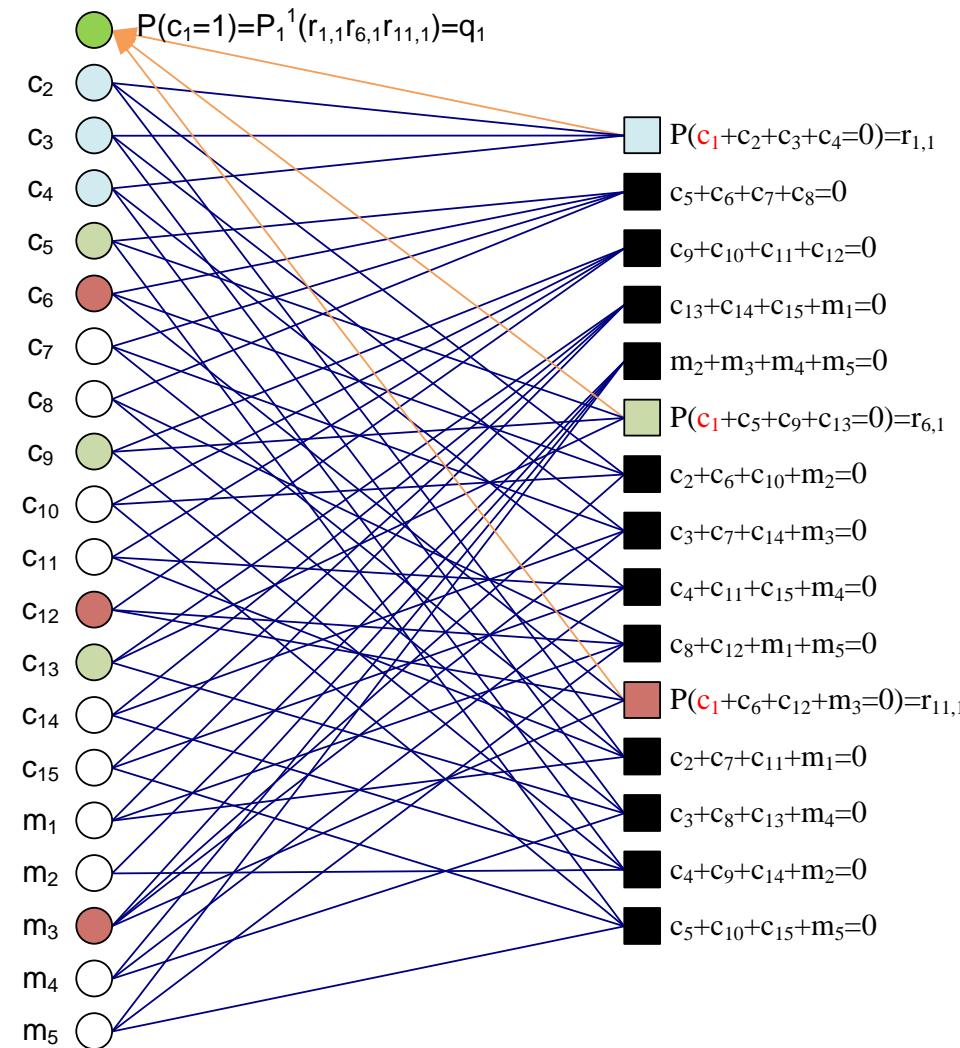


Decodarea codurilor LDPC

- **Pasul 2 Actualizarea nodurilor de bit**
- La sfârșitul fiecărei iterații se determină probabilitățile a posteriori pentru fiecare bit, adică probabilitatea P ca bitul transmis pe poziția n să fie „1” condiționat de setul de simboluri y recepționate și de evenimentul S ca cele j_n ecuații de control, în care intră bitul n , să fie satisfăcute.
- Fiindcă ecuațiile de control sunt independente, probabilitatea ca toate cele j_n ecuații de control în care intră bitul n să fie satisfăcute este produsul probabilităților individuale ca fiecare din aceste ecuații să fie satisfăcută



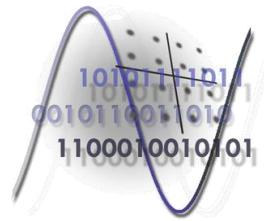
Decodarea codurilor LDPC



$$P(x_n = 1 | S, \{y\}) = \alpha_n \cdot P_n^1 \cdot \prod_{i=1}^{j_n} (r_{i,n}^1)$$

- În ecuația de mai sus constanta α se alege astfel încât

$$P(x_n = 1 | S, \{y\}) + P(x_n = 0 | S, \{y\}) = 1$$

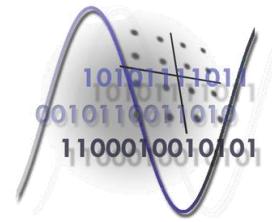


Decodarea codurilor LDPC

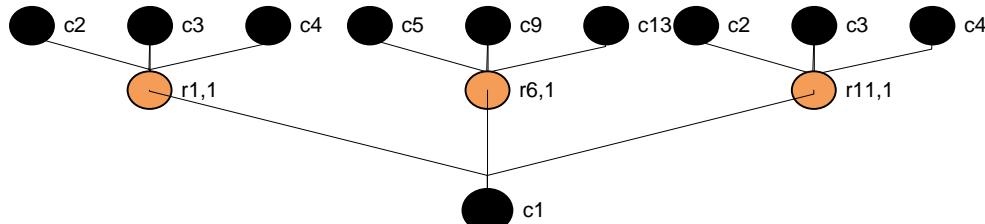
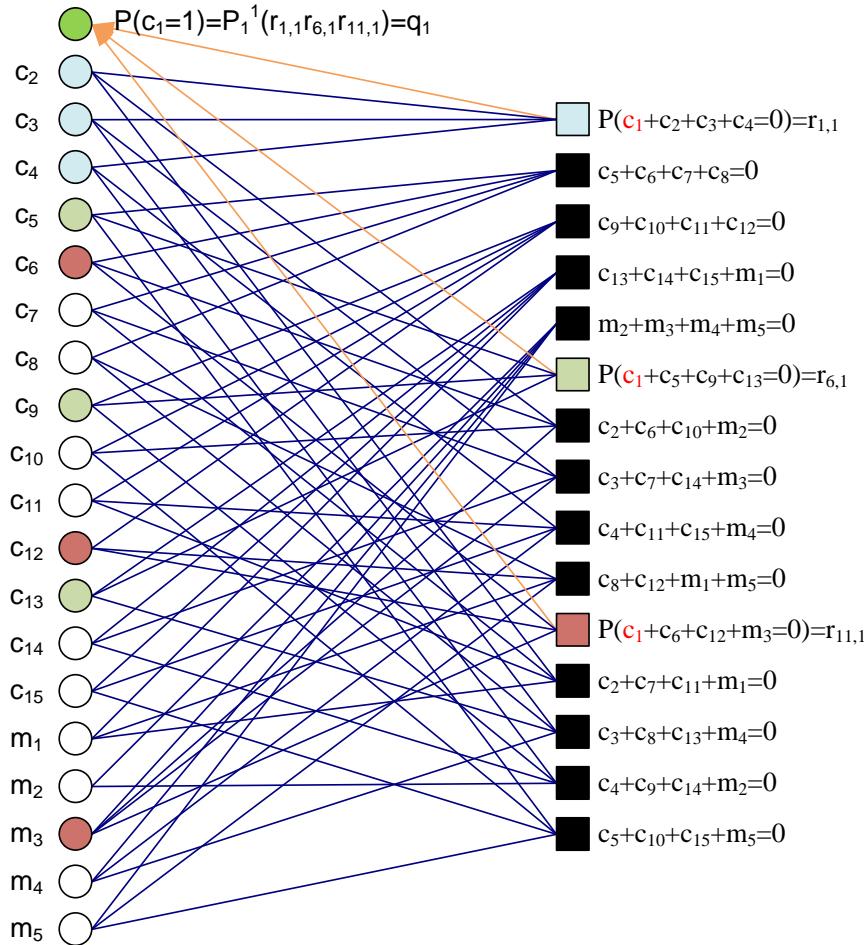
- **Pasul 3 Decizia hard a biților decodați și calcularea sindromului**
 - Dacă cuvântul binar x în care bitul n are valoarea 1, cu respectarea condiției $P(x_n=1|S,\{y\})>0.5$, satisfacă ecuația $Hx^T = 0$, algoritmul de decodare este încheiat.
 - Dacă sindromul nu este nul, adică Hx^T nu este zero, se efectuează o nouă iterare.
 - Acest algoritm iterativ continuă până când se găsește un cuvânt de cod valid, sau până când graful de decodare va deveni un arbore cu l nivele, cazul în care se consideră că decodarea a eşuat

$$P(x_n=1|S,\{y\}) \geq 0.5$$

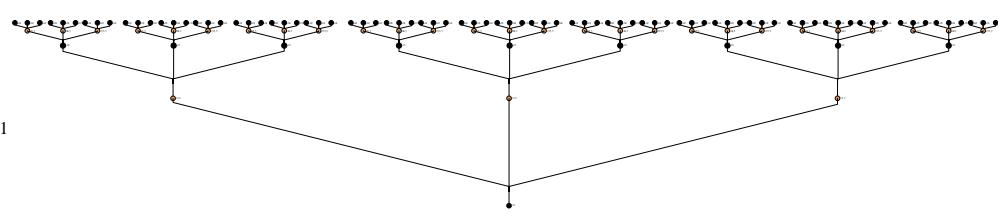
Decodarea codurilor LDPC



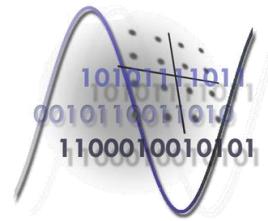
- Graful vecinilor



Graful vecinilor nodului de bit c_1 după prima iteratie



Graful vecinilor nodului de bit c_1 după adoua iteratie

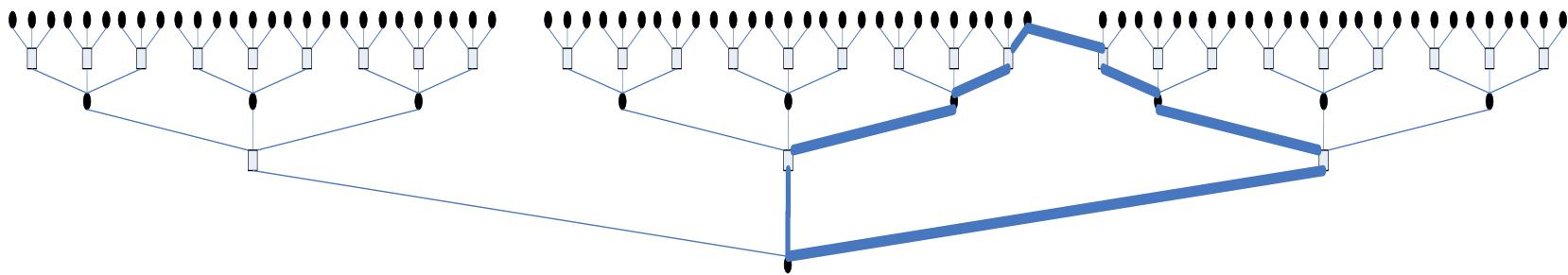


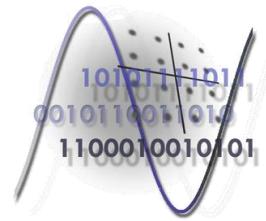
Decodarea codurilor LDPC

- ca toți vecinii de gradul L unui nodului să fie independenți (să apară o singură dată în graful vecinilor) după iteratăția L , lungime cuvântului de cod ar trebui să fie :

$$M \geq [(k-1) \cdot (j)]^L$$

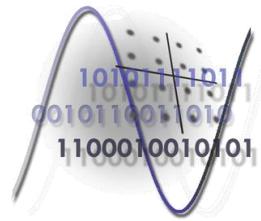
- subgraful vecinilor nodului de bit n rezultat în urma decodării iterative, după un număr de I iterății, nu va fi un arbore fără bucle





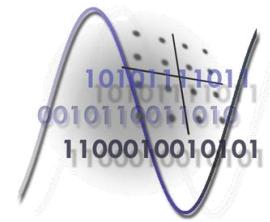
Decodarea codurilor LDPC

- Trebuie reținut că acest algoritm nu încearcă să gasească cel mai apropiat cuvânt de cod, față de secvența recepționată, ci încearcă să corecteze fiecare bit al secvenței recepționate, folosind informațiile oferite de ceilalți biți cu care bitul dat intră în ecuațiile de control. Sindromul este folosit ca o verificare (CRC).
- Datorită acestui fapt, numărul de biți eronați după o decodare nereușită, este mai mic decât numărul bițiilor eronați înaintea decodării (spre deosebire de codurile bloc ciclice sau de cele convecționale).
- Numărul maxim de iterații este un parametru care se stabilește în funcție de durata permisă pentru decodare de aplicația în care este utilizat codul



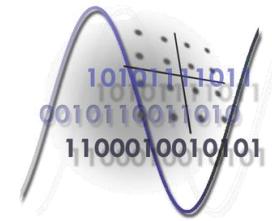
Diferite tipuri de coduri Raptor

- Coduri LT
 - Codurile LT pot fi considerate coduri Raptor de forma $(k, F_2^k, \Omega(d))$, unde F_2^k este practic un cod de lungime k , care asociază simbolurile de intrare la simbolurile de ieșire
 - Inexistența puterii de corecție a pre-codului impune utilizarea unei distribuții $\Omega(d)$ sofisticate, cu complexitatea logaritmică a codării și decodării (vezi cursul anterior)



Diferite tipuri de coduri Raptor

- PCO (Pre-Code-Only)
 - Aceste coduri reprezintă extremitatea cealaltă a codurilor *Raptor*, adică au un precodor sofisticat și o distribuție de ieșire foarte simplă de forma $\Omega(1)=1$
 - Algoritmul de decodare recepționează un număr de m simboluri, din care se decodează n simboluri intermediare independente (este posibilă recepționarea de două sau de mai multe ori ale aceluiași simbol), și pre-decodorul pe baza acestor n simboluri intermediare determină cele k simboluri informaționale

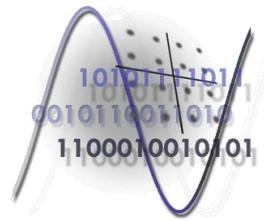


Diferite tipuri de coduri Raptor

- Coduri optimale (Asymptotically Optimal Raptor Codes)
 - Costuri de codare și decodare constante, overheadul necesar decodării tinde spre unu
 - Este format dintr-un precodor LDPC și o distribuție de graduri de forma:

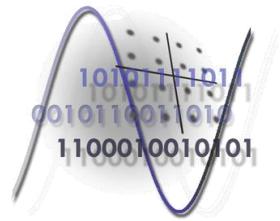
$$\Omega(d) = \frac{1}{\mu+1} \left(\mu d + \frac{d^2}{1 \cdot 2} + \cdots + \frac{d^D}{(D-1) \cdot D} + \frac{d^{D+1}}{D} \right)$$

- Unde $\mu \leq \varepsilon/2$ și $D \geq 2/\varepsilon$ astfel se poate obține un cod LDPC cu decodor MP liniar și un overhead ε

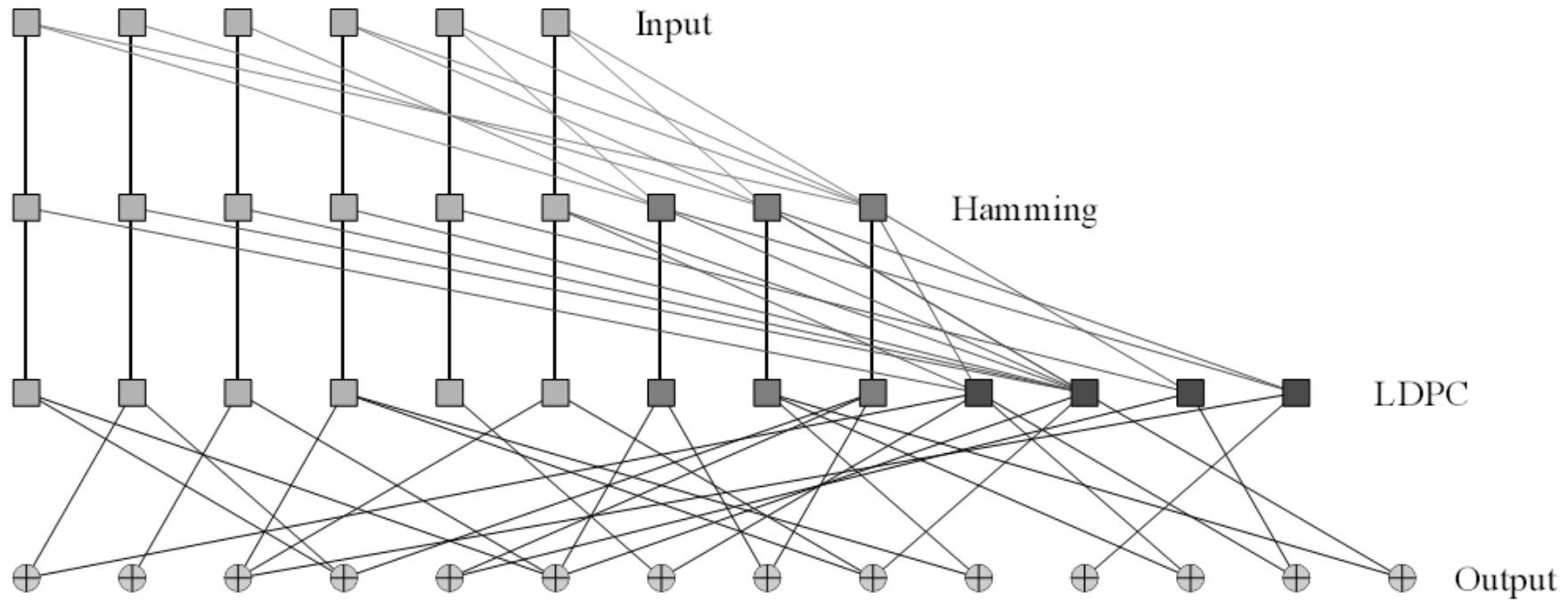


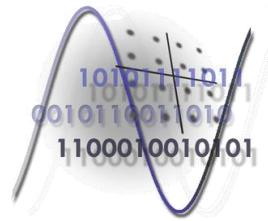
Diferite tipuri de coduri Raptor

- Cod eficient și implementabil (practic)
 - Este format din două precodoare:
 - Precodor 1 cod Hamming
 - Precodor 2 cod LDPC
 - LT cu distribuția gradurilor:
$$\Omega(d) = 0.008d + 0.494d^2 + 0.166d^3 + 0.073d^4 + 0.083d^5 + 0.056d^8 + 0.037d^9 + 0.056d^{19} + 0.025d^{65} + 0.003d^{66}$$
 - Codul rezultat are probabilitatea de eroare de bloc de maxim 10^{-14} pentru $k \geq 2^{16} = 65536$



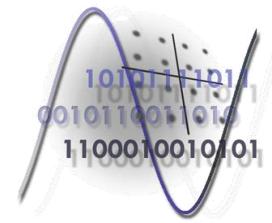
Cod Raptor practic





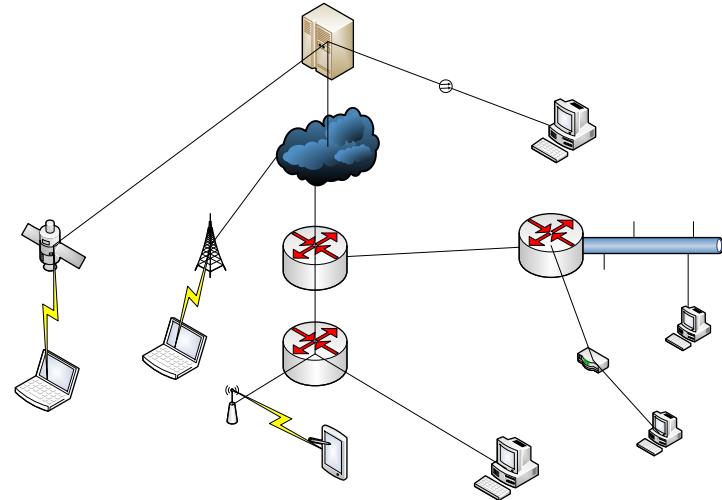
Utilizarea Codurilor DF

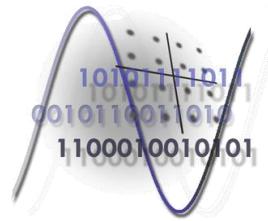
- Multicast
- Descărcare în paralel
- One-to-Many TCP
- Video streaming
- Transmisii în Overlay Networks
- Stocare distribuită
- Rutare dispersivă



Multicast

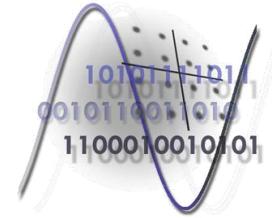
- DF a fost conceput pentru a asigura un multicast fiabil
- Sursa transmite într-un moment dat același informație către toți utilizatorii
- Nu este nevoie de feedback
- Fiecare utilizator poate să îintrerupe și să reîncepe achiziționarea datelor fără să informeze sursa despre acest lucru





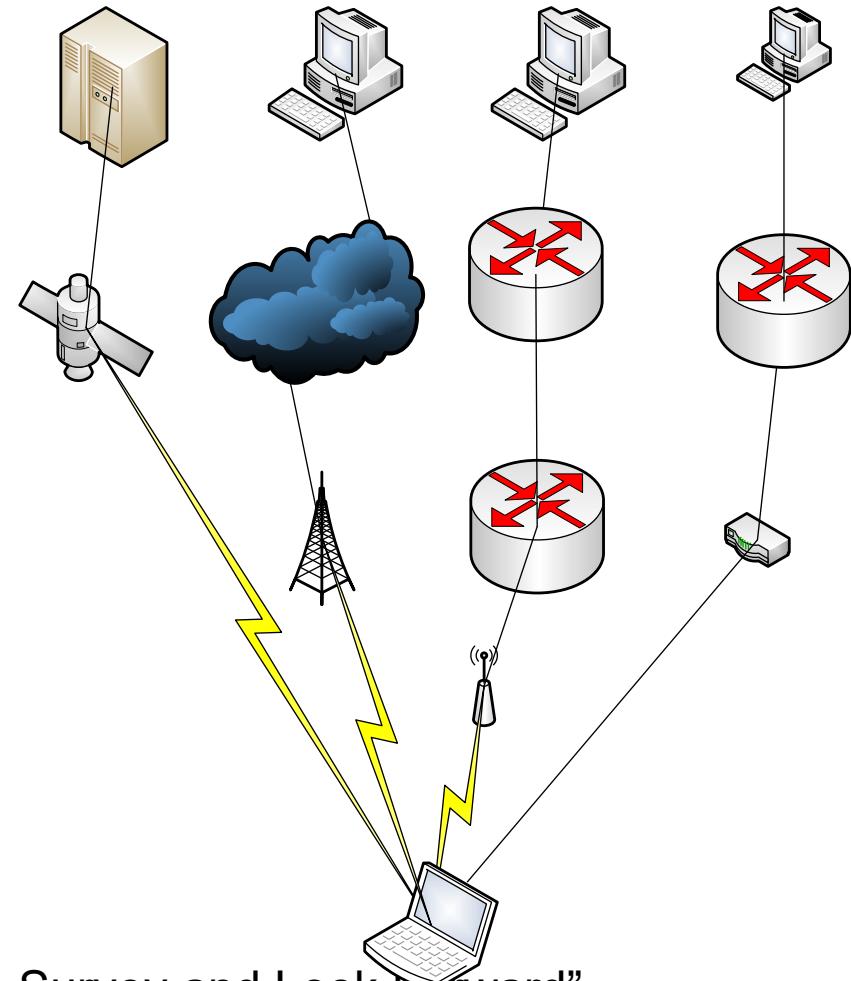
Transmisiile punct la punct

- TCP are “probleme” în cazul transmisiilor la distanțe mari
 - Timpul de reacție a sistemului este mare
 - Fereastra glisantă relativ mare, rezultă memorie mare
 - Retransmisiile introduc alte complicații
- În cazul utilizării unei cod DF destinația trebuie să transmită numai o cerere de transmisie și o notificare despre decodarea reușită la sfărșitul transmisiei
 - Sursa nu are nici o informație despre canal
 - poate să “inunde” cu pachete primele hopuri din rețea



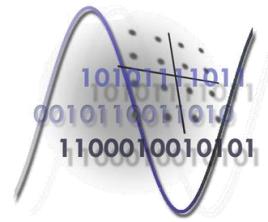
Descărcare în paralel

- DF poate simplifica unui utilizator descărcarea unui bloc de informații, în paralel de la mai multe surse
- Fiecare sursă transmite secvența codată
- Destinația nu trebuie să gestioneze informațiile recepționate pe diferite legături
- Trebuie să recepționeze numărul minim de pachete necesare decodării, indiferent pe ce legătură ajunge asta



[6] Michael Mitzenmacher, "Digital Fountains: A Survey and Look Forward",

<http://www.eecs.harvard.edu/~michaelm/postscripts/itw2004.pdf>

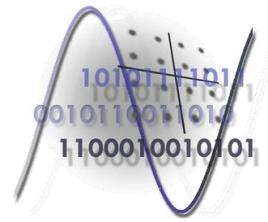


One-to-Many TCP

- Multicastul peste TCP însemenă că sursa, pentru fiecare legătură trebuie să țină evidența pachetelor transmise, pachetelor confirmate respectiv neconfirmate.
- Această evidență ocupă o cantitate de memorie pentru fiecare legătură, rezultă limitarea numărului de conexiuni deservite
- Cu DF serverul nu trebuie să gestioneze independent fiecare conexiune
- Într-un moment dat serverul trimite același pachet la toți utilizatori indiferent dacă a primit ACK sau NACK

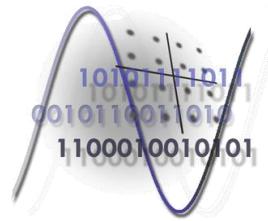
[6] Michael Mitzenmacher, “Digital Fountains: A Survey and Look Forward”,

<http://www.eecs.harvard.edu/~michaelm/postscripts/itw2004.pdf>



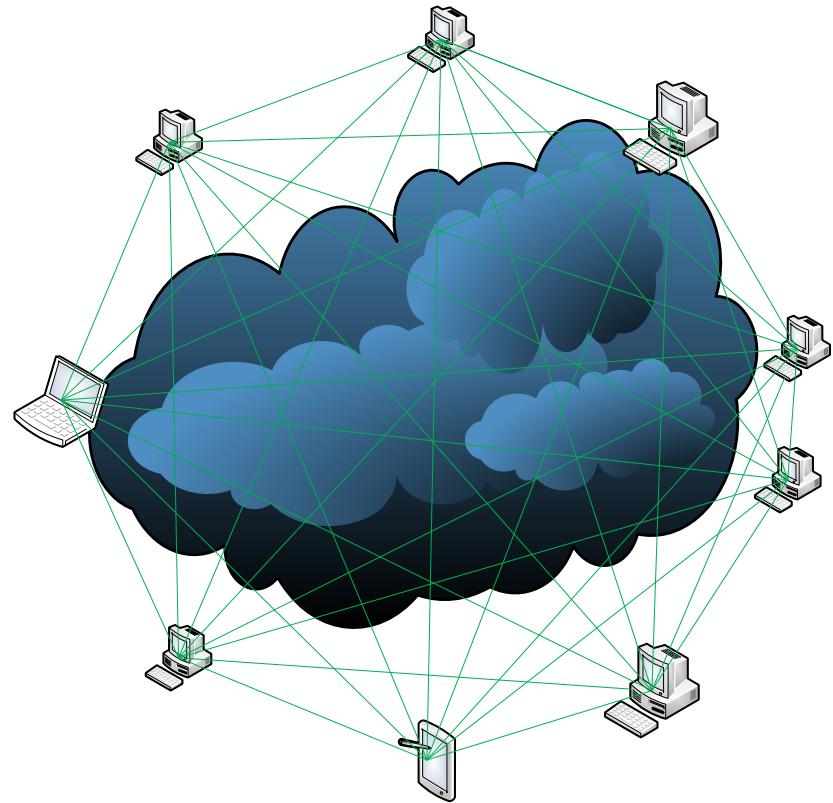
Video streaming

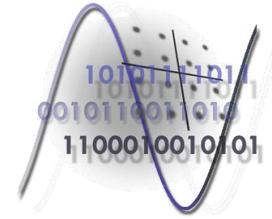
- Aplicație în timp real
- Dacă se consideră că întregul fișier reprezintă cele k simboluri de intrare, decodare poate să inceapă după recepționarea a cel puțin k pachete codate – nu este utilizabil pt video streaming
- Fișierul se împarte în segmente, fiecare segment este codat separat
- În timp ce primul segment este redat, se receptionează al doilea segment....
- Probleme la proiectarea sistemului:
 - Dimensiunea segmentelor
 - Numărul de pachete codate necesare (rata de transmisie)
 - Combinare cu multicastul classic (pt on demand)



Overlay Networks

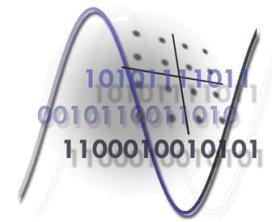
- Fiecare utilizator transmite pachetele codate în loc de pachete informaționale
- Un utilizator când a reușit să decodeze informația poate să genereze alt set de pachete codate
- Probleme ex:
 - Dacă sursa inițială dispare din rețea înainte ca să ajungă în rețea un număr suficient de pachete codate pt reconstruirea informației originale





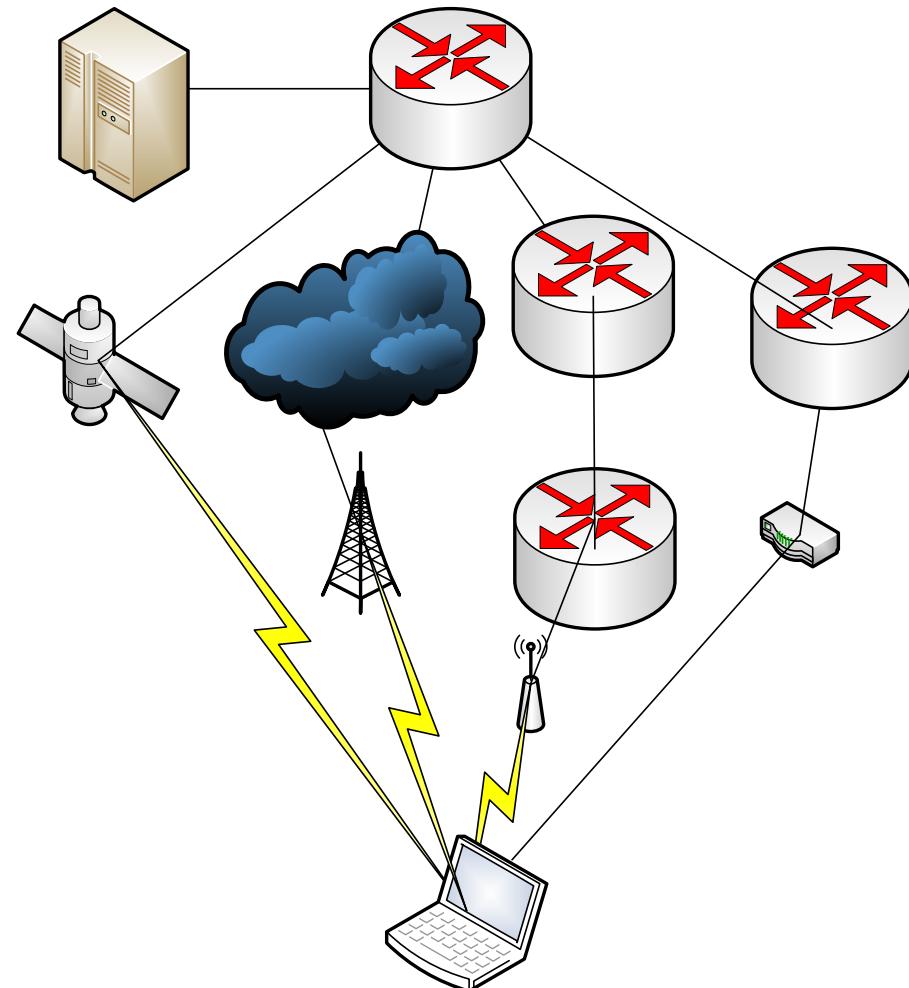
Stocare distribuită

- Fișierul este împărțit în n blocuri de dimensiune fixă, aceste blocuri sunt repetate și distribuite prin sistem.
- Fiecare bloc trebuie repetat de m ori pentru a tolera căderea a $m-1$ servere
- În cazul utilizării DF calculează $n+m$ pachete codate, și acestea se stochează.
- Clientul trebuie să descarce numai orice $n \cdot (1+\varepsilon)$, $\varepsilon \rightarrow 0$, blocuri, și din acestea se pot calcula cele n blocuri ale fișierului.
 - Se simplifică găsirea și descărcarea unui anumit bloc.
 - Accesul lent și pierderile de pachete sunt eliminate
 - Se obține un spațiu de stocare stabil pentru arhivare.



Rutare dispersivă

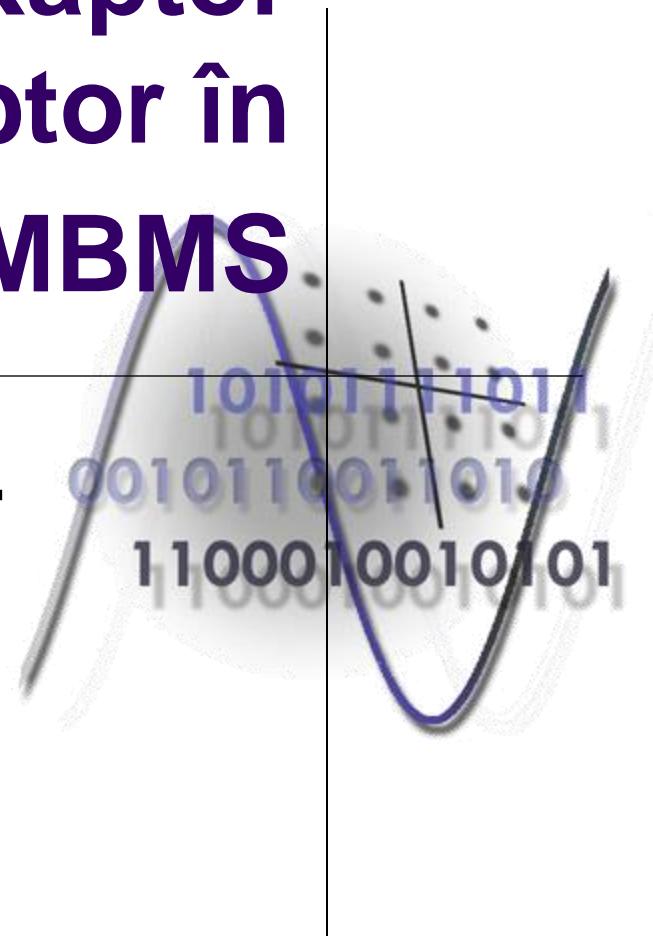
- Spre deosebire de procedurile de rutare convenționale, care rutează un mesaj de-a lungul unei anumite căi între sursă și destinație, mecanismele de rutare dispersive (multicale) subdivizează mesajul și îl împărăștie pe câteva căi din rețea

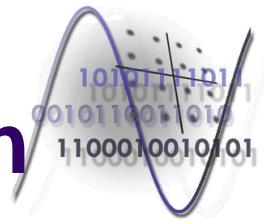


Algoritmi practici de codare și decodare Raptor

Utilizarea codurilor Raptor în MBMS

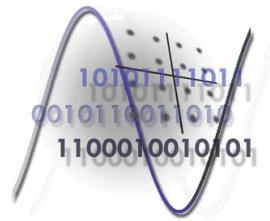
TACCFDRT Curs 4





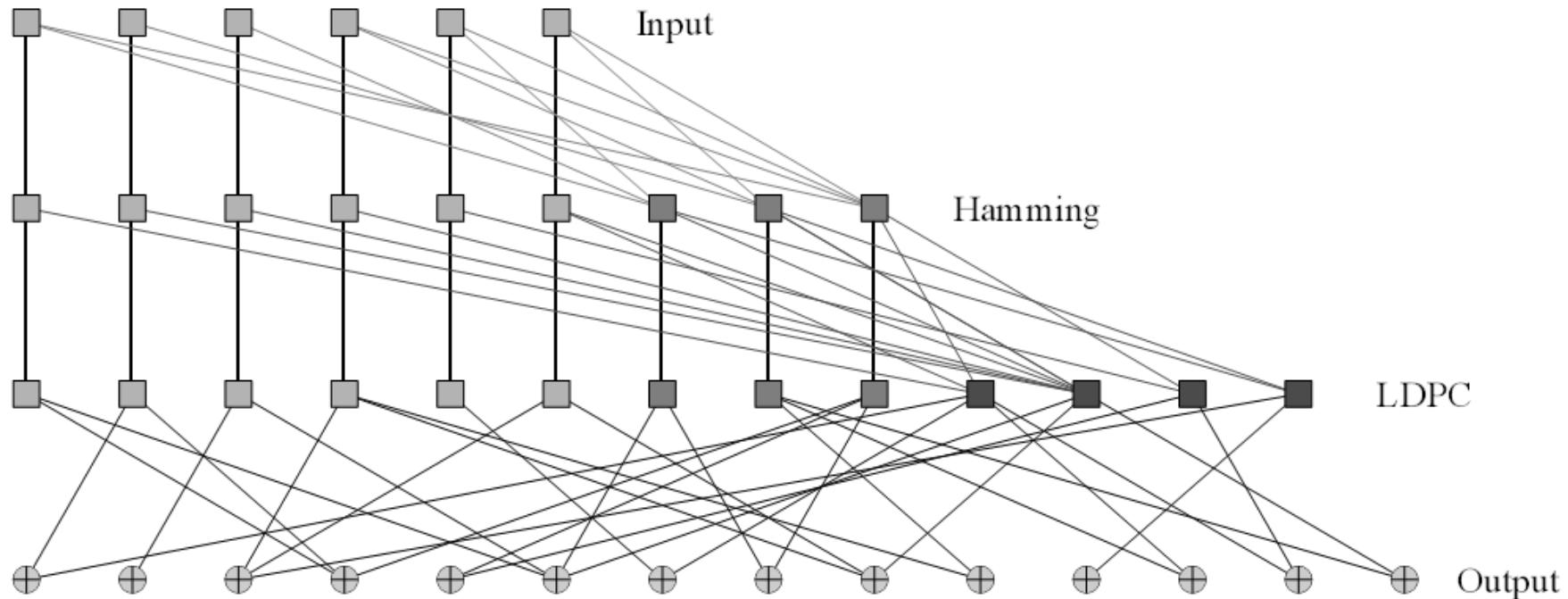
Tehnici de codare de tip Digital Fountain

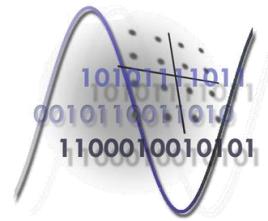
- Coduri Raptor sistematice
 - Algoritm de codare
 - Algoritm de decodare
- Servicii MBMS
- Utilizarea codurilor raptor în MBMS



Coduri Raptor

- Codurile Raptor prezentate nu sunt sistematice!!



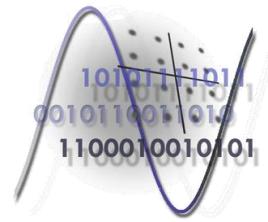


Coduri Raptor Sistematice

- Se consideră un cod Raptor ($k, C, \Omega(d)$), care pentru decodare “sigură” necesită un overhead ϵ
- Pachetele (simbolurile) informaționale se notează cu $x = x_1, x_2, x_3, \dots, x_k$
- Pre-codorul C generează un cuvânt de cod cu lungime n , pe baza relației:

$$u^T := G \cdot x^T$$

- G este matricea generatoare a codului
- u este cuvântul de cod la ieșirea pre-codorului, și reprezintă simbolurile de intrare în codorul LT

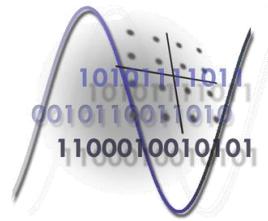


Coduri Raptor Sistematice

- Pentru obținerea unui simbol de ieșire se generează un grad d după distribuția $\Omega(d)$ și se selectează uniform d pachete din u
 - Simbolul de la ieșire se obține prin înmulțire vectorului u^T cu un vector aleator v, de lungime n care conține d valori de 1 și restul 0

$$z_t = v_t \cdot u^T$$

- cu z_t se notează simbolul codat în momentul t iar vectorul v_t este vectorul asociat simbolului codat z_t



Coduri Raptor Sistematice

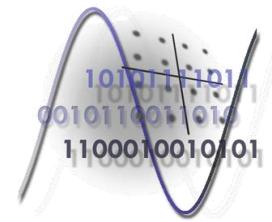
- La un set de N simboluri codate se poate asocia o matrice S cu dimensiuni NxN, a cărei linii sunt vectorii de codare asociate simbolurilor de ieșire
- Putem scrie ca:

$$z^T = S \cdot u^T$$

unde Z este vectorul format din cele N simboluri de ieșire

- Dacă se tine cont de relația de codare a precodorului, se obține:

$$z^T = S \cdot G \cdot x^T$$

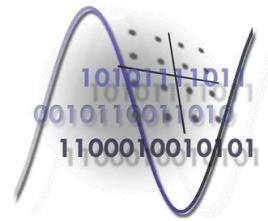


Coduri Raptor Sistematice

- Pentru decodarea codului Raptor trebuie rezolvat sistemul de ecuații dat de:

$$z^T = S \cdot G \cdot x^T$$

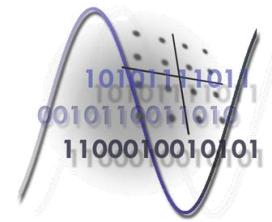
- Acest sistem de ecuații are soluție numai dacă matricea $S \cdot G$ are rangul egal cu k
- Prin eliminare Gaussiană se identifică k linii a matricei S , cu indecsă i_1, i_2, \dots, i_k , a.î. matricea A formată din aceste linii, înmulțită cu matricea G să fie inversabilă
- Se notează cu $R = A \cdot G$



Coduri Raptor Sistematice

- Algoritmul de generare a matricei R (necesar pentru implementarea codorului sistematic)
 - se generează $k(1+\varepsilon)$ vectori v_t conform distribuției $\Omega(d)$, acești vectori reprezintă liniile matricei S
 - Se calculează produsul $S \cdot G$
 - Prin eliminare Gaussiană se determină indecsăi i_1, i_2, \dots, i_k astfel încât submatricea R a matricei $S \cdot G$ formată din liniile indicate de acești indecsăi să fie inversabil.
 - R este o matrice de $k \times k$

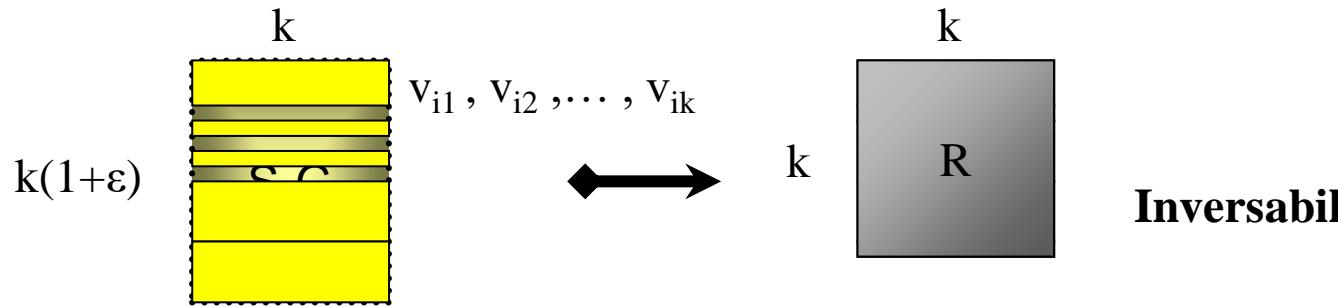
Amin Shokrollahi, "Raptor Codes", IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 52, NO. 6, JUNE 2006



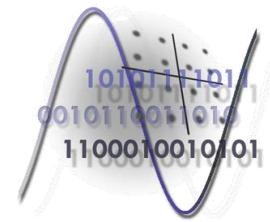
Coduri Raptor Sistematice

$$k(1+\varepsilon) \times n \text{ } S \times n \text{ } G = k(1+\varepsilon) \text{ } S \cdot G$$

A horizontal line separates this diagram from the one below.



Amin Shokrollahi, "Raptor Codes", IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 52, NO. 6, JUNE 2006

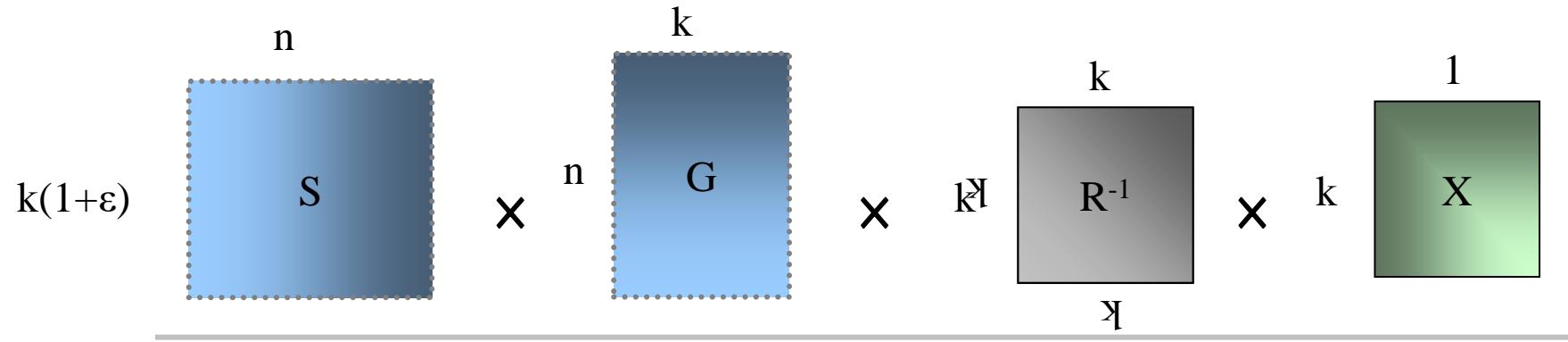
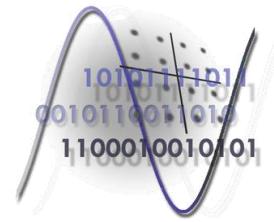


Coduri Raptor Sistematice

- ***Algoritmul de codare a codurilor Raptor sistematice***
 - se generează vectorul $y = (y_1, y_2, \dots, y_k)$ astfel:
$$y^T = R^{-1}x^T$$
 - Simbolurile y se aplică la intrarea precodorului și se obțin simbolurile u
$$u^T = G \cdot y^T$$
 - Se calculează $z_i := v_i \cdot u^T$ pentru $1 \leq i \leq k(1+\varepsilon)$
 - Se generează simbolurile de ieșire $z_{k(1+\varepsilon)+1}, z_{k(1+\varepsilon)+2}, \dots$ pe baza codului LT ($n, \Omega(d)$)

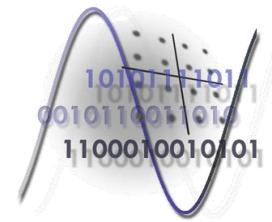
Amin Shokrollahi, "Raptor Codes", IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 52, NO. 6, JUNE 2006

Coduri Raptor Sistematice-Codare



Intrarea prima dată este înmulțit cu matricea de decodare și după aceea este codat, deci ieșirile $z_{i1}, z_{i2}, \dots, z_{ik}$ sunt egale cu intrările x_1, x_2, \dots, x_k

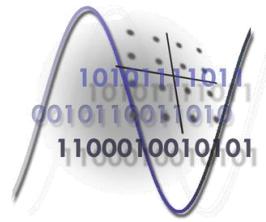
Amin Shokrollahi, "Raptor Codes", IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 52, NO. 6, JUNE 2006



Coduri Raptor Sistematice

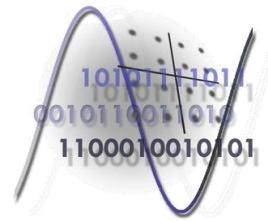
- ***Algoritmul de decodare a codurilor Raptor sistematice***
 - Se decodează codul Raptor normal, se obțin simbolurile $y=(y_1, y_2, \dots, y_k)$
 - Vectorul y înmulțește matricea R și se obține
$$x^T = Ry^T$$
- Vectorul $x=(x_1, x_2, \dots, x_k)$ reprezintă simbolurile informaționale recepționate

Amin Shokrollahi, "Raptor Codes", IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 52, NO. 6, JUNE 2006



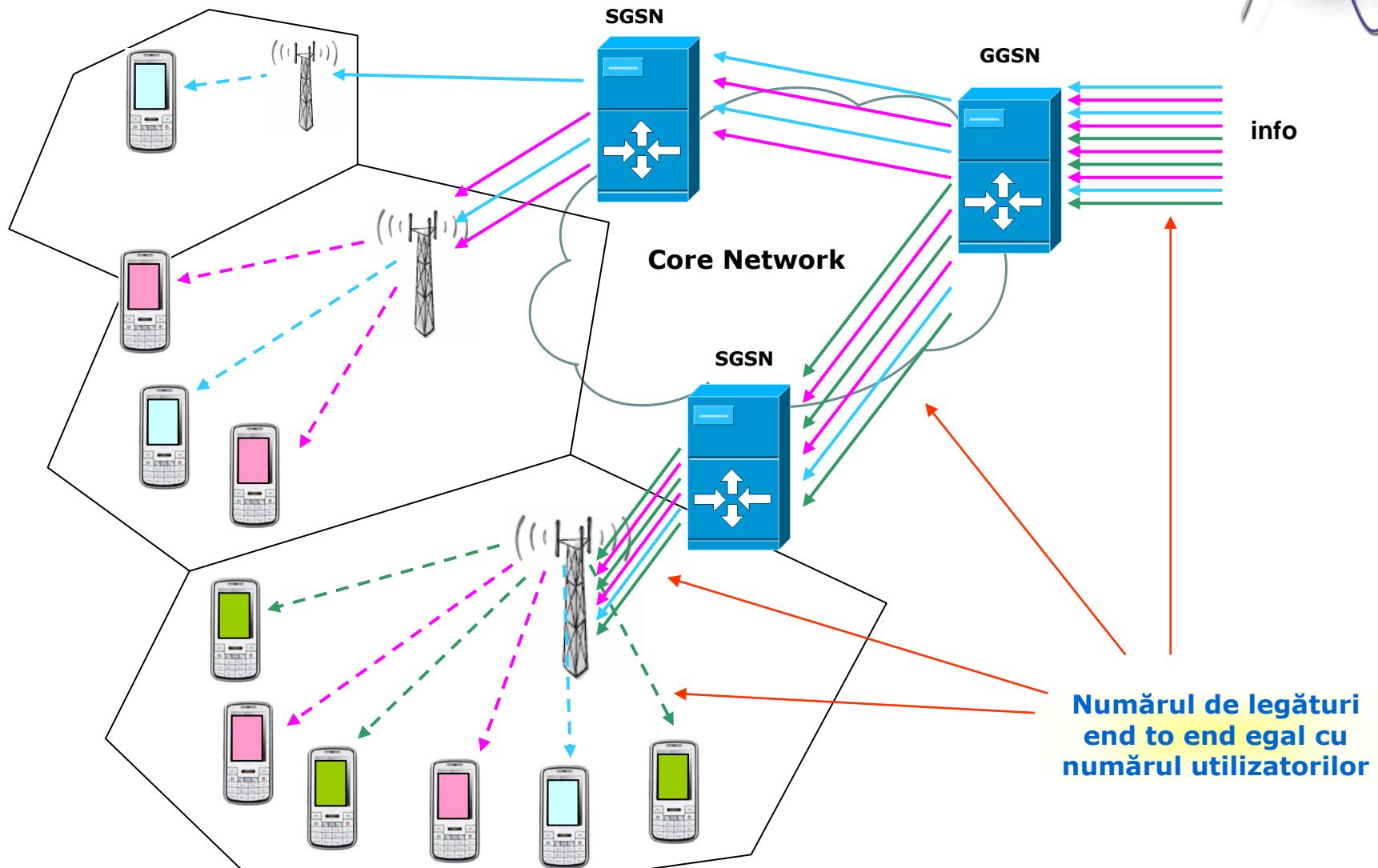
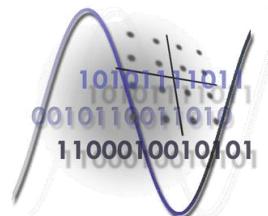
MBMS

- MBMS - **M**ultimedia **B**roadcast/**M**ulticast **S**ervice
 - Scopul este transmisia informației în mod eficient de la o sursă la mai multe destinații mobile
 - Canal foarte variabil
 - Resurse limitate
 - Multe pachete pierdute
 - Sistemele celulare sunt optimizate pentru transmisii punct la punct
 - Nu se utilizează caracterul “broadcast” a canalului radio



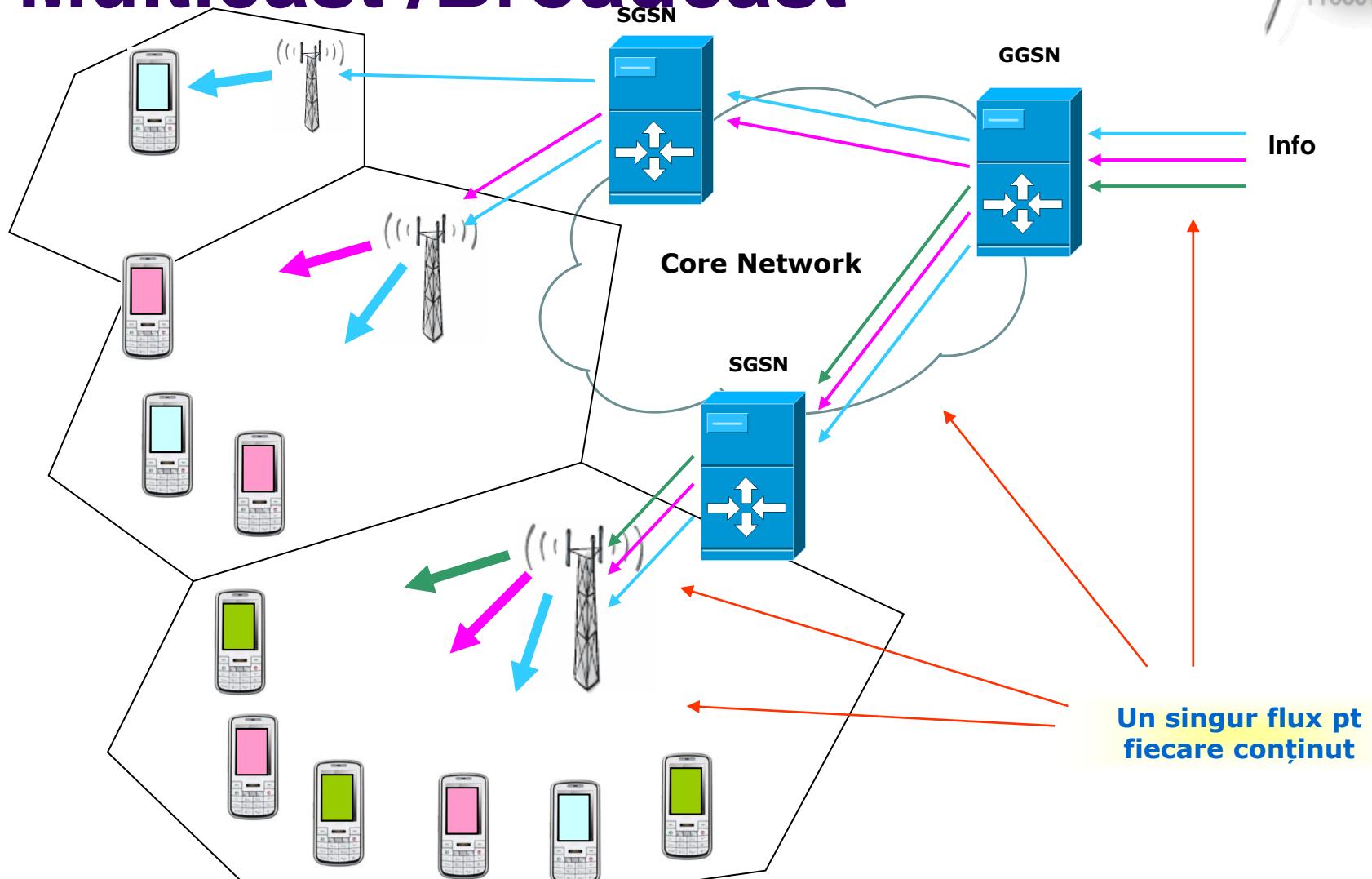
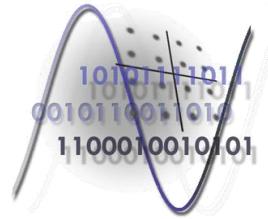
- În cazul unor aplicații mai mulți utilizatori recepționează același date în același timp: ar fi benefic pentru rețea să se transmită informația o singură dată pe o anumită legătură
- Pentru transmiterea informației la mai mulți utilizatori poate fi utilizat Cell Broadcast Services (CBS).
- IP multicast, aşa cum este implementat, nu permite utilizatorilor să partajeze resursele în rețea *core* sau pe canalul radio

Unicast

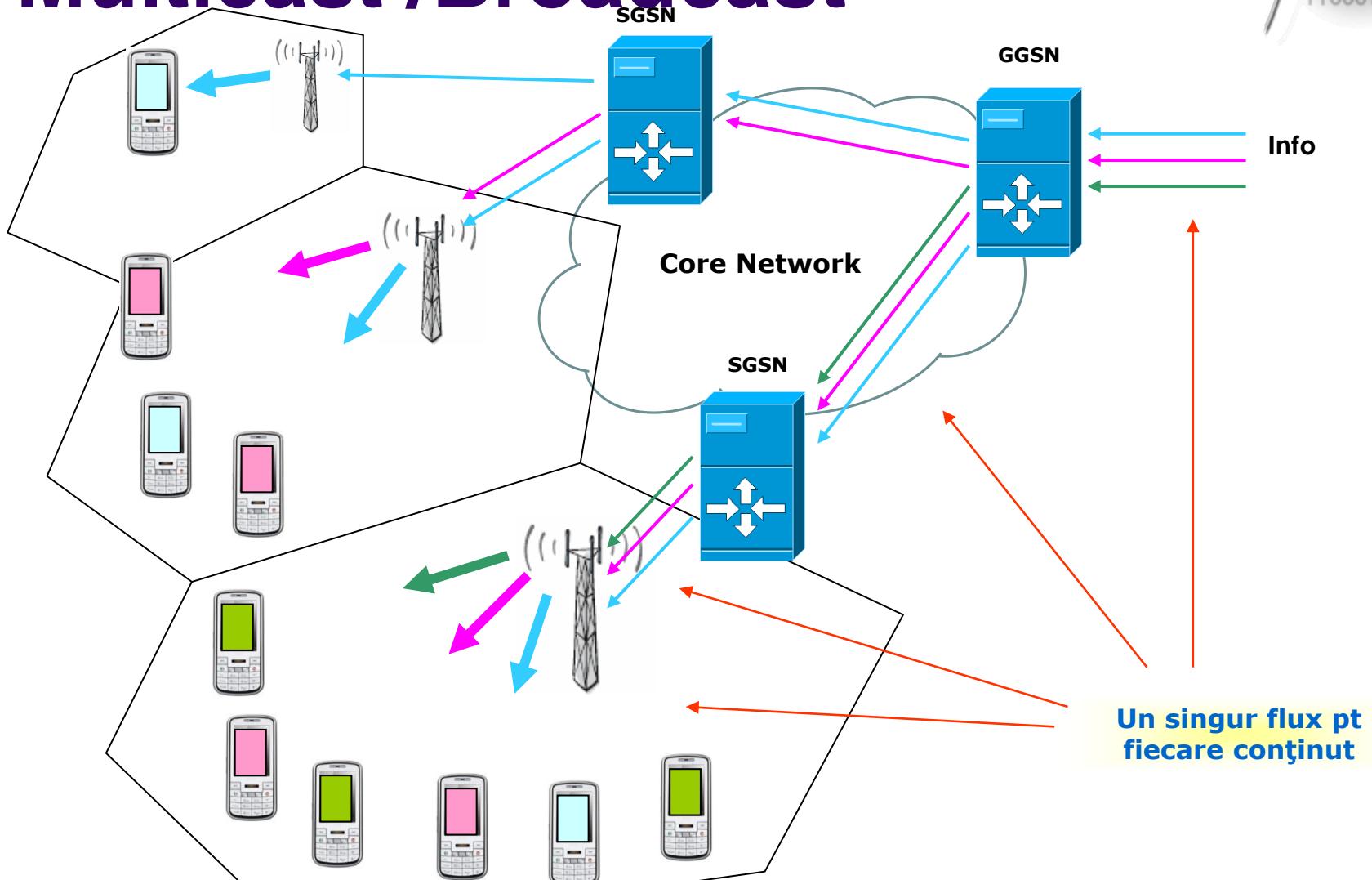
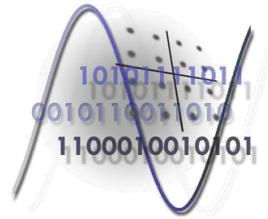


Numărul de legături
end to end egal cu
numărul utilizatorilor

Multicast /Broadcast

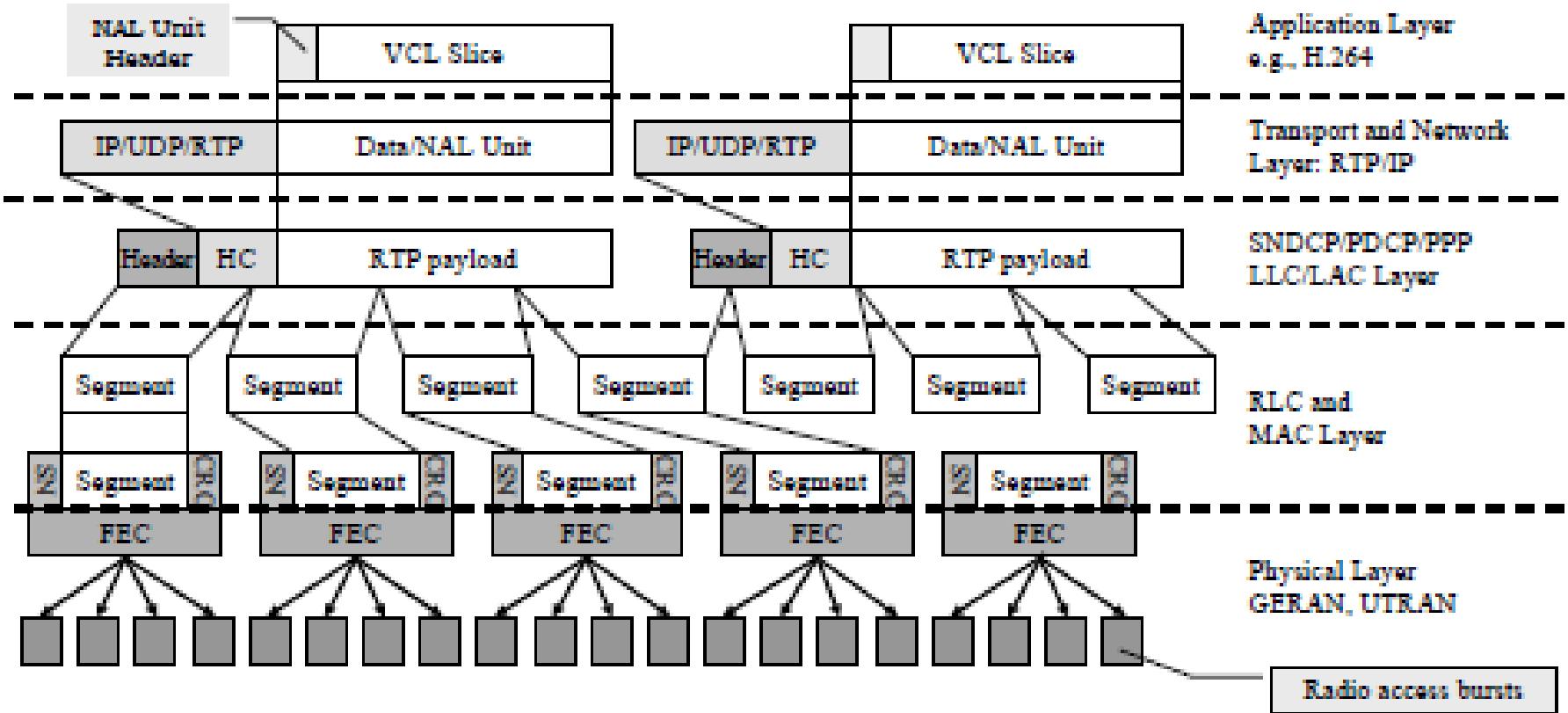
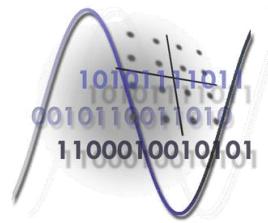


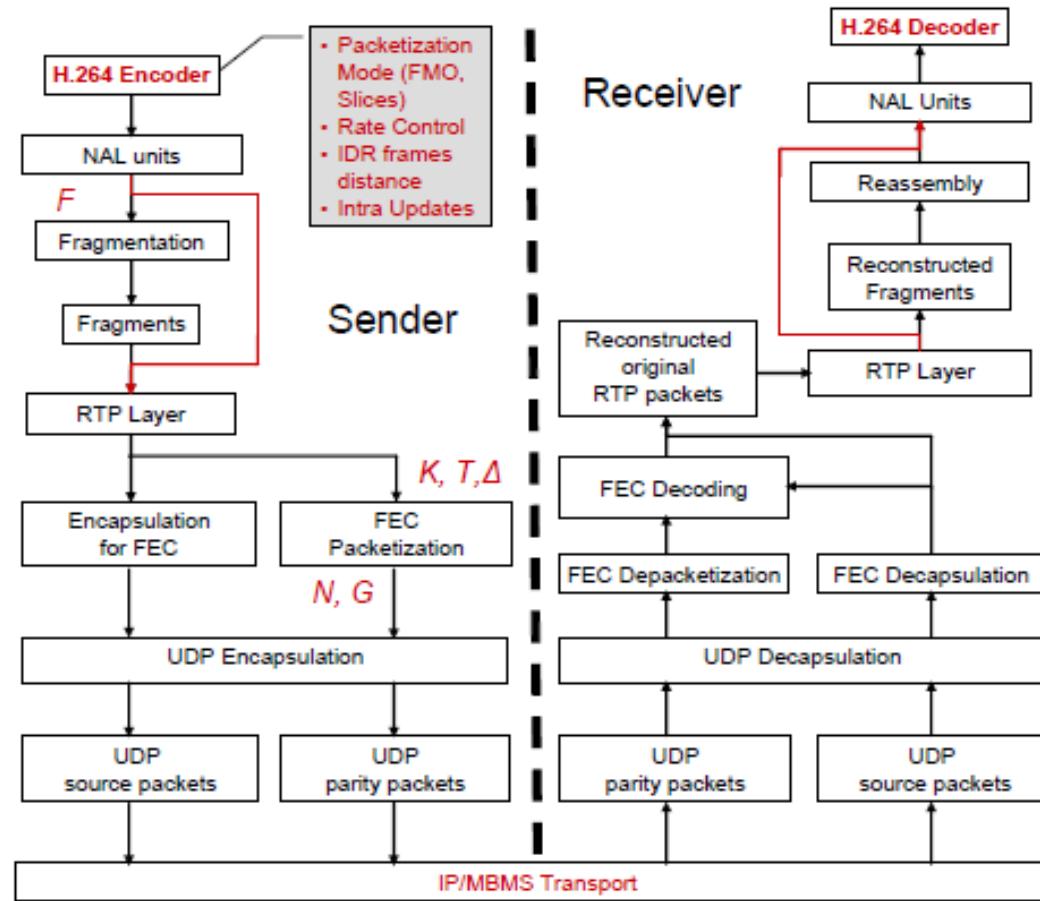
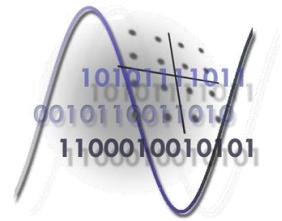
Multicast /Broadcast



Un singur flux pt fiecare conținut

Utilizarea codurilor raptor în MBMS





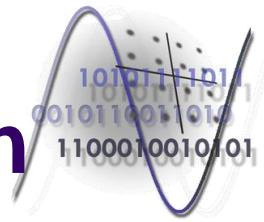
Algoritm practici de codare și decodare Raptor

Utilizarea codurilor Raptor în

MBMS

TACCFDRT Curs 5

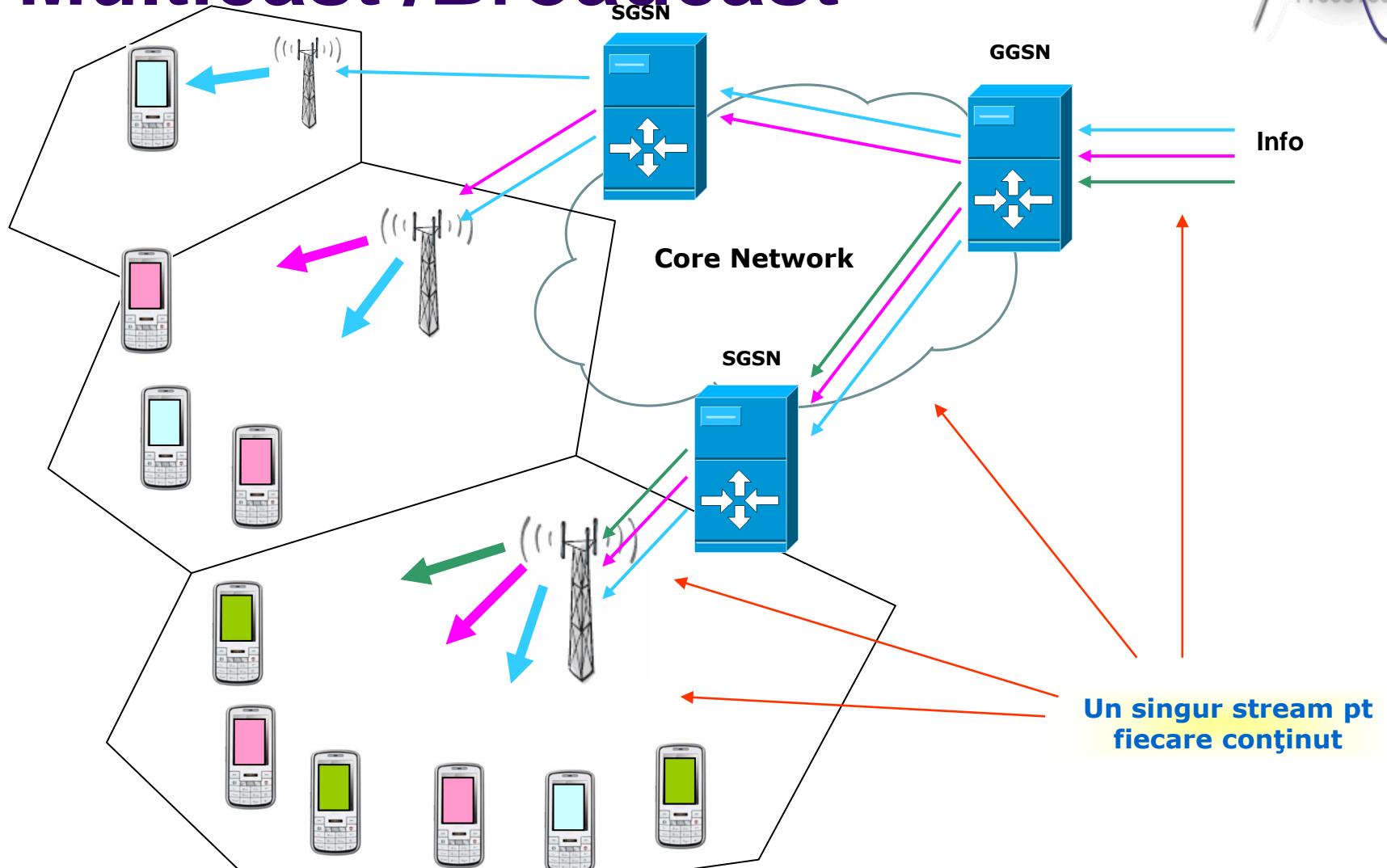
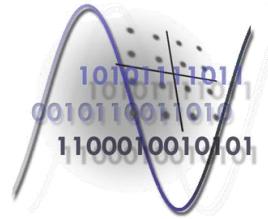




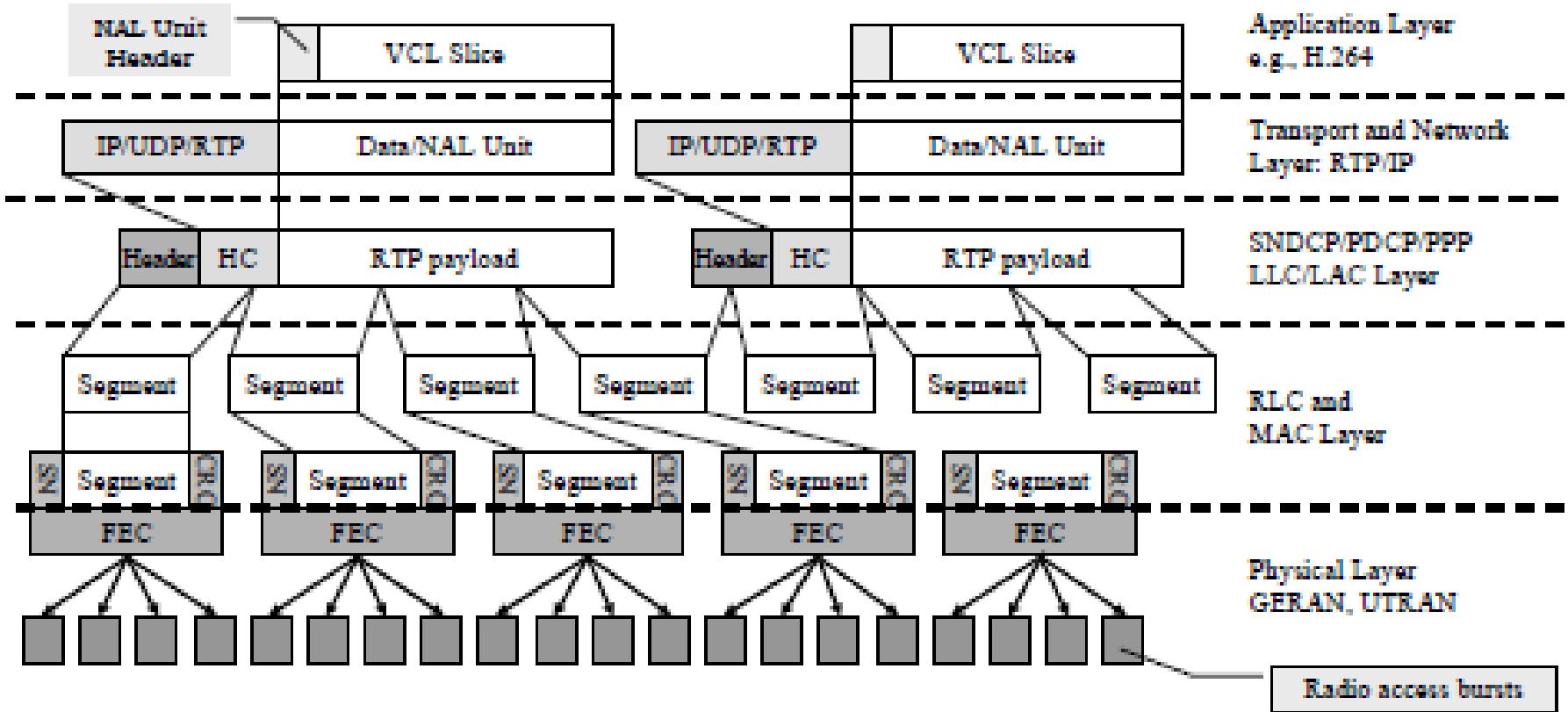
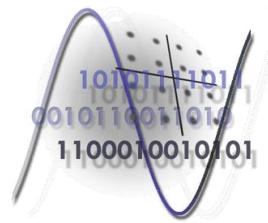
Tehnici de codare de tip Digital Fountain

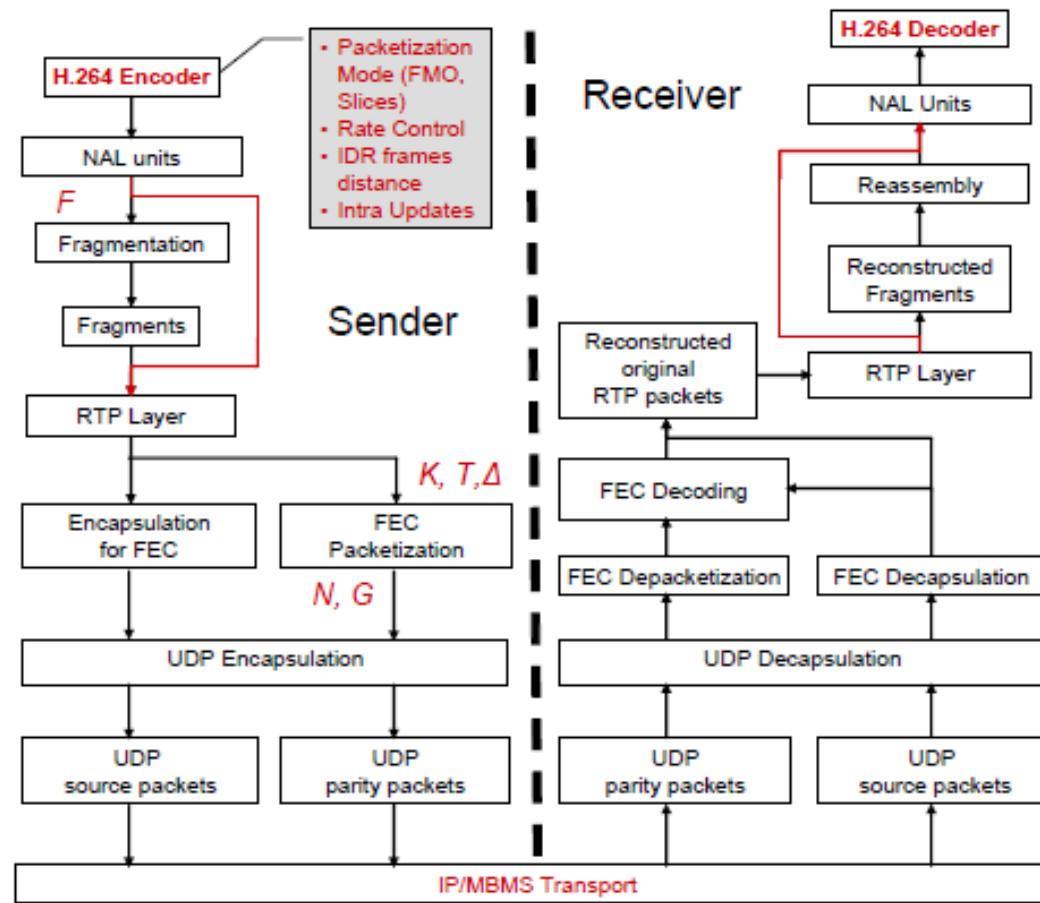
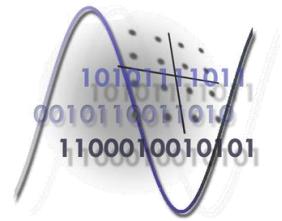
- Utilizarea codurilor raptor în MBMS
- DF în layered coding
- NC

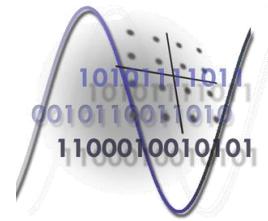
Multicast /Broadcast



Utilizarea codurilor raptor în MBMS

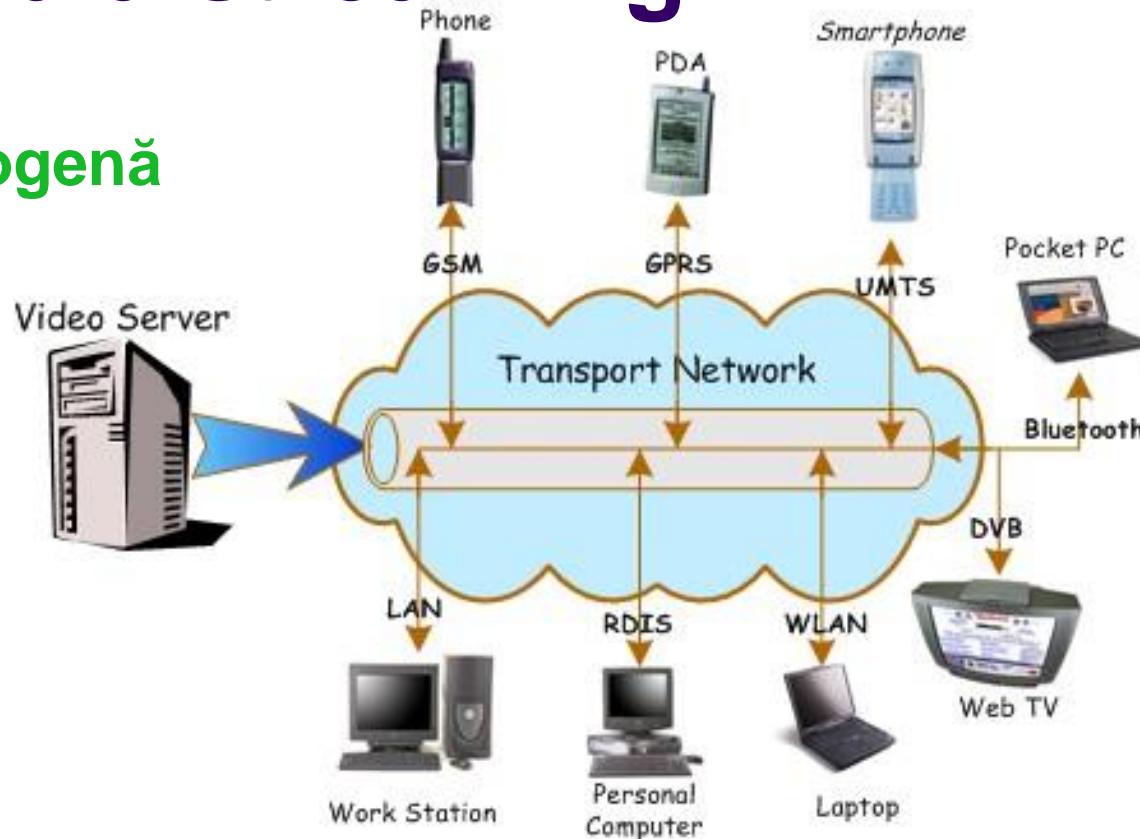






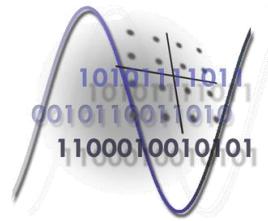
Multimedia streaming

Retea eterogenă



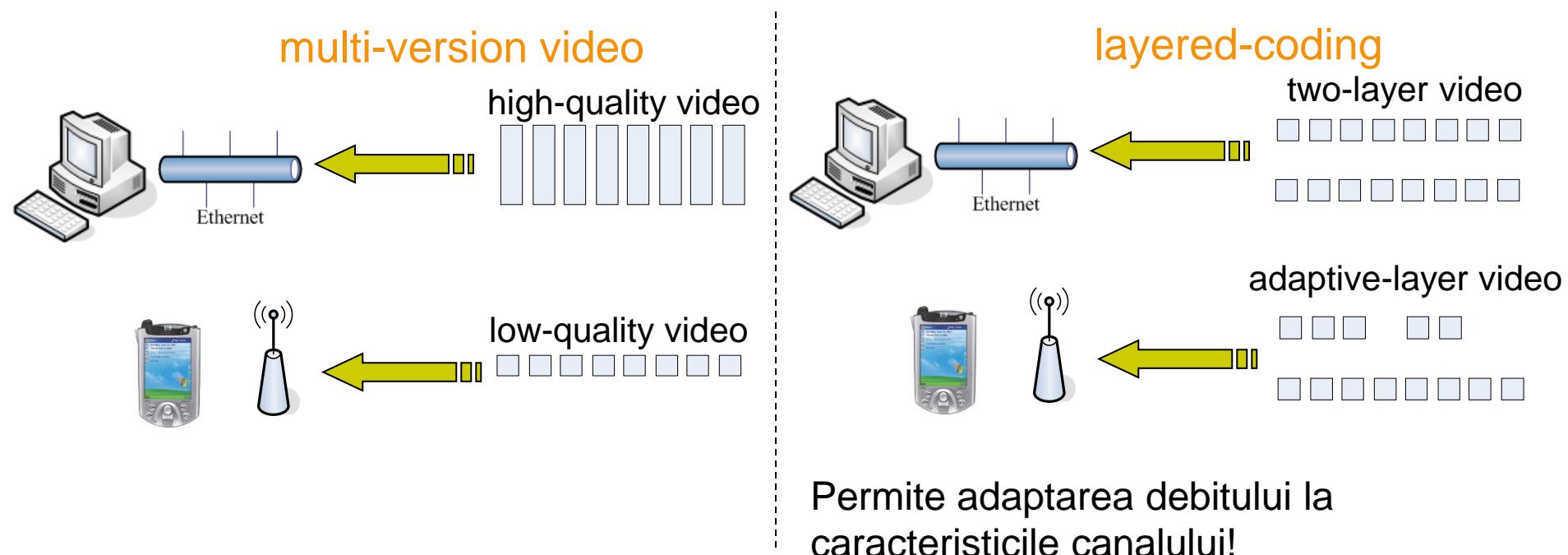
Cum se transmite un video stream?

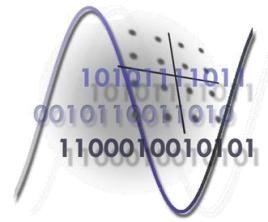
Xin Xiao, Yuanchun Shi, Yuan Gao, "Adaptive Transmission for layered streaming in heterogeneous Peer-to-Peer networks" <http://citeseerx.ist.psu.edu>



Layered (scalable) coding

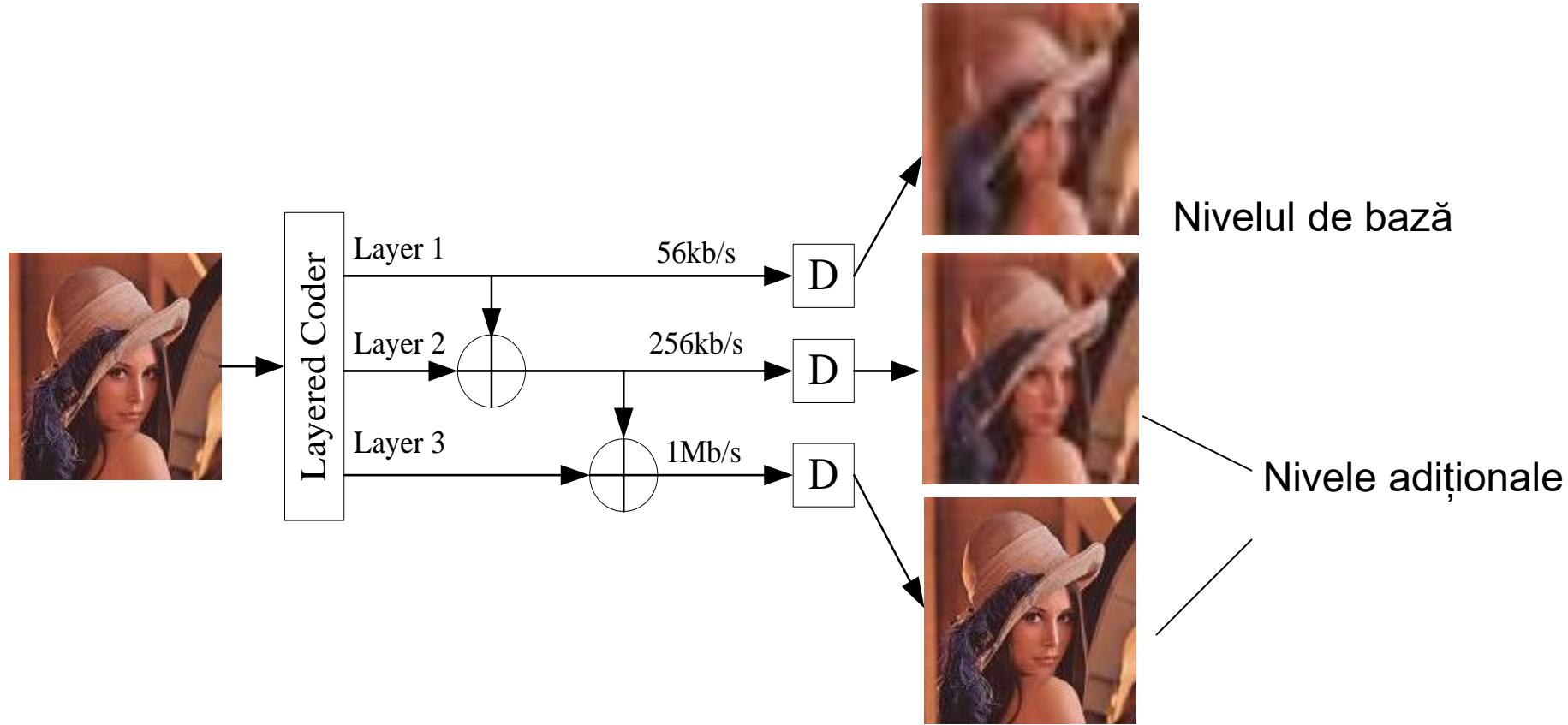
- multi-version v.s. layered coding



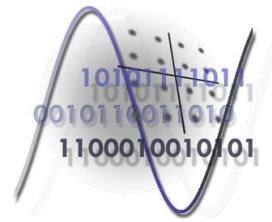


layered coding

- Mecanismul de funcționare

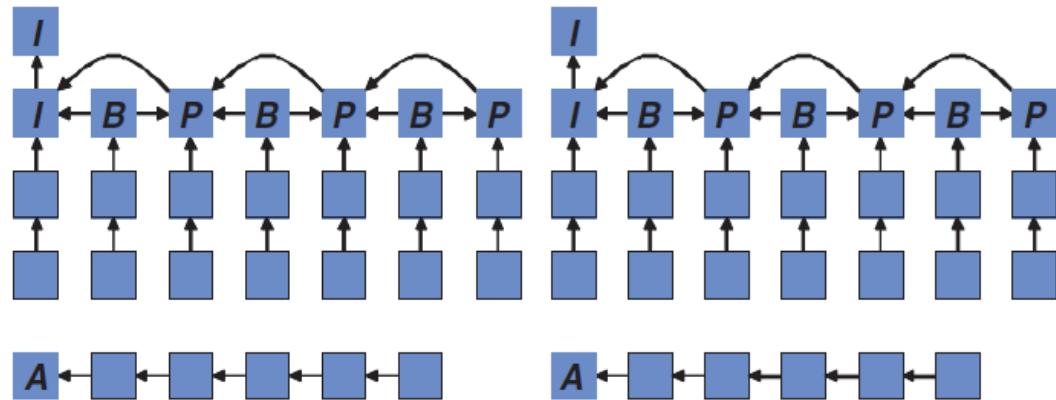


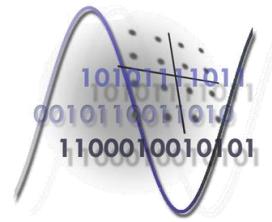
Xin Xiao, Yuanchun Shi, Yuan Gao, "Adaptive Transmission for layered streaming in heterogeneous Peer-to-Peer networks" <http://citeseerx.ist.psu.edu>



layered coding

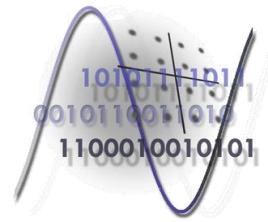
- Datele codate sunt împărtite în unități elementare
- Interdependențele între unitățile elementare pot fi reprezentate pe un graf direcționat aciclic
 - Fiecare nod al grafului reprezintă o unitate elementară
 - O legătură direcționată de la nodul I' către nodul I înseamnă că nodul I poate fi decodat numai după decodarea nodului I'





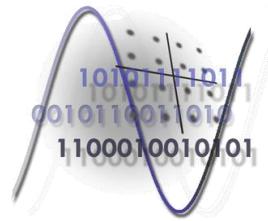
layered coding

- La fiecare unitate elementară I este caracterizat de:
 - Dimensiune B_I
 - Timpul de decodare t_D
 - Reprezintă intervalul maxim de timp în care pachetul trebuie să fie decodat
 - Factor de importanță



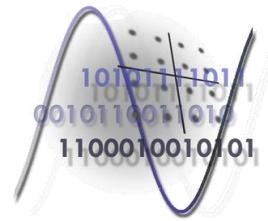
layered coding

- Se utilizează protecție neuniformă
 - Nivelul de bază trebuie codat cu un cod puternic
 - Nivelele adiționale pot fi codate cu coduri mai slabe sau să fie transmise necodate
- Protocol de transport trebuie să asigure ca pachetele din nivelul de bază să ajungă înaintea pachetelor din nivele adiționale



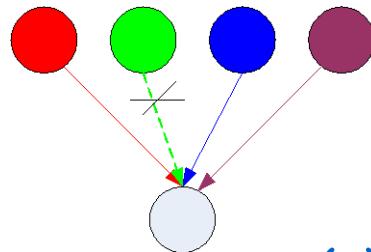
layered coding

- Dezavantajul
 - dacă pachetul de pe nivelul de bază nu a sosit în timp util pachetele recepționate aparținând nivellelor adiționale devin redundante

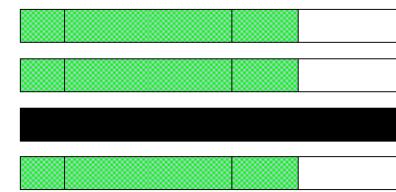
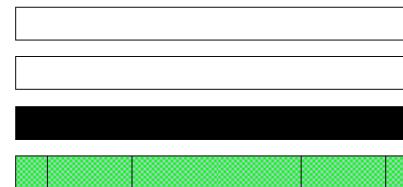


Multiple description coding

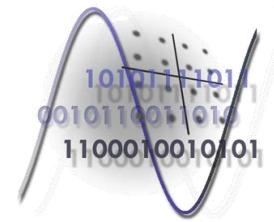
- A fost propus ca o alternativă pentru LC în video streaming
- Fiecare “descriere” conține informația necesară pentru o redare cu calitatea de bază, și fiecare “descriere” în plus îmbunătățește calitatea redării



(a) Layered coding

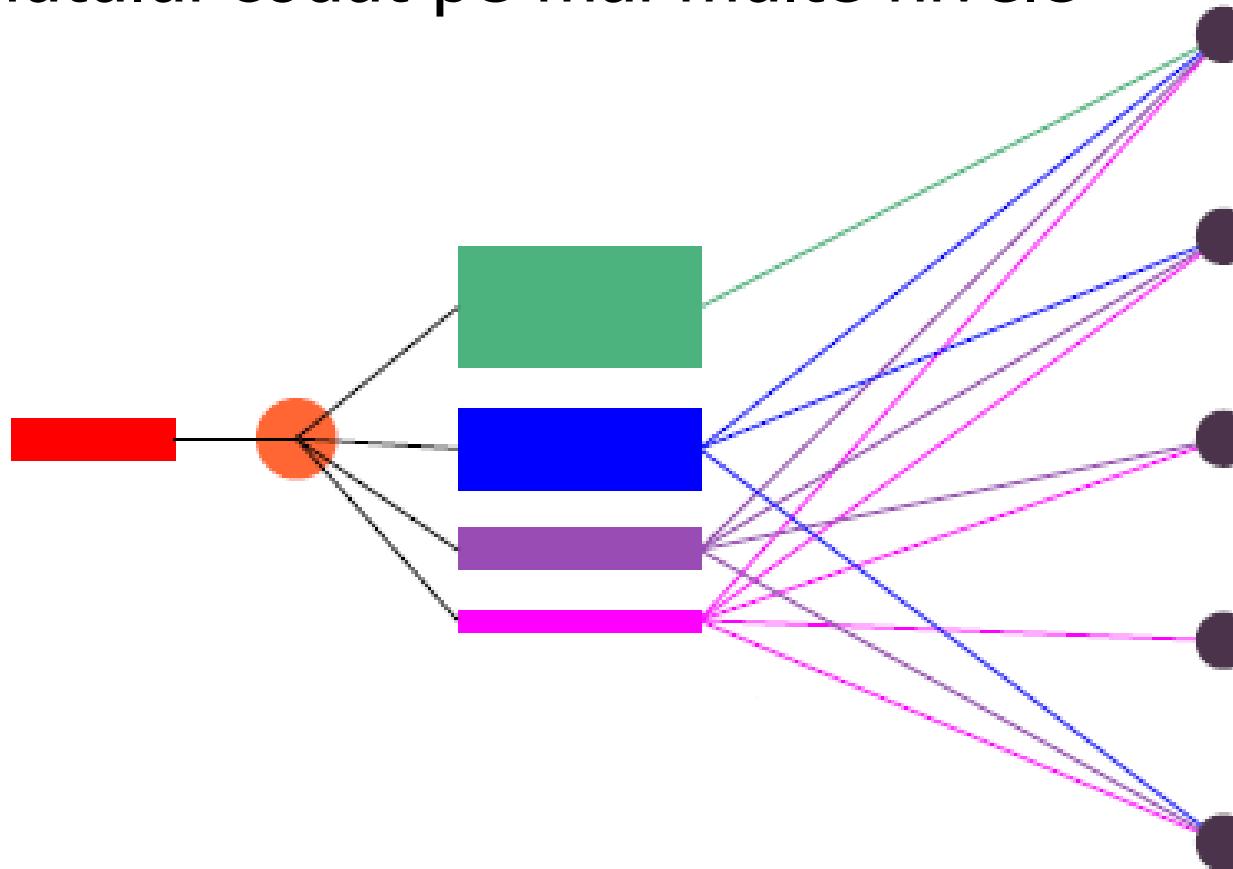


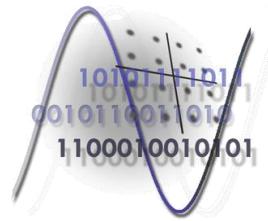
(b) MDC



layered coding & MDC

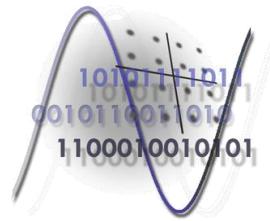
- Prin utilizarea codurilor DF se simplifica multicastul conținutului codat pe mai multe nivele



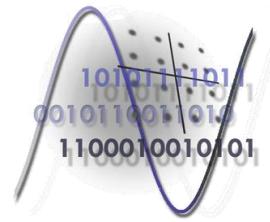


layered coding & MDC

- pentru fiecare nivel se initializează o legătură între serverul multimedia și aplicația de redare a fluxului multimedia
- Legătura pe care se transmite nivelul de bază are prioritatea cea mai mare
- Nivelele adiționale se transmit pe legături cu prioritate mai redusă

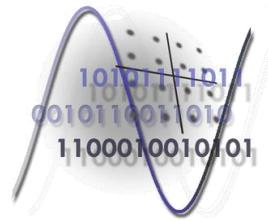


Network coding



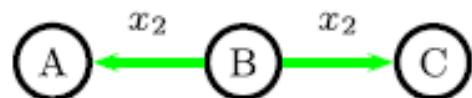
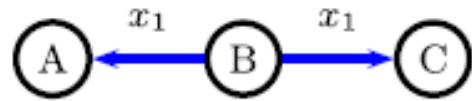
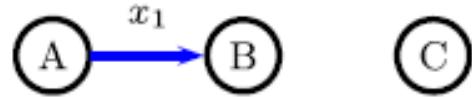
Introducere în NC

- În rețelele de calculatoare fiecare flux de date se propagă independent chiar dacă împarte același resurse cu alte fluxuri de date
- NC este o tehnică de combinare a fluxurilor de date care utilizează aceleași resurse

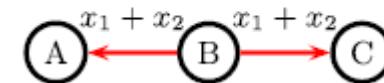
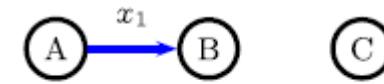


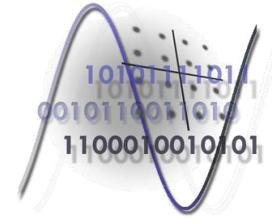
Wireless network

- Fără NC



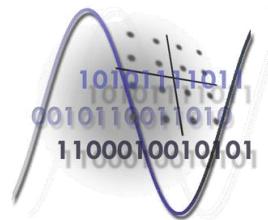
- Cu NC





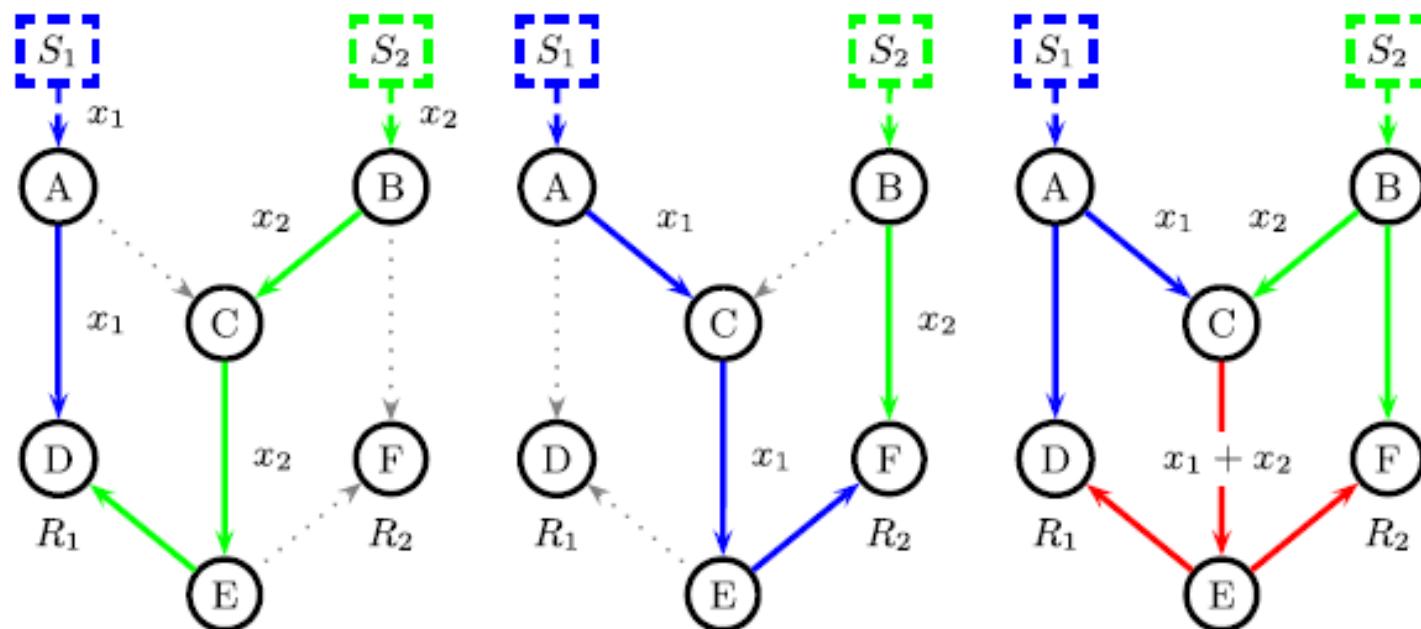
Modelare Matematică

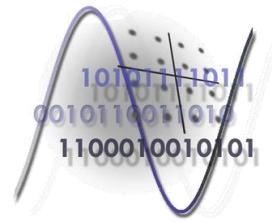
- O rețea de comunicații poate fi reprezentată printr-un graf orientat $G=(V,E)$
 - V reprezintă mulțimea nodurilor,
 - E reprezintă ramurile grafului, fiecare ramură din graf reprezintă un canal de comunicație având capacitate de o unitate de date pe o unitate de timp
 - pot exista mai multe ramuri între două noduri
- Un nod care nu are ramuri de intrare se numește sursă
- Un nod care nu are legături de ieșire se numește destinație



Exemple

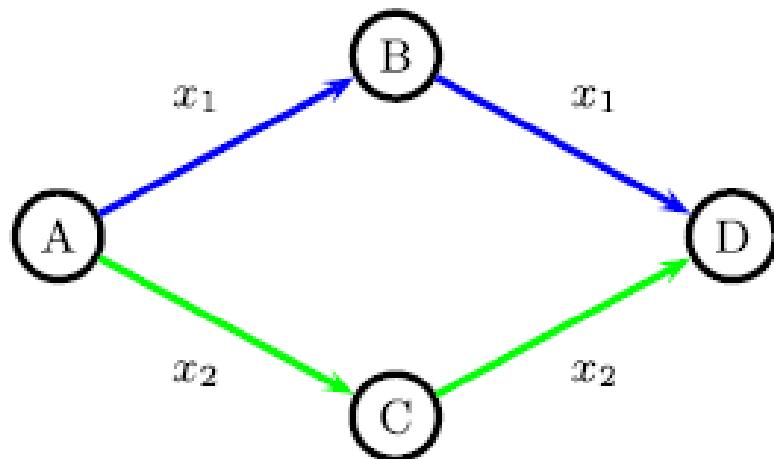
- Butterfly network



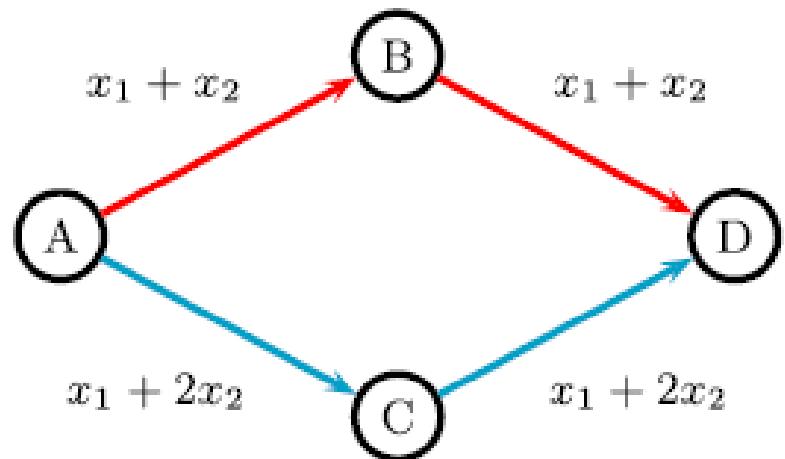


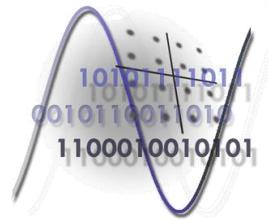
Criptare cu NC

- Fără NC



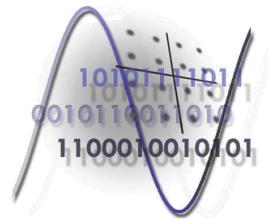
- Cu NC



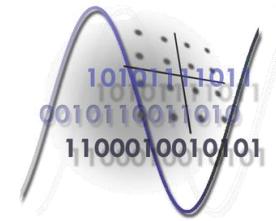


Cuprins

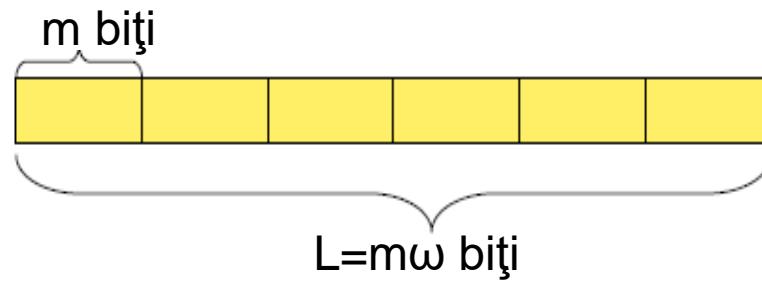
- Introducere în NC
- Descriere matematică
- Random NC
- NC în rețele cu erori
- Optimizarea rețelei



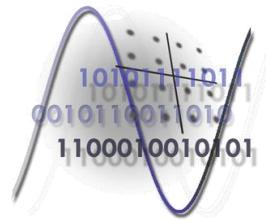
- Pentru fiecare nod T din rețea $In(T)$ este setul de canale de intrare în T , iar $Out(T)$ este setul de canale de ieșire din T
- $In(S)$ este setul de canale imaginare, care se termină la nodul sursă S , dar nu au nod de origine. Numărul acestor canale imaginare se notează cu ω
- Ramurile grafului sunt notate cu $(v_1, v_2, i) \in E$
- Începutul și sfârșitul unei muchii $e = (v_1, v_2, i)$ vor fi notate cu $v_1 = head(e)$ și $v_2 = tail(e)$.



- O unitate de date (simbol) este un element dintr-un câmp finit F
- Un mesaj constă din ω unități de date și va fi reprezentat printr-un vector $x \in F^\omega$

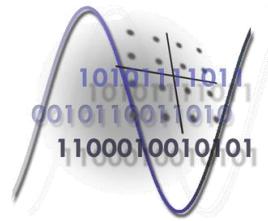


- Adică fiecare pachet poate fi privit ca un vector de ω simboluri din câmpul $GF(2^m)$



NC liniare

- Fiecare simbol din rețea aparține câmpului $GF(2^m)$
- Nodurile pot să proceseze simbolurile recepționate, efectuând combinații liniare ale acestora
 - Simbolul obținut după o combinație liniară aparține tot câmpului $GF(2^m)$
 - Lungimea pachetului după o combinație liniară nu se modifică
- Notăm cu
 - $X(v,l)$, $l=1, \dots, \omega(v)$ - simbolurile de intrare a unui nod sursă
 - $Y(e)$, simbolul transmis pe o ramură
 - $Z(v,l)$ simbolurile de ieșire a unui nod destinație

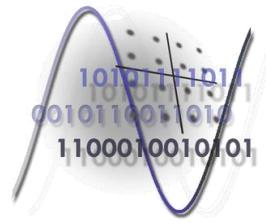


- **Definiție** Fie $G=(V,E)$ o rețea fără întârzieri. G va fi liniară peste câmpul $GF(2^m)$, dacă pentru toate legăturile, “procesele” $Y(e)$ pe o legătură $e=(v,u,i)$ satisfac

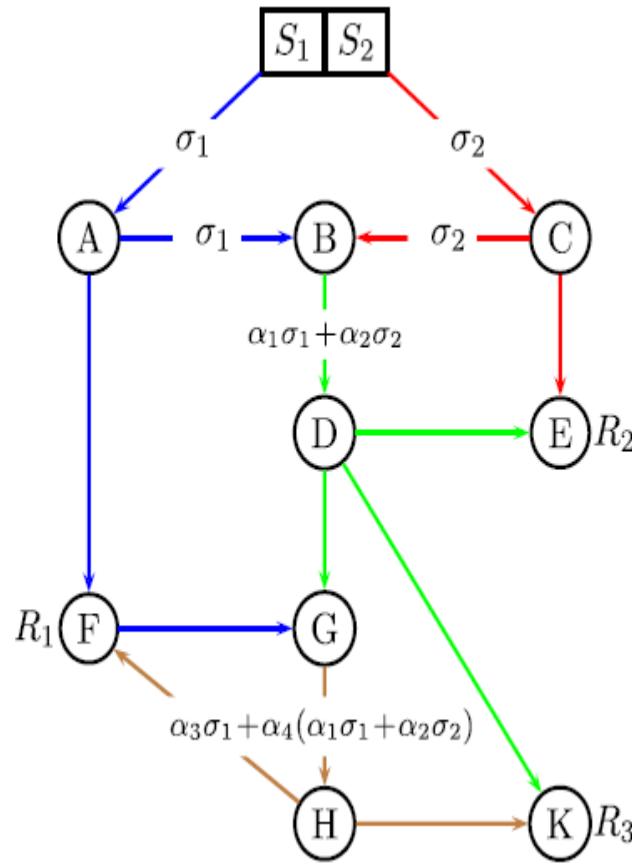
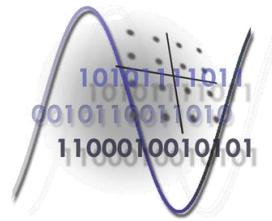
$$Y(e) = \sum_{j=1}^{\omega(v)} \alpha_{e,j} X(v, j) + \sum_{e': \text{head}(e') = \text{tail}(e)} \beta_{e',e} Y(e')$$

Unde coeficienții $\alpha_{e,j}$ și $\beta_{e,e}$ sunt elemente din $GF(2^m)$
Ieșirile $Z(v,j)$ a unui nod v vor fi

$$Z(v, j) = \sum_{e': \text{head}(e') = v} \varepsilon_{e',j} Y(e')$$

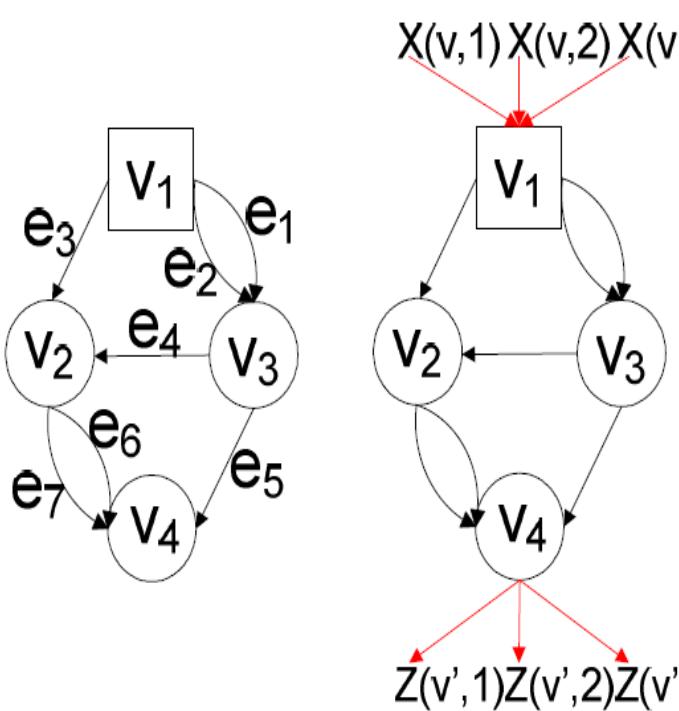


- Vectorul de codare locală pentru o ramură e este definită ca un vector de dimensiune $1 \times |In(v)|$ care conține coeficienții din câmpul $GF(2^m)$ cu care sunt înmulțiți simbolurile de pe fiecare intrare a nodului v de unde pornește ramura e
- Vectorul de codare globală pentru ramura e este un vector de lungime ω , elementele acestui vector arată cum sunt mapate în mesajul transmis pe ramura e cele ω simboluri de intrare a rețelei



Vectorul de codare locală $f(BD)$
pentru ramura BD este $[\alpha_1 \alpha_2]$

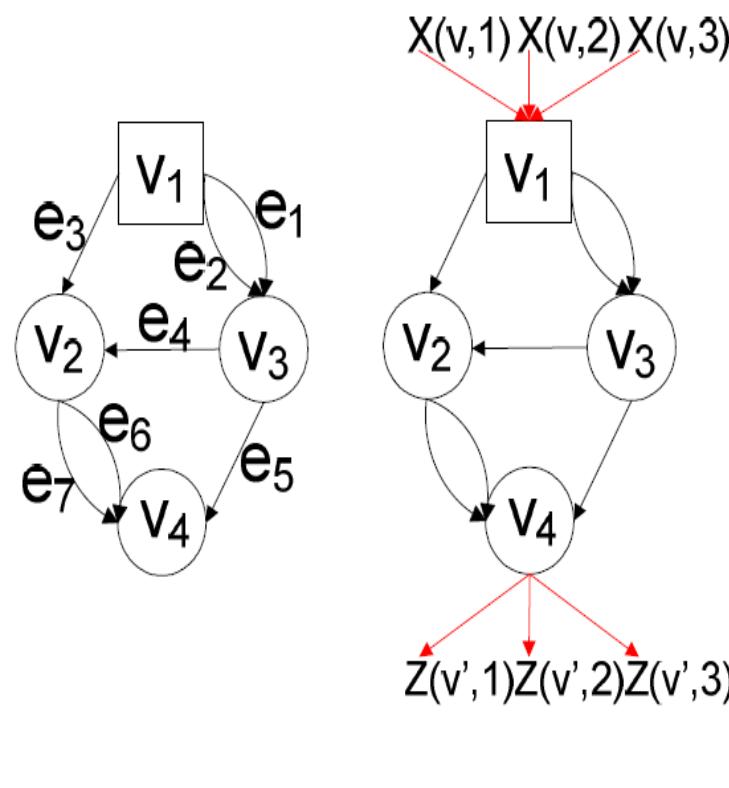
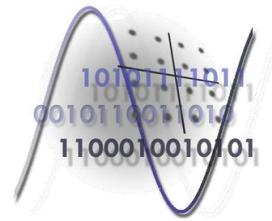
Vectorul de codare globală
pentru ramura GH este
 $[\alpha_3 + \alpha_1 \alpha_4, \alpha_2 \alpha_4]$



$$\begin{cases} Y(e_1) = \alpha_{e_1,1}X(v,1) + \alpha_{e_1,2}X(v,2) + \alpha_{e_1,3}X(v,3) \\ Y(e_2) = \alpha_{e_2,1}X(v,1) + \alpha_{e_2,2}X(v,2) + \alpha_{e_2,3}X(v,3) \\ Y(e_3) = \alpha_{e_3,1}X(v,1) + \alpha_{e_3,2}X(v,2) + \alpha_{e_3,3}X(v,3) \end{cases}$$

$$\begin{cases} Y(e_4) = \beta_{e_1,e_4}Y(e_1) + \beta_{e_2,e_4}Y(e_2) \\ Y(e_5) = \beta_{e_1,e_5}Y(e_1) + \beta_{e_2,e_5}Y(e_2) \\ Y(e_6) = \beta_{e_3,e_6}Y(e_3) + \beta_{e_4,e_6}Y(e_4) \\ Y(e_7) = \beta_{e_3,e_7}Y(e_3) + \beta_{e_4,e_7}Y(e_4) \end{cases}$$

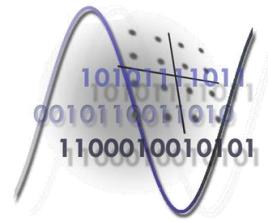
$$\begin{cases} Z(v',1) = \varepsilon_{e_5,1}Y(e_5) + \varepsilon_{e_6,1}Y(e_6) + \varepsilon_{e_7,1}Y(e_7) \\ Z(v',2) = \varepsilon_{e_5,2}Y(e_5) + \varepsilon_{e_6,2}Y(e_6) + \varepsilon_{e_7,2}Y(e_7) \\ Z(v',3) = \varepsilon_{e_5,3}Y(e_5) + \varepsilon_{e_6,3}Y(e_6) + \varepsilon_{e_7,3}Y(e_7) \end{cases}$$



$$A = \begin{pmatrix} \alpha_{e_1,1} & \alpha_{e_2,1} & \alpha_{e_3,1} \\ \alpha_{e_1,2} & \alpha_{e_2,2} & \alpha_{e_3,2} \\ \alpha_{e_1,3} & \alpha_{e_2,3} & \alpha_{e_3,3} \end{pmatrix} B = \begin{pmatrix} \varepsilon_{e_5,1} & \varepsilon_{e_5,2} & \varepsilon_{e_5,3} \\ \varepsilon_{e_6,1} & \varepsilon_{e_6,2} & \varepsilon_{e_6,3} \\ \varepsilon_{e_7,1} & \varepsilon_{e_7,2} & \varepsilon_{e_7,3} \end{pmatrix}$$

Matricea de transfer M a sistemului este egală cu:

$$M = A \begin{pmatrix} \beta_{e_1,e_5} & \beta_{e_1,e_4}\beta_{e_4,e_6} & \beta_{e_1,e_4}\beta_{e_4,e_7} \\ \beta_{e_2,e_5} & \beta_{e_2,e_4}\beta_{e_4,e_6} & \beta_{e_2,e_4}\beta_{e_4,e_7} \\ 0 & \beta_{e_3,e_6} & \beta_{e_3,e_7} \end{pmatrix} B^T$$



- Se definește matricea de adiacență F cu elementele $F_{i,j}$ date prin

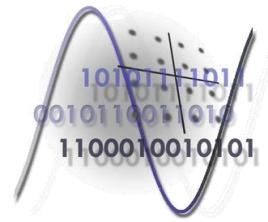
$$F_{i,j} = \begin{cases} \beta_{e_i, e_j} & \text{head}(e_i) = \text{tail}(e_j) \\ 0 & \text{in rest} \end{cases}$$

- Elementele matricii A de dimensiune $\omega \times |E|$ definește ca

$$A_{i,j} = \begin{cases} \alpha_{e_j, l} & x_i = X(\text{tail}(e_j), l) \\ 0 & \text{in rest} \end{cases}$$

- Elementele matricii B de dimensiune $\omega \times |E|$ definește ca

$$B_{i,j} = \begin{cases} \varepsilon_{e_j, l} & z_i = Z(\text{head}(e_j), l) \\ 0 & \text{in rest} \end{cases}$$



- Matricea de transfer a rețelei este dată de

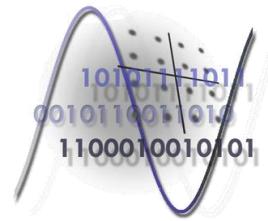
$$M = A(I - F)^{-1}B^T$$

Coeficientii se aleg astfel încât toate matricele de transfer între sursă și oricare nod terminal să pot fi inversate

Putem scrie ca

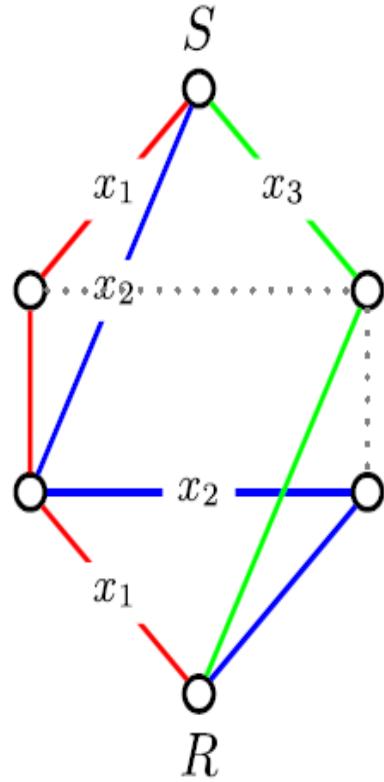
$$z = Mx^T$$

Adică NC transformă “rețeaua” într-un “canal”

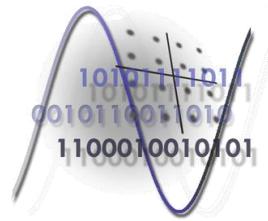


Teorema min cut max flow

- *Min cut* numărul de ramuri minime care trebuie eliminate dintr-un graf pentru ca un nod destinație să nu fie legată la un nod sursă

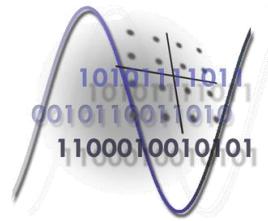


- Rata maximă cu care se poate transmite date de la sursa S la destinația R este egală cu *min cut*, adică se pot găsi exact *min cut* căi direcționate între S și R



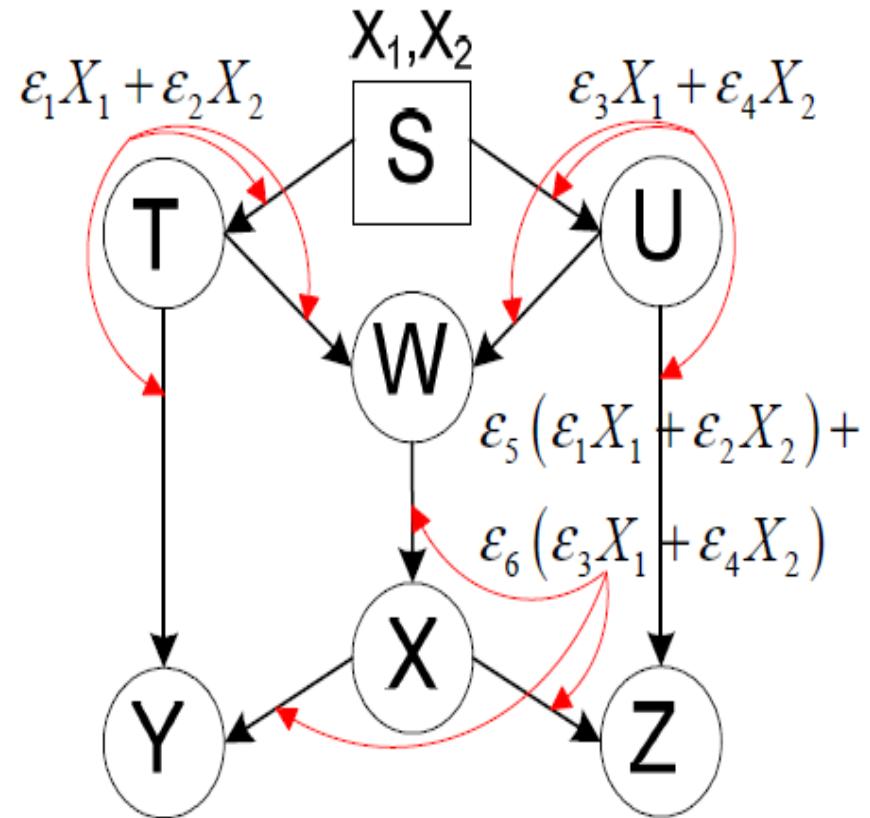
Teorema min cut max flow

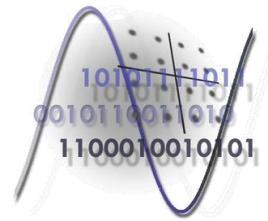
Prin utilizarea NC într-o rețea, o sursă poate să transmită simultan la mai multe destinații cu debitul maxim egal cu *min cut*-ul minim dintre destinații și surse.



Random NC

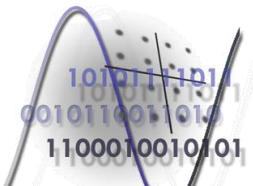
- În acest caz fiecare nod va efectua o mapare liniară aleatoare a simbolurilor de intrare în simbolurile de ieșire
- Nodurile terminale trebuie să știe numai vectorul de codare globală a ramurilor de intrare



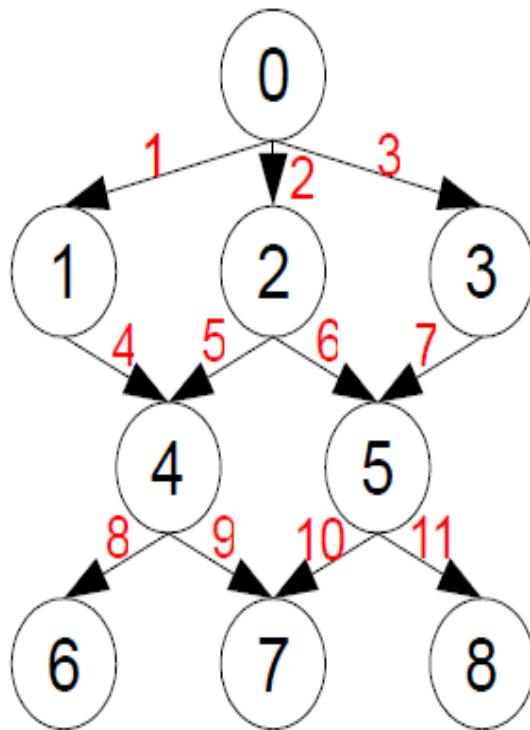


RNC cu întârzieri

- Dacă rețeaua este aciclică operațiunea de decodare poate fi sincronizată, astfel întârzierile nu au efect asupra decodării
- Dacă rețeaua este ciclică, decodare nu tolerăza întârzierile



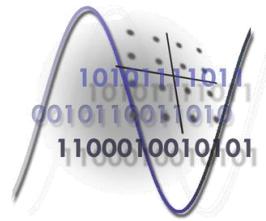
NC în rețelele cu erori



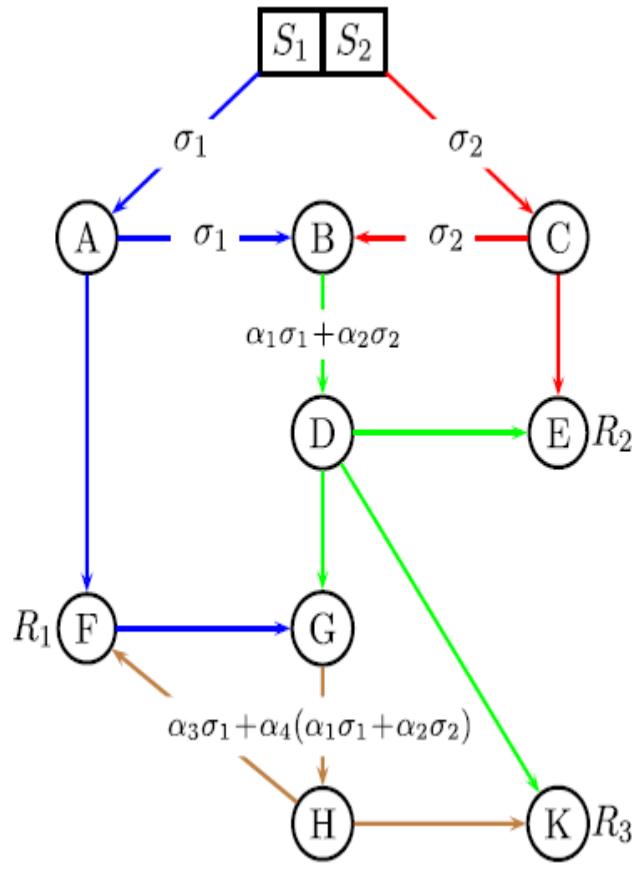
$$A = \begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & 0 & 0 & \textcolor{red}{0} & 0 & 0 & 0 & 0 & 0 \\ a_{1,0} & a_{1,1} & a_{1,2} & 0 & 0 & \textcolor{red}{0} & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$F = \begin{pmatrix} 0 & 0 & 0 & f_{0,3} & 0 & \textcolor{red}{0} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & f_{1,4} & \textcolor{red}{f}_{1,5} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcolor{red}{0} & f_{2,6} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcolor{red}{0} & 0 & f_{3,7} & f_{3,8} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcolor{red}{0} & 0 & 0 & 0 & f_{4,9} & f_{4,10} \\ \textcolor{red}{0} & \textcolor{red}{0} \\ 0 & 0 & 0 & 0 & 0 & \textcolor{red}{0} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcolor{red}{0} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcolor{red}{0} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcolor{red}{0} & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$B_0 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & \textcolor{red}{0} & 0 & b_{0,0,7} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcolor{red}{0} & 0 & b_{0,1,7} & 0 & 0 & 0 \end{pmatrix}$$



Optimizarea arhitecturii rețelei

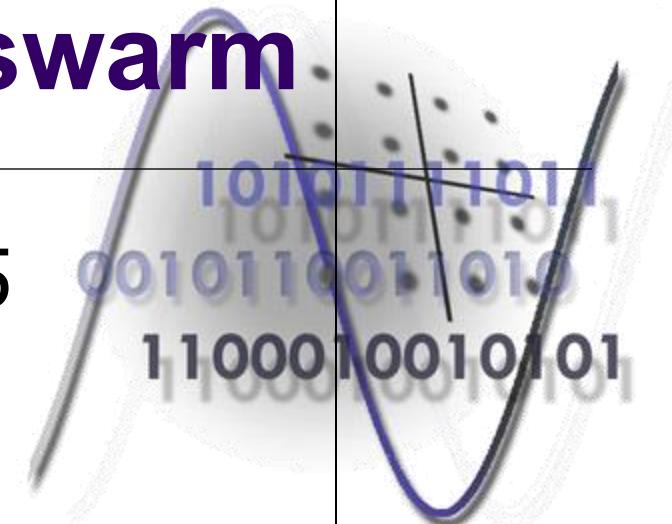


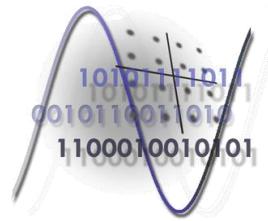
- Un algoritm genetic centralizat are următorii pași

```
[C1] initializare populatie;
[C2] evaluare populatie;
[C3] WHILE se ajunge la criteriul de terminare
{
    [C4] selectare solutii pentru populatia urmatoare;
    [C5] crossover;
    [C6] mutatie;
    [C7] evaluarea populatiei;
}
```

Tehnici de codare network coding utilizate în sisteme de comunicații de tip swarm

TACCFDRT Curs 5

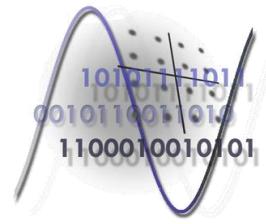




Cuprins

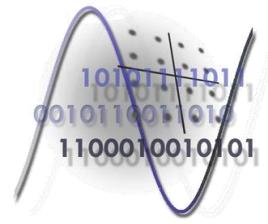
- Metode propuse pentru „content distribution”
 - Distribuție cooperativă a datelor
 - NC in swarm
- Tehnici de codare de tip “Xor in the Air”
 - MIXIT
 - COPE

Metode propuse pentru „content distribution”



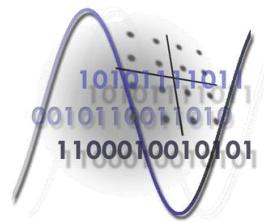
- *Sisteme cooperative bazate pe arbori*
 - crearea și menținerea unui arbore de multicast ce selectează căile cele mai scurte
 - În rețelele complexe este dificil construirea și optimizarea rețelei
- *Arhitecturi cooperative de tip Mesh*
 - Cresc eficiența transmisiei prin exploatarea căilor paralele
 - Existenta rutelor paralele și deciziile locale conduc la recepționarea multiplă a unor pachete (scade eficiența)
 - Cea mai cunoscută arhitectură de acest tip este BitTorrent

Metode propuse pentru „content distribution”



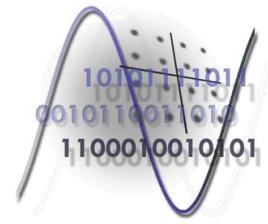
- *Coduri corectoare de stergeri – DF*
 - Nodul terminal (destinatar) reconstruiește informația originală dintr-un număr suficient de mare de pachete receptionate
 - Deoarece raportul pachetelor dublate la un nod terminal într-o rețea de tip mesh este relativ mare trebuie găsite metode de cooperare eficiente între noduri
- *Network Coding*
 - Utilizarea optima a NC presupune o cunoaștere detailată a topologiei și posibilitatea calculării centralizate a schemei de distribuție

Model de distribuție cooperativă a datelor



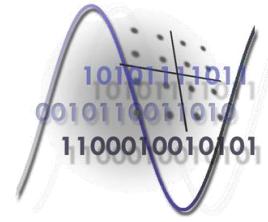
- Se consideră o rețea formată dintr-un grup de utilizatori interesați în descărcarea unui fișier încărcat pe un singur server.
- Capacitatea serverului este limitată.
- Fiecare client poate contribui cu resursele lui pentru a ajuta pe alți utilizatori.
- Serverul divide fișierul în k blocuri și trimite blocurile în mod aleator pentru clienți diferiți.
- Utilizatorii nu știu identitățile tuturor celorlalți utilizatori ci numai a celor situați într-o anumită "vecinătate".
- Fiecare nod poate comunica numai cu vecinii, numărul acestora fiind relative mic.
- Alegerea vecinătății se poate realiza cu ajutorul unui server centralizat sau în mod distribuit.
- Vecinătățile se pot reorganiza datorită plecării nodurilor sau datorită necesității de a crește debitul

Model pentru distribuția datelor necodate sau cu codarea sursei

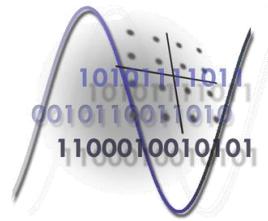


- La fiecare transfer a unui bloc are loc un proces de decizie legat de selectarea blocului care se va transmite
 - Se presupune că nici serverul și nici utilizatorii nu au informații complete despre blocurile deținute de nodurile din sistem
- Reguli utilizate pentru selecția blocurilor sunt:
 - *Random block*. Blocul care se transferă se alege aleator dintre blocurile care există la sursă. Dacă sursa este serverul atunci blocul aleator se alege dintre toate blocurile disponibile.
 - *Local Rarest*. Blocul care se transferă este ales dintre blocurile cele mai rare care se găsesc în vecinătate. Dacă există mai multe astfel de blocuri se alege unul aleator.
 - *Global Rarest*. Este schema de bază care nu este adecvată pentru rețele extinse. Blocul care se transferă este blocul cel mai rar la nivel de sistem din vecinătate. Această metodă dă prioritate blocurilor foarte rare pentru creșterea performanțelor

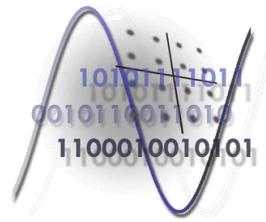
Model pentru distribuția datelor necodate sau cu codarea sursei



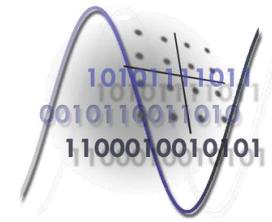
- Sistemul BitTorrent utilizează o combinație a selectiei de tip *Random* și *Local Rarest*. La început pornește cu metoda *Random* și după câteva blocuri comută în modul *Local Rarest*.
- Când codarea serverului (sursei) este utilizată atunci sistemul lucrează foarte asemănător cu sistemul descris, dar serverul generează blocuri codate și nu blocurile originale.
 - Serverul generează un bloc codat de fiecare dată când se cere un bloc nou.
 - Fiecare utilizator trebuie să descarce un număr de k blocuri distincte direct de la server sau de la alți utilizatori.
- Solutii posibile sunt coduri cu rată finită ca și codurile Reed-Solomon, sau coduri rateless care generează un număr de pachete nelimitat, dar necesită mai mult de k pachete pentru reconstrucția datelor.



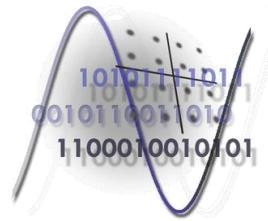
- la momentul zero serverul (nodul sursă) este singurul nod care deține o copie a întregului fișier
 - timpul de descărcare al nodului cel mai rapid este limitat inferior de timpul necesar de a transfera o copie a fiecărui bloc în rețea
- fiecare nod ar trebui să știe ce blocuri au descărcat celelalte noduri(pt. ca un pachet să nu fie cerut de mai multe ori).
 - nu se poate realiza pe scară largă, un nod fiind capabil să cunoască cel mult pachetele (blocurile) deținute de noduri localizate în vecinătatea lui.
- Pe măsură ce rețeaua crește informația pe care un nod o are asupra vecinilor va scădea, crescând probabilitatea de a se cere blocuri duplicate



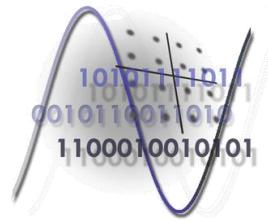
- Singurul lucru pe care un nod îl poate face este de a presupune că nodurile vecine au seturi sau subseturi similare și să ceară în consecință blocuri care nu se suprapun.
- Acest lucru este foarte dificil de realizat datorită următoarelor motive:
 - noduri noi care intră în swarm,
 - capacitateți de transmisie eterogene,
 - plecarea unor noduri la moment aleatoare (churn), pot crea swarm-uri în care noduri diferite nu dețin seturi diferite de blocuri
- Acest fapt complică estimarea de către un nod, numai pe baza unor observații locale, a blocurilor care există în alte părți ale rețelei.
- De exemplu în cazuri extreme când nodurile NU dețin seturi diferite de blocuri și vizibilitatea în rețea este foarte mică, numărul de blocuri care trebuie generate de server, pentru ca un număr de N blocuri diferite să fie plasate în swarm tinde către $N \log(N)$, ceea ce pentru $N=10000$ blocuri este egal cu 80000.



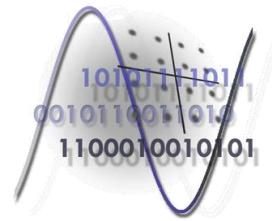
- Problemele menționate pot fi rezolvate prin utilizarea NC,
- toate pachetele generate de server fiind inovative – depinde de numărul de pachete și coeficienții de codare



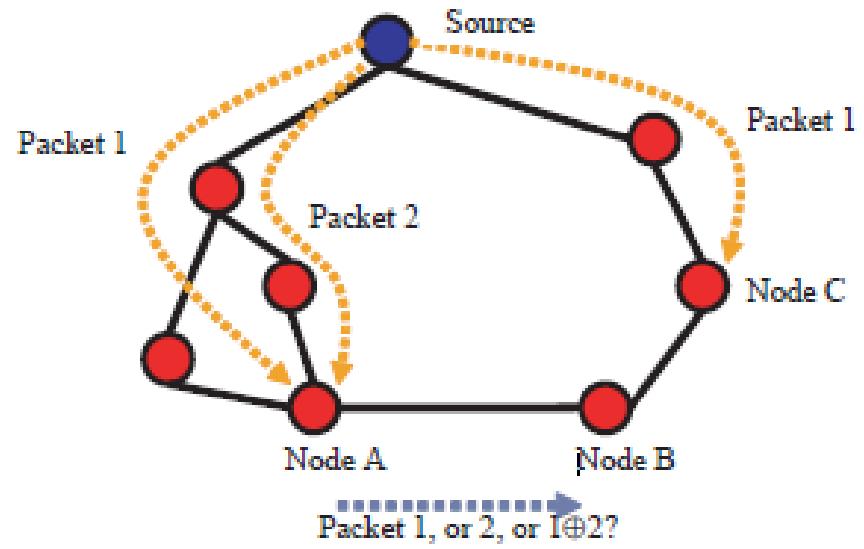
- Dacă se presupune că toate nodurile au aceeași capacitate atunci viteza de propagare a informației în rețeaua P2P este determinată de politica (regula) de selecție a blocurilor și de distanța dintre noduri.
- În cazul în care un anumit nod (sursa) deține un pachet particular, care este cerut într-o altă parte a rețelei (la destinație), nodurile intermediare combină mai multe pachete (blocuri) pe care destinația le are deja
- network coding va asigura că un astfel de bloc se va propaga într-un număr de etape egal cu distanța dintre sursă și destinație(exprimat în numărul de ramuri parcuse).



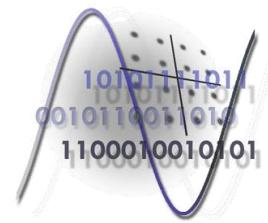
- Comparat cu o abordare tradițională, network coding utilizează optimal resursele din rețea
 - fără a fi necesare proceduri sofisticate de scheduling
 - asigură un grad ridicat de robustețe, chiar dacă pleacă noduri din sistem sau dacă decizii se iau pe baza unor informații parțiale



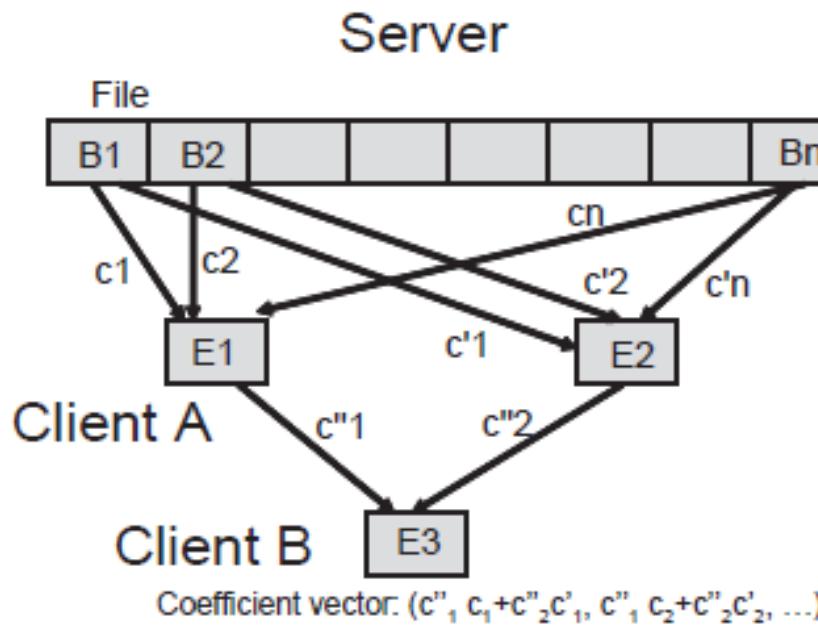
- Dacă nodul B a descărcat pachetul 1 atunci legătura dintre nodurile B și C nu poate fi utilizată pentru propagarea unor informații noi
- Dacă se utilizează network coding atunci nodul B va descărca da la A o combinație liniară a pachetelor 1 și 2, decodarea acestor pachete realizând-se cu ajutorul pachetului 1 de la nodul C

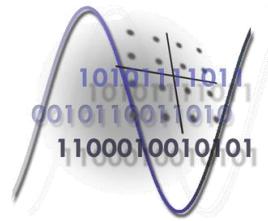


Distribuția datelor cu utilizarea network coding

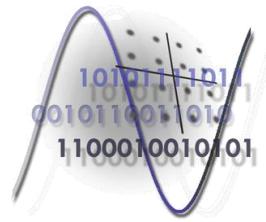


- În cazul utilizării network coding, atât serverul cât și utilizatorii realizează operații de codare.
- Oricând nodul sau serverul trebuie să trimită un bloc la un alt nod realizează o combinație liniară a tuturor (sau a unui subset) blocurilor pe care le detine



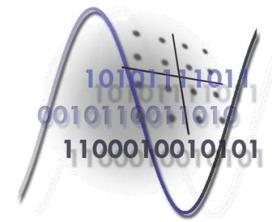


- un nod poate recupera fișierul original după recepționarea a k blocuri pentru care coeficienții asociati sunt liniar independenti
- Beneficiile unei astfel de abordări sunt datorate procesului de aleatorizare introdus în fiecare moment în care se generează un nou bloc codat
- Singura posibilitate ca un bloc codat să nu fie util este ca același bloc să fie generat independent într-un alt nod, sau de același nod într-un alt moment al transmisiei.
- Pe măsură ce blocurile sunt transmise în rețea se combină cu alte blocuri scăzând astfel probabilitatea de a se genera blocuri liniar dependente



Incentive Mechanisms

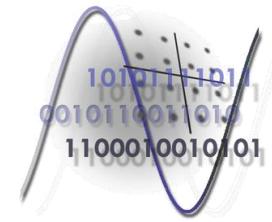
- O problemă importantă este legată de fenomenul de free-riding, adică mulți utilizatori utilizează resursele oferite rețelei de către alți utilizatori fără a contribui cu resursele lor proprii.
- Acest fenomen poate genera o scădere semnificativă a performanțelor sistemului, fiind necesare mecanisme pentru evitarea acestui fenomen. Două astfel de mecanisme sunt următoarele:
 - se dă prioritate schimbului de informații în locul transmiterii libere a datelor către alte noduri.
 - când se cere capacitate de transfer de la un nod (adică upload) utilizatorul va transmite în mod preferențial blocuri la utilizatorii de la care descarcă informație.
 - Al doilea mecanism este inspirat din mecanismul tit-for-tat utilizat de către sistemul BitTorrent.
 - Un user nu încarcă date la alt utilizator până nu a recepționat o anumită cantitate de date de la acel utilizator
- Un astfel de mecanism face operația de scheduling și mai complicată într-o rețea extinsă.
- Prin utilizarea network coding aproape fiecare bloc este unic crescând astfel semnificativ şansele ca să fie util pentru alți utilizatori și să poată fi transmis ușor la acești utilizatori.



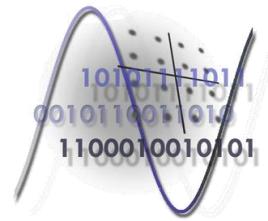
Concluzii. Elemente necesare

- decodarea necesită multiplicări de matrici – proces destul de lent.
- Trebuie stocate mai multe blocuri recepționate pentru generarea unui nou pachet codat – pentru a se asigura o probabilitate scăzută de apariție a unor pachete liniar dependente – apar întârzieri suplimentare.
- Concluzie: “în legătură cu performanțele network coding trebuie manifestat mai puțin optimism”

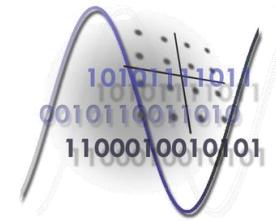
Tehnici de codare de tip “Xor in the Air”



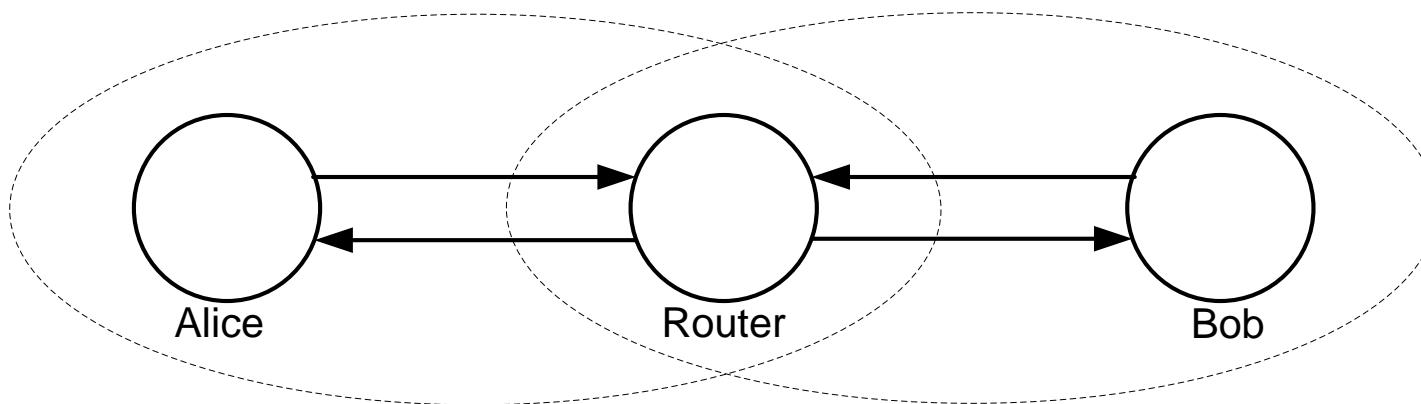
- Rețelele wireless prezintă o serie de probleme:
 - throughput redus
 - intermitente și de calitate redusă
 - puncte situate în afara accesului radio („dead spots”)
 - suport inadecvat pentru mobilitate
- natura broadcast a mediului radio
- diversitatea spațială



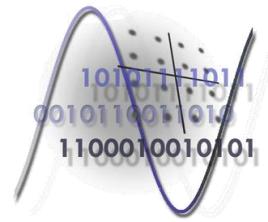
- Rețelele wireless sunt proiectate în general pornindu-se de la modelul rețelelor „wired”
- canalul wireless este abstractizat ca și o legătură punct la punct și sunt controlate de protocoale de comunicații preluate din rețelele fixe
- există o redundanță semnificativă a datelor într-o rețea wireless, deoarece există o suprapunere semnificativă a datelor care sunt disponibile la diferite noduri.
- Soluția posibilă: o proiectare alternativă a rețelelor wireless care să poată exploata caracteristicile intrinseci cum ar fi diversitatea spațială și redundanța datelor, în loc să forțeze o abstractizare artificială corespunzătoare rețelelor pe cablu.



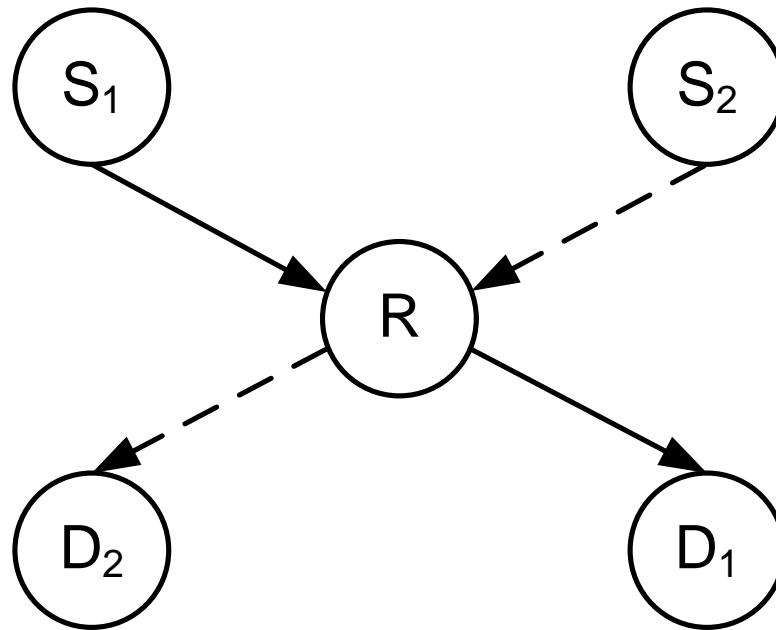
- Pentru explicarea modului în care se poate utiliza tehnica NC în rețelele wireless se consideră următorul exemplu concret

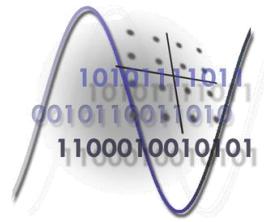


- Operațiile de codare reprezintă aşa numitul „*inter-flow network coding*” pentru că se face codarea unor pachete care au nexthop diferit, deci aparțin unor fluxuri diferite.



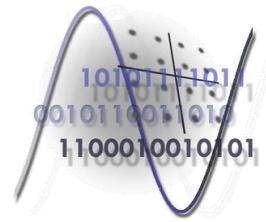
- Ideea prezentată se poate extinde simplu la un scenariu mai complex, în care un singur router deservește mai multe stații



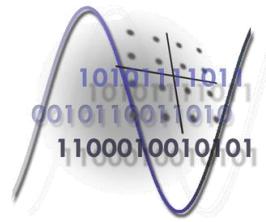


- Utilizarea codării de tip NC în rețelele wireless nu este însă aşa de simplă precum pare în exemplele simple prezentate.
- Simpla combinare a pachetelor recepționate de la diverse stații și distribuirea oarbă a acestor pachete, fără să se cunoască nimic despre pachetele pe care le detin stațiile cărora le este destinat pachetul codat nu va îmbunătăți substanțial sau chiar va reduce performanțele.
- Din acest motiv alegerea soluției de codare trebuie să fie bazată pe o cunoaștere a topologiei locale a rețelei și a datelor (pachetelor) existente în stațiile vecine, aflate în aria de acoperire radio

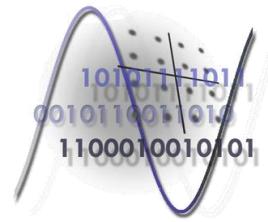
Creșterea throughput-ului în rețelele wireles



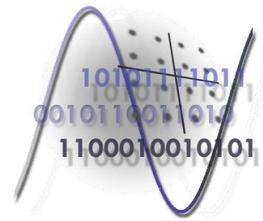
- codarea de tip NC poate asigura o creștere substanțială a throughput-ului, dacă operațiile de codare se efectuează în mod corespunzător
 - La baza creșterii throughput-ului se află posibilitatea reducerii numărului de intervale de timp în care se poate realiza transmiterea datelor, pachetele codate transportând combinația mai multor pachete de date originale.



- Îmbunătățirea procesului de transfer a pachetelor de date într-o rețea wireless descentralizată este bazată pe două elemente fundamentale:
- *Așteptare oportunistă (Opportunistic Listening)*: se exploatează natura distribuită a mediului wireless, care asigură automat distribuirea pachetelor transmise într-o anumită vecinătate, adică în zone de receptie radio.
 - Fiecare nod stochează toate pachetele pe care le „aude” un interval limitat de timp.
 - anunță vecinii asupra pachetelor pe care le deține prin adnotarea pachetelor de date pe care le trimite. În acest fel se creează premise pentru codare deoarece nodurile rețelei vor deține pachete (date) care se suprapun parțial cu pachetele deținute de nodurile vecine.

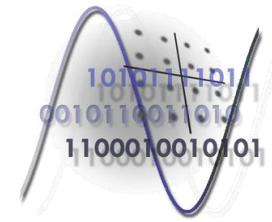


- *Codare oportunistă (Opportunistic Coding)*: când un nod trimite un pachet utilizează informațiile pe care le are despre pachetele vecinilor pentru a transmite pachete multiple într-un singur acces la mediu, combinând aceste pachete într-un singur pachet codat.
 - Transmisia codată are loc numai dacă nodurile vecine (nodurile „nexthop”) au destulă informație pentru a decoda pachetele codate.
 - fiecare nod folosește următoarea regulă de codare: pentru a transmite n pachete, p_1, \dots, p_n , la n noduri „nexthop”, r_1, \dots, r_n , un nod poate aduna modulo doi n pachete pe care le deține numai dacă fiecare nod „nexthop” r_i are $n - 1$ pachete din setul în discuție mai puțin pachetul pe care îl dorește.



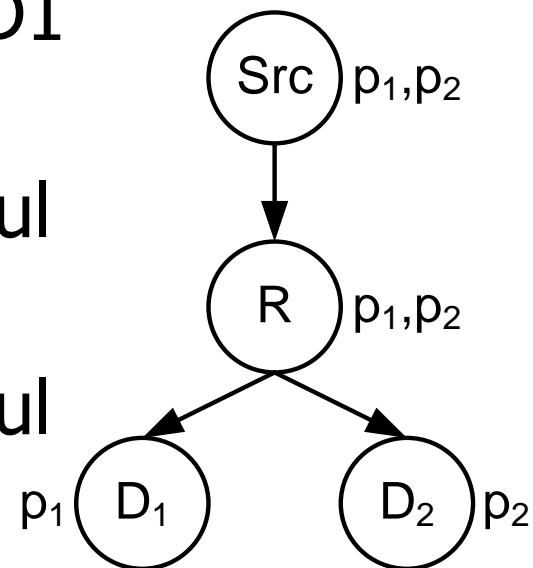
Creșterea calității transmisiei

- Metoda consacrată de asigurare a calității transmisiei în rețelele curente constă în retransmiterea pachetelor pierdute
- O soluție alternativă este codarea NC între pachetele aceluiași flux (*intraflow network coding*)
 - ruterele combină pachetele trimise către aceeași destinație
 - fiecare pachet recepționat conține informații despre toate pachetele din fișierul original
 - Este necesar numai un feedback de la receptie că s-a recepționat numărul corespunzător de pachete pentru decodare

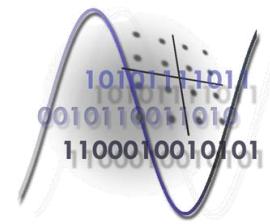


Operații de tip multicast

- În operațiile de multicast este posibil ca noduri diferite să recepționeze corect pachete diferite.
- R trimite pachetele p_1 și p_2 la stațiile D_1 și D_2 .
- D_1 recepționează corect numai pachetul p_1 ,
- D_2 recepționează corect numai pachetul p_2
- atunci o singură retransmisie de la nodul R , conținând $p_1 \oplus p_2$ poate asigura recuperarea datelor la ambele destinații.

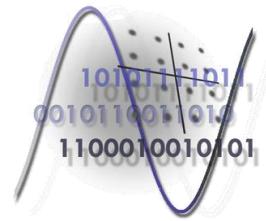


Asigurarea unor condiții echivalente pentru toate transmisiile

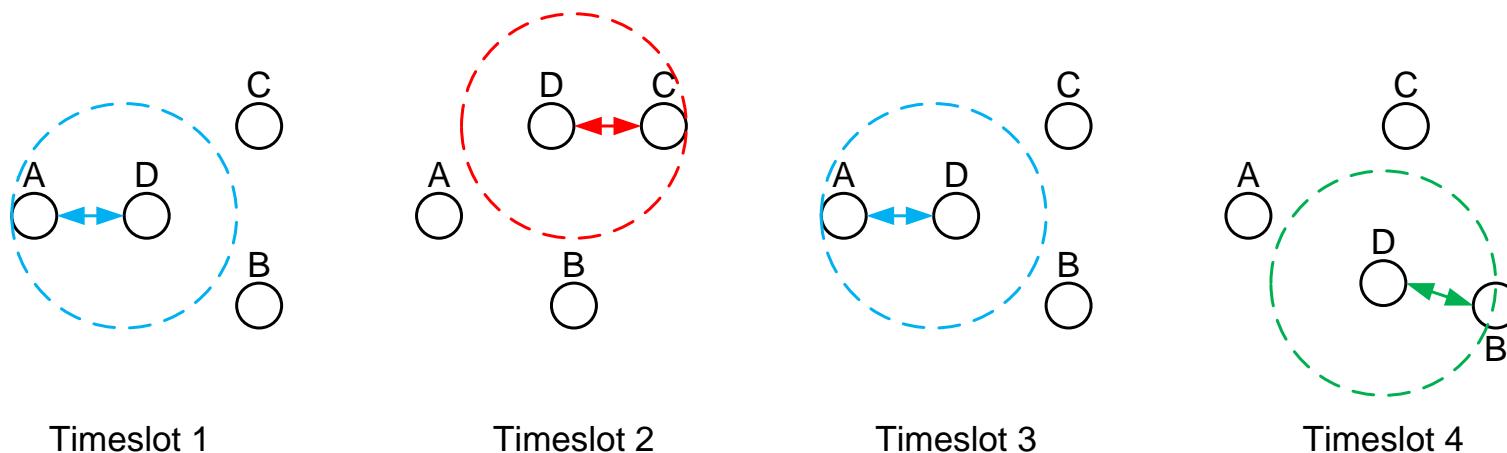


- O caracteristică de bază a rețelelor wireless o reprezintă variabilitatea în timp a calității semnalului recepționat, afectat de interferențe și fading.
 - Fluctuațiile semnalului radio determină pierderi de pachete (erasures) la nivelele superioare ale stivei de protocoale pe intervale variabile de timp (de regulă intervale scurte de timp).
- Aplicațiile în timp real, și nu numai aceste aplicații (de ex. Transferuri de date controlate de protocolul TCP) nu tolerează variații ale valorii debitului recepționat (datorat retransmisiilor) și nici pachetele pierdute, dacă rata de pierdere este peste o anumită limită.
- Prin utilizarea tehnicii NC combinată cu operații de broadcast, este posibil să se realizeze o „netezire” a variațiilor de debit sesizate de stațiile receptoare.

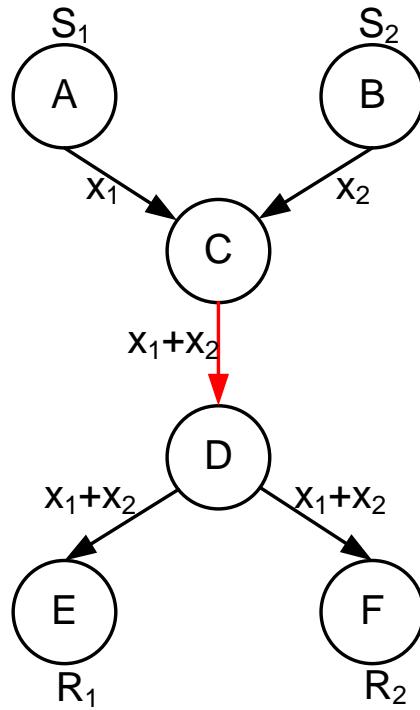
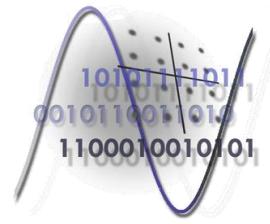
Asigurarea unei rutări simple în cazul mobilității terminalelor



- Într-un mediu mobil, în care configurația rețelei se schimbă frecvent și pe durata unor intervale reduse de timp realizarea eficientă a rutării este complicată
- O simplificare substanțială se poate obține prin utilizarea tehniciilor NC



Monitorizarea stării legăturilor wireless



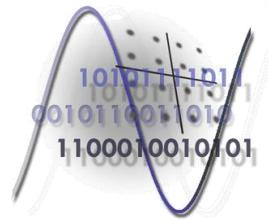
- Utilizarea combinațiilor dintre pachetele de test generate de surse diferite permite implementarea unei monitorizări mai eficiente a unui rețele wireless.

TACCFDRT Curs 7

MIXIT

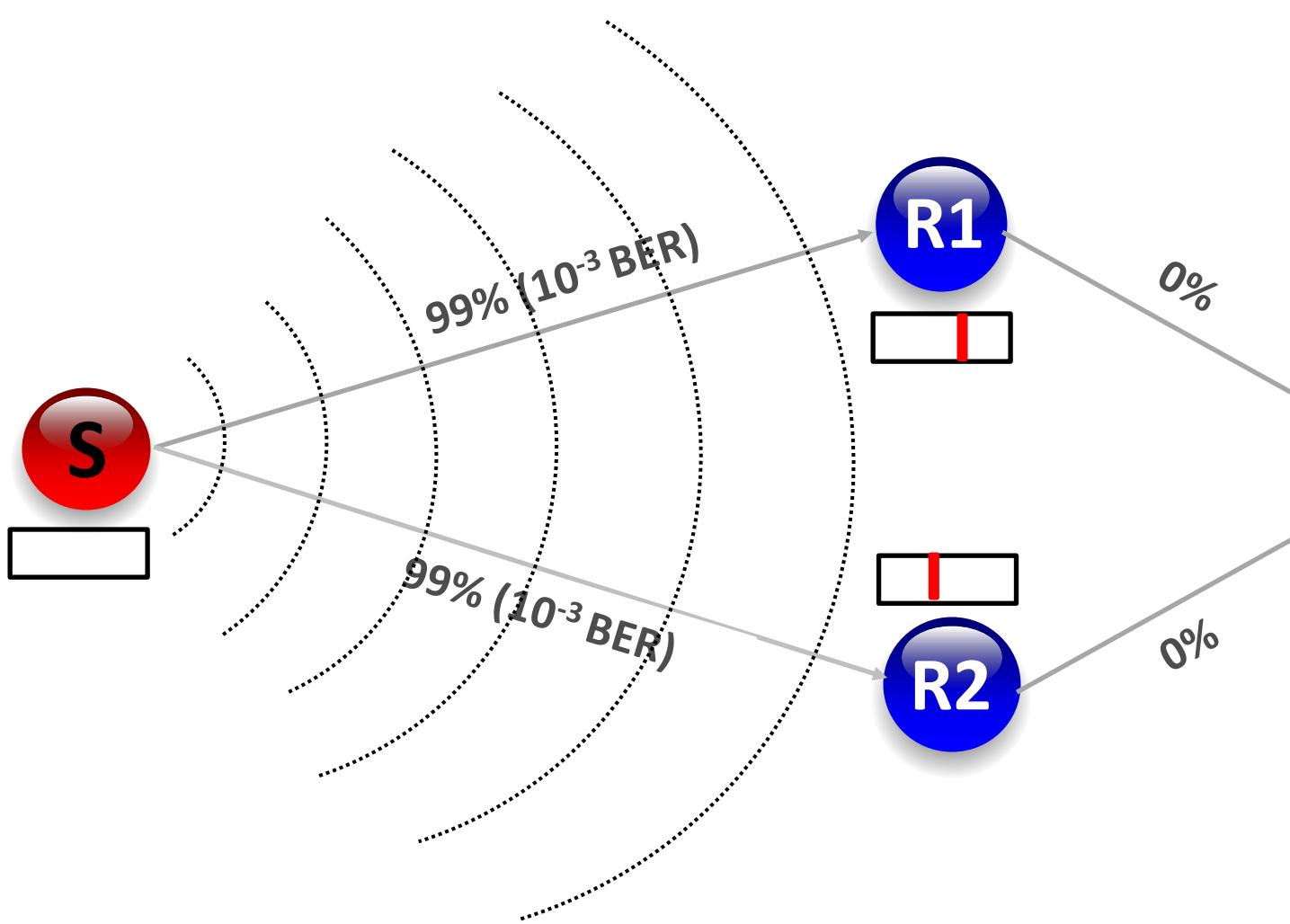
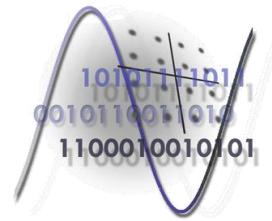


0010110011010
1100010010101

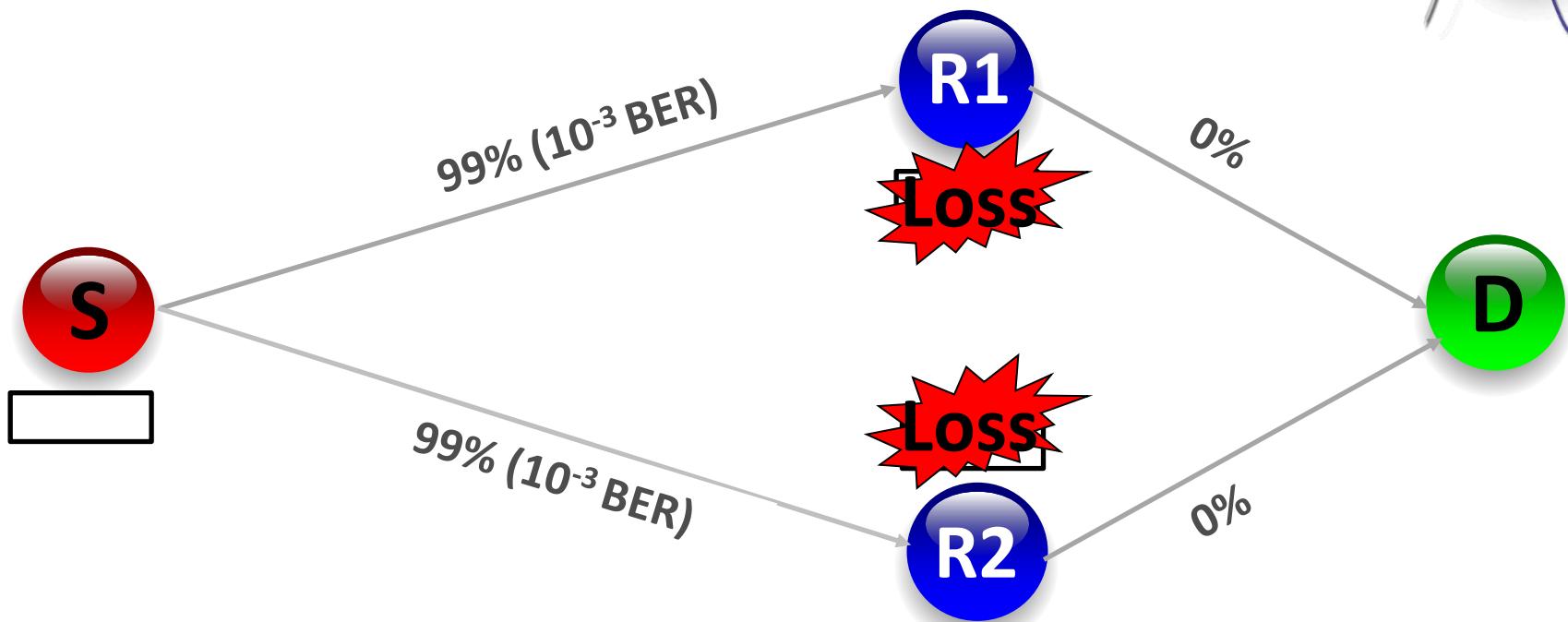
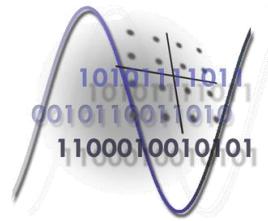


Cuprins

- Tehnici de codare de tip “Xor in the Air”
 - MIXIT

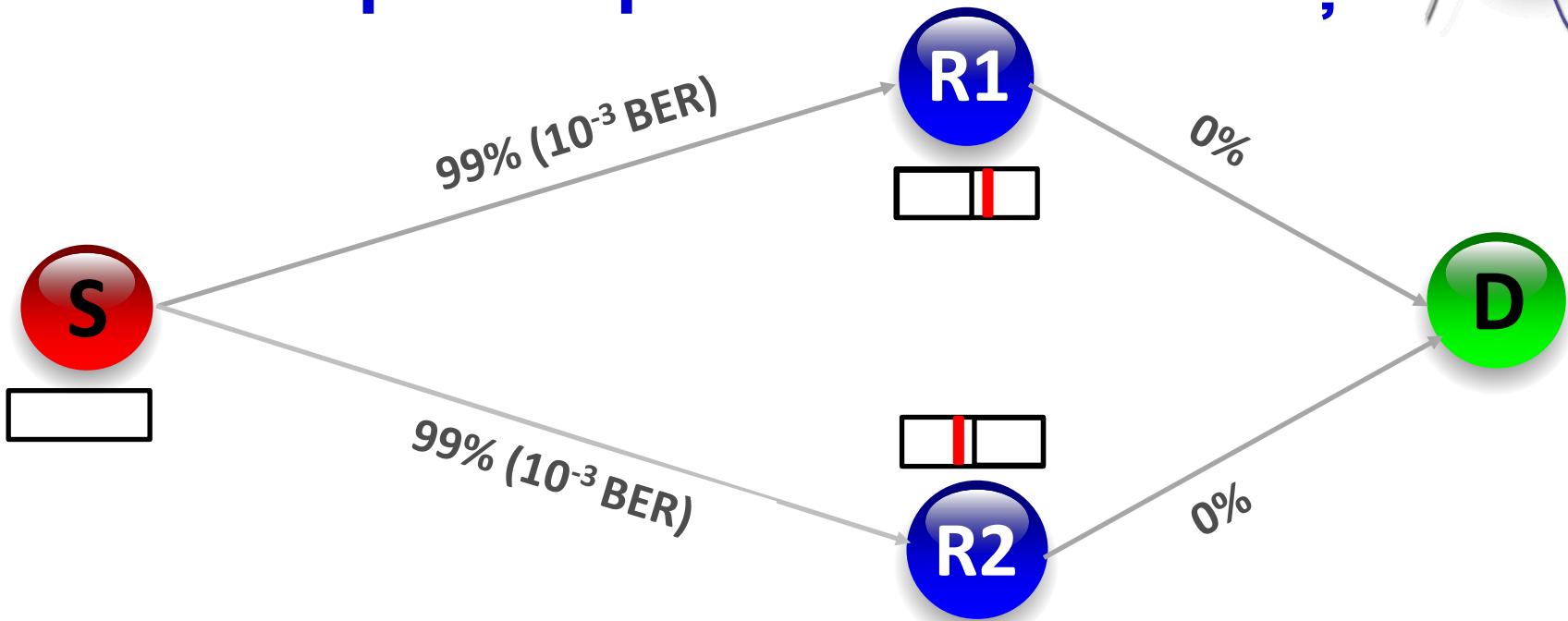
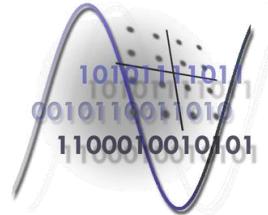


Chiar daca un bit este eronat din 1000 → 99% probabilitate de eronare a pachetelor



**Soluția „clasică” → Asigurarea fiabilității pe fiecare legătură
→ 50 re-transmisi.....**

Asigurarea fiabilității Rețelelor Wireless prin exploatarea diversității

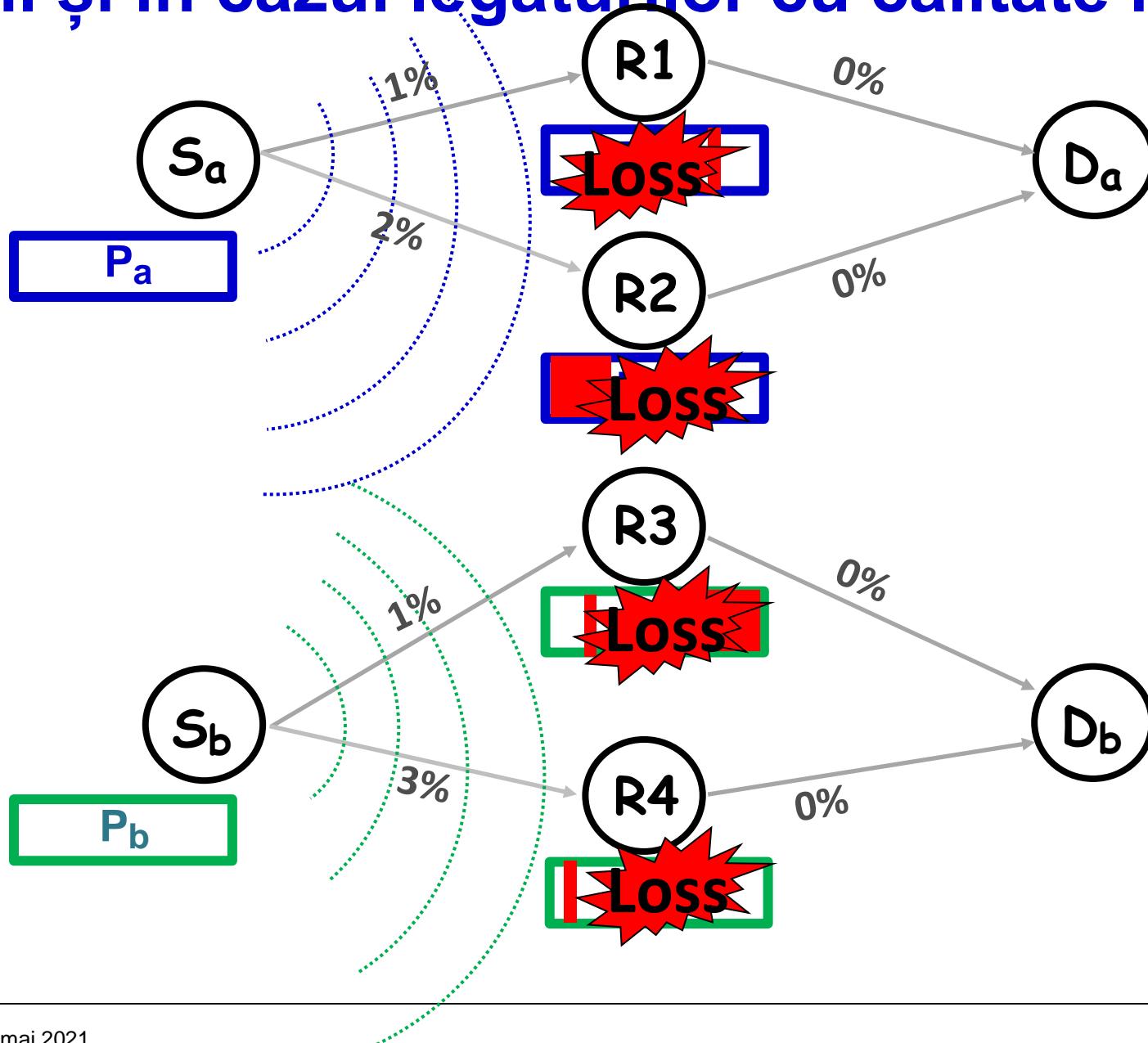
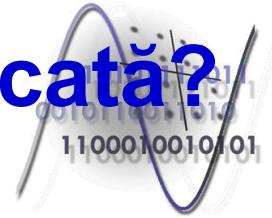


Diversitate spațială → 50 tx → throughput scăzut

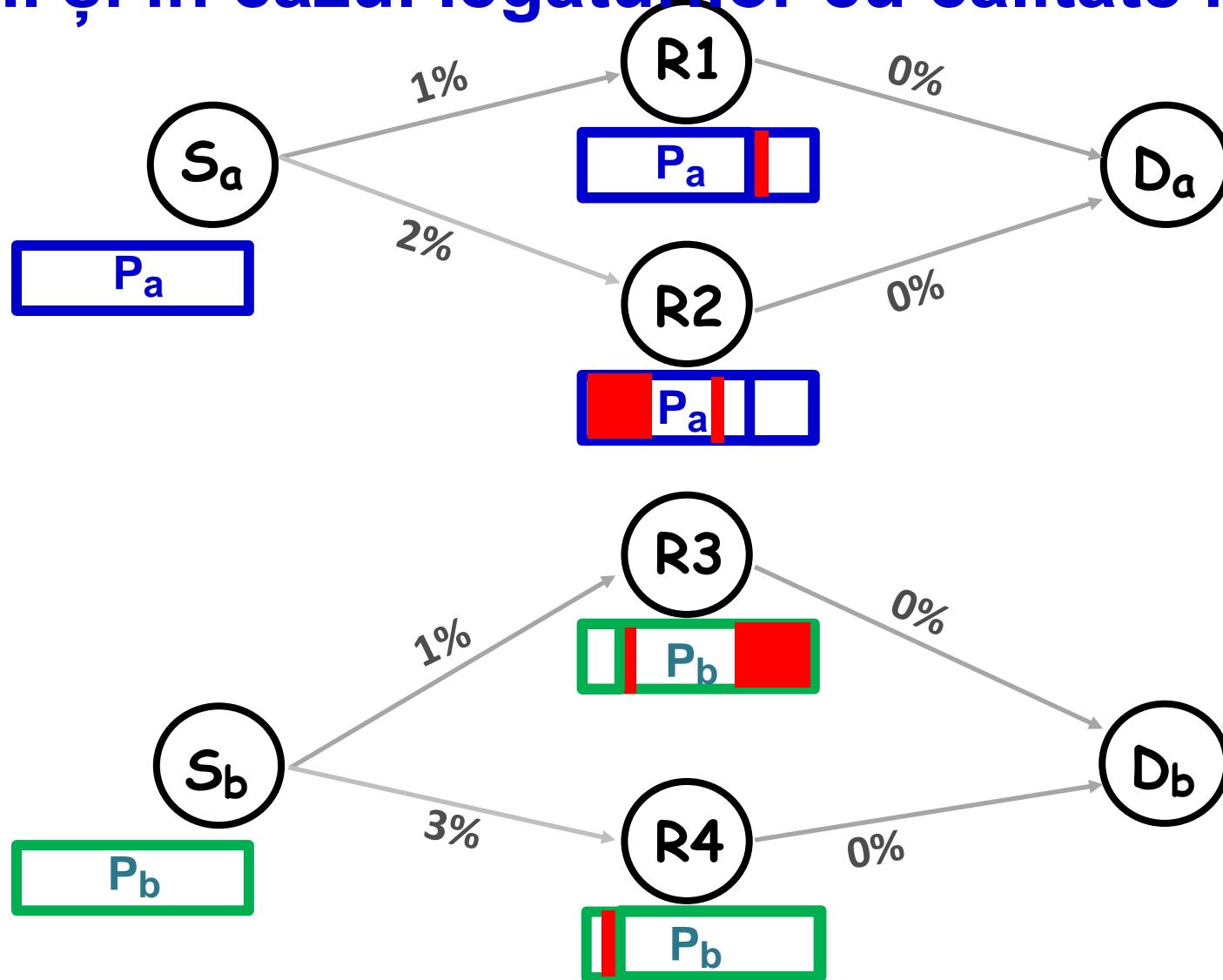
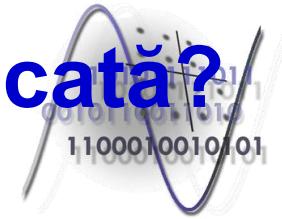
fiecare bit este recepționat corect de un nod al rețelei

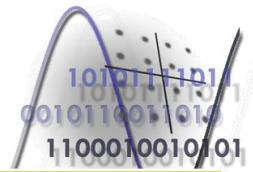
Explorând caracteristicile rețelei wireless → 3 tx → throughput ridicat

Util și în cazul legăturilor cu calitate ridicată?



Util și în cazul legăturilor cu calitate ridicată?



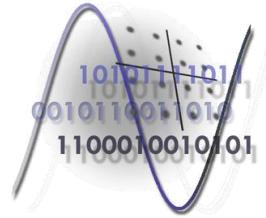


Soluția clasică

PHY + LL	Păstrează pachetele corecte
Network	Retransmite pachetele corecte către destinație

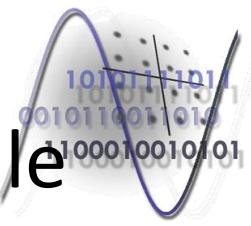
Exploatând noile paradigmă...

PHY + LL	Furnizează simbolurile corecte către nivelele superioare
Network	Retransmite pachetele corecte către destinație Throughput ridicat, competitivitate ridicată....

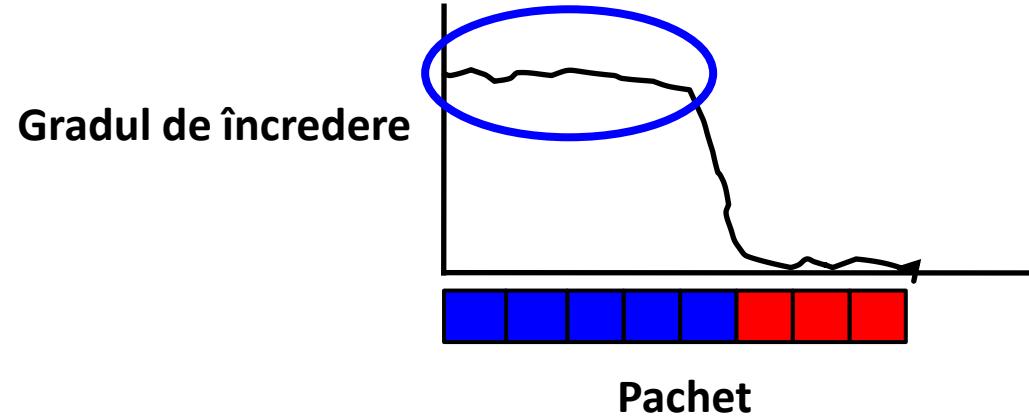


- Alt tip de colaborare dintre straturile rețelei pentru combaterea erorilor în rețele fără-fir
- Network coding la nivel de simbol care asigură transportul simbolurilor corecte către destinație
- MAC cu competitivitate ridicată
- În urma implementării și evaluării
 - Câștig 3-4x față de *shortest path routing*
 - Câștig 2-3x față de *rutare op. la nivel de pachet*

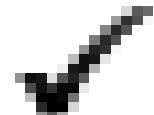
Cum identifică router-ul simbolurile corecte?



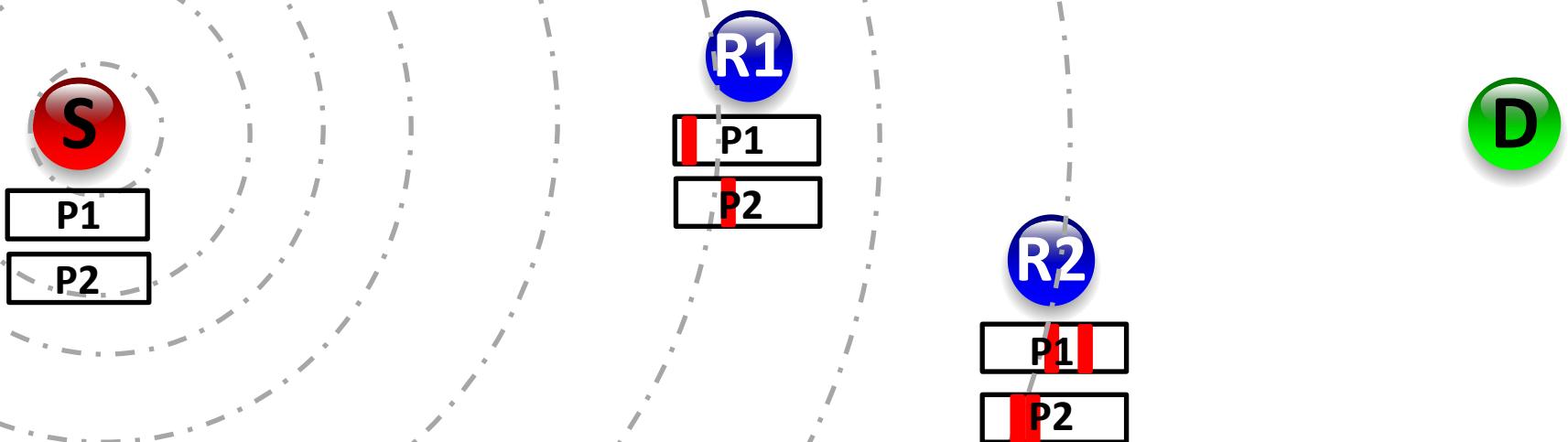
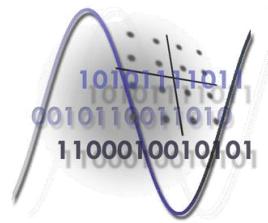
- PHY oricum estimează gradul de încredere ale simbolurilor recepționate
- PHY + LL furnizează numai simbolurile cu grad ridicat de încredere



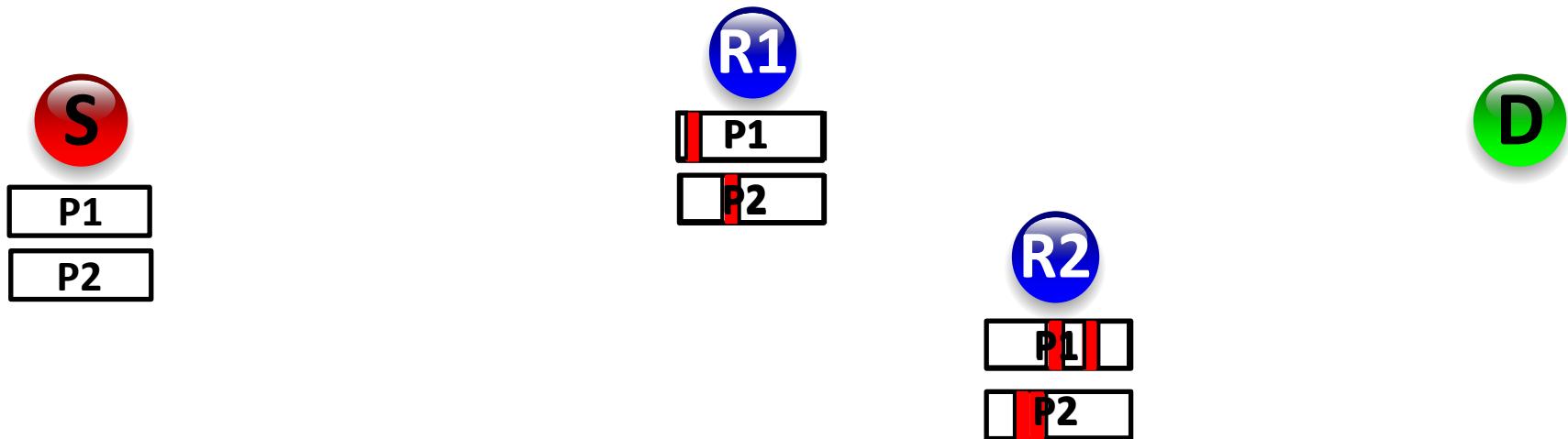
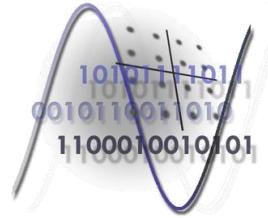
PHY + LL	Furnizează simbolurile corecte către nivele superioare
Network	Retransmite simbolurile corecte către destinație



Ce să retransmîtă nodurile intermediare?



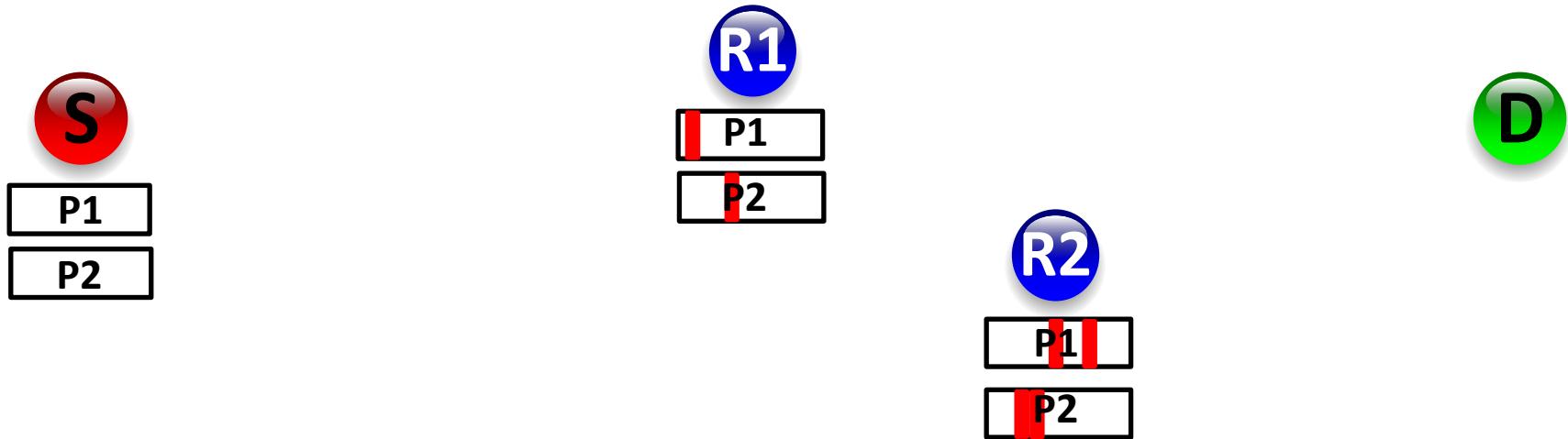
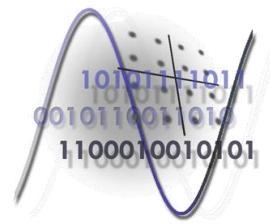
Ce să retransmîtă nodurile intermediare?



Suprapuneri între simboluri corecte
Soluții posibile

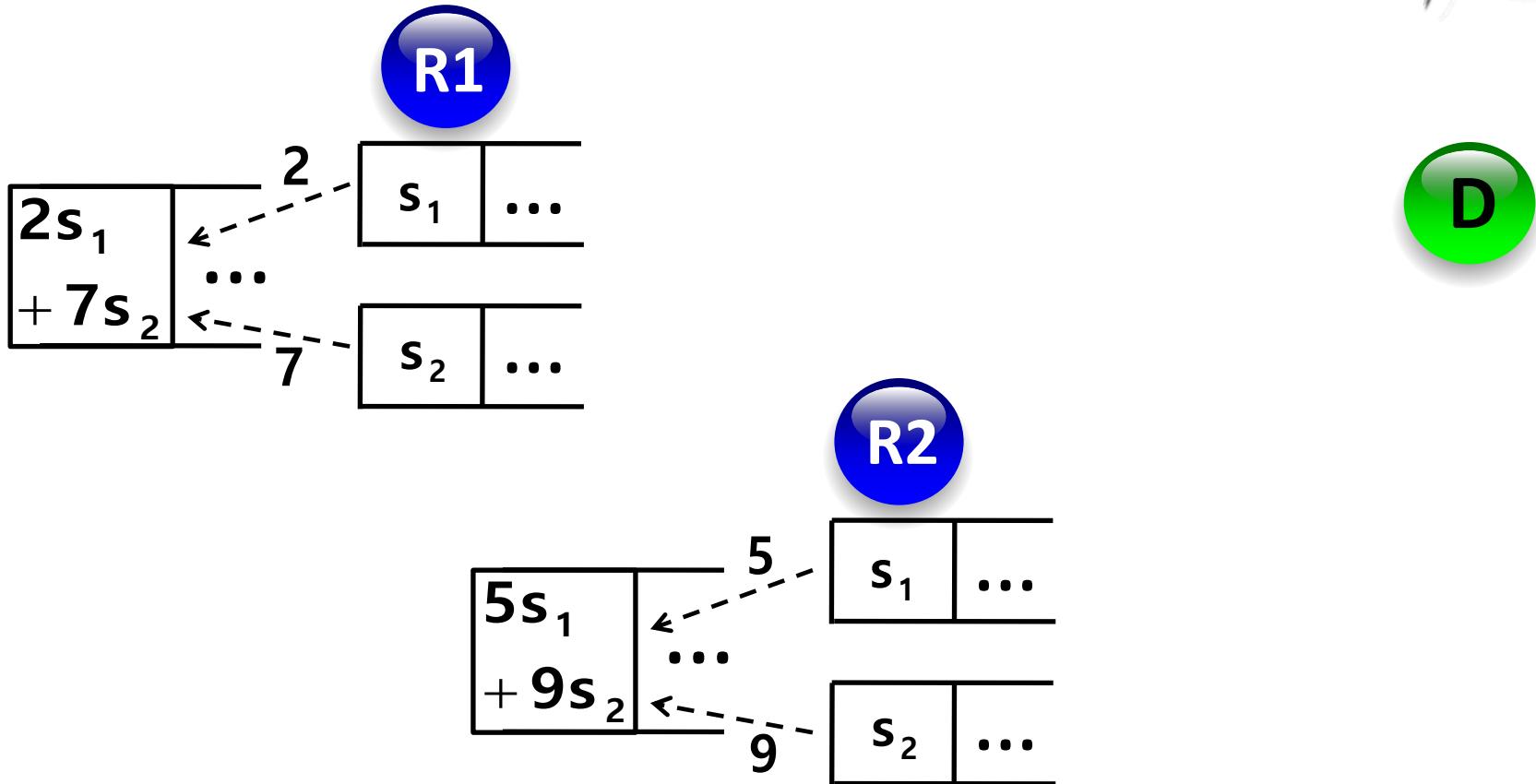
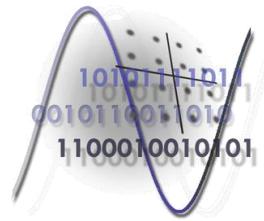
- 1) Să retransmîtă tot ce este corect → Ineficient
- 2) Retransmisie coordonată → Ne-scalabil

MIXIT elimină duplicatele utilizând Network Coding la nivel de simbol



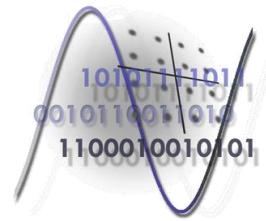
Retransmite combinații aleatoare ale simbolurilor corect recepționate

MIXIT elimină duplicatele utilizând Network Coding la nivel de simbol



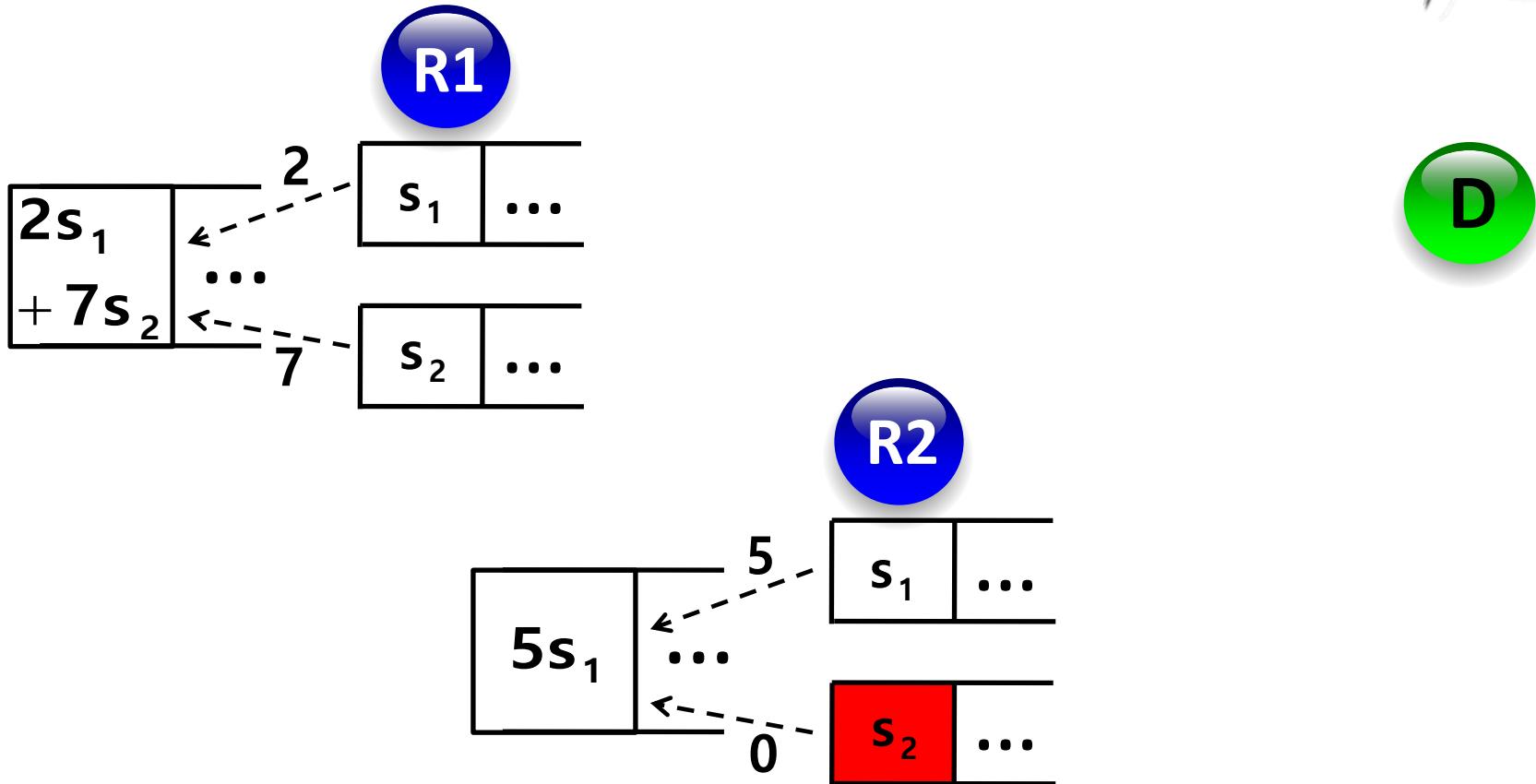
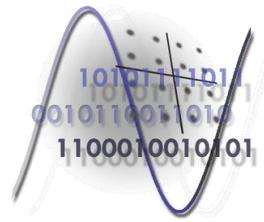
Routerurile creează combinații aleatoare ale simbolurilor corect recepționate

MIXIT elimină duplicatele utilizând Network Coding la nivel de simbol



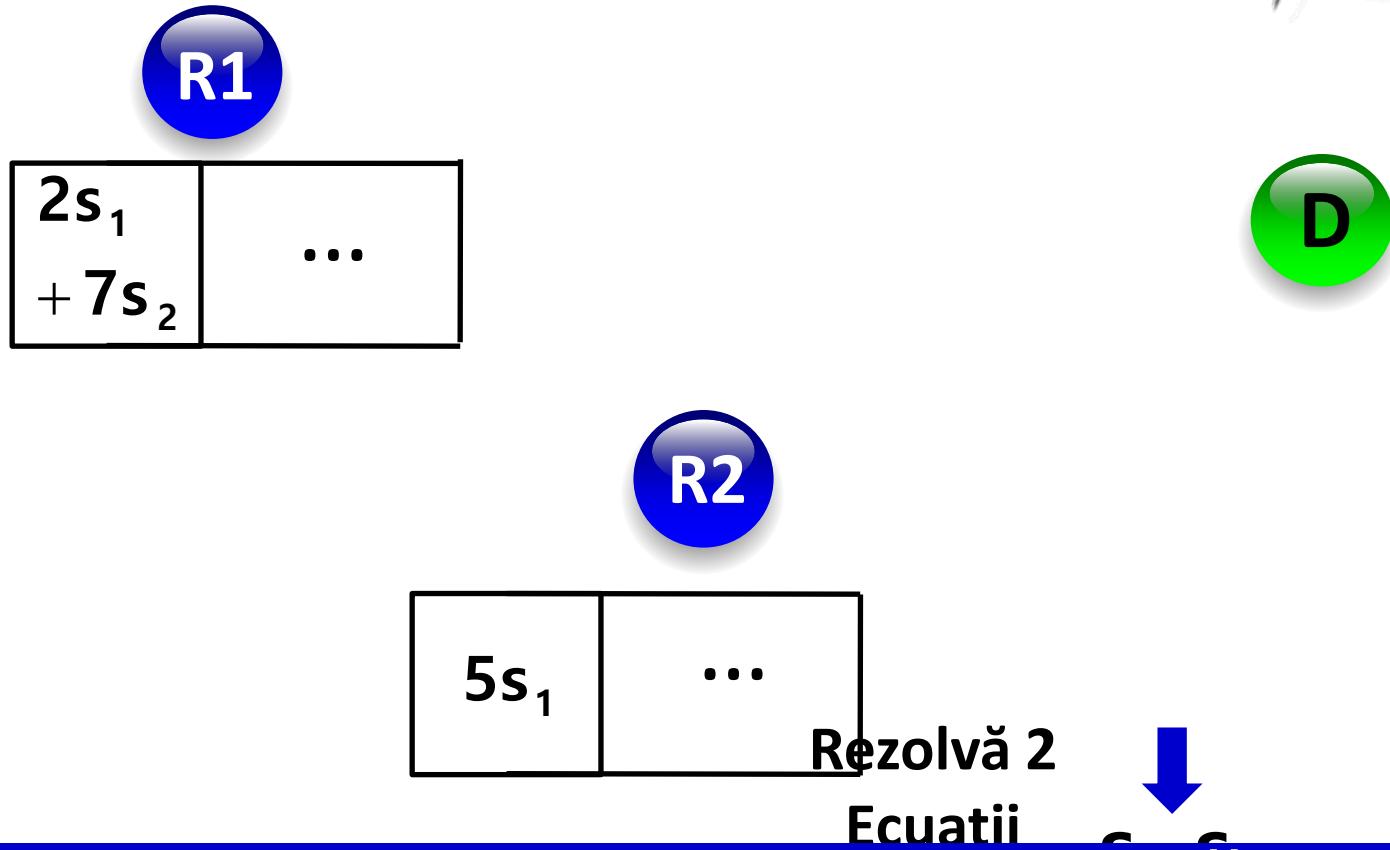
Aleatorizarea elimină transmisii multiple fără coordonare

MIXIT elimină duplicatele utilizând Network Coding la nivel de simbol



Routerurile creează combinații aleatoare ale simbolurilor corect recepționate

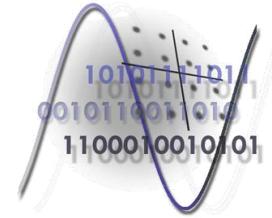
MIXIT elimină pachete duplicate prin Symbol Level Network Coding



Network Coding la nivel de simbol asigură:

- Eliminarea transmisiilor multiple → Eficient
- Nu necesită coordonare → Scalabil

Destinația trebuie să cunoască ce combinație a recepționat



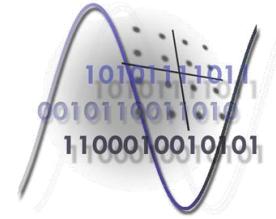
5s₁ + 9s₂ (daca ambele simboluri sunt corecte)

5s₁ + 0s₂ (daca numai s₁ este corect)

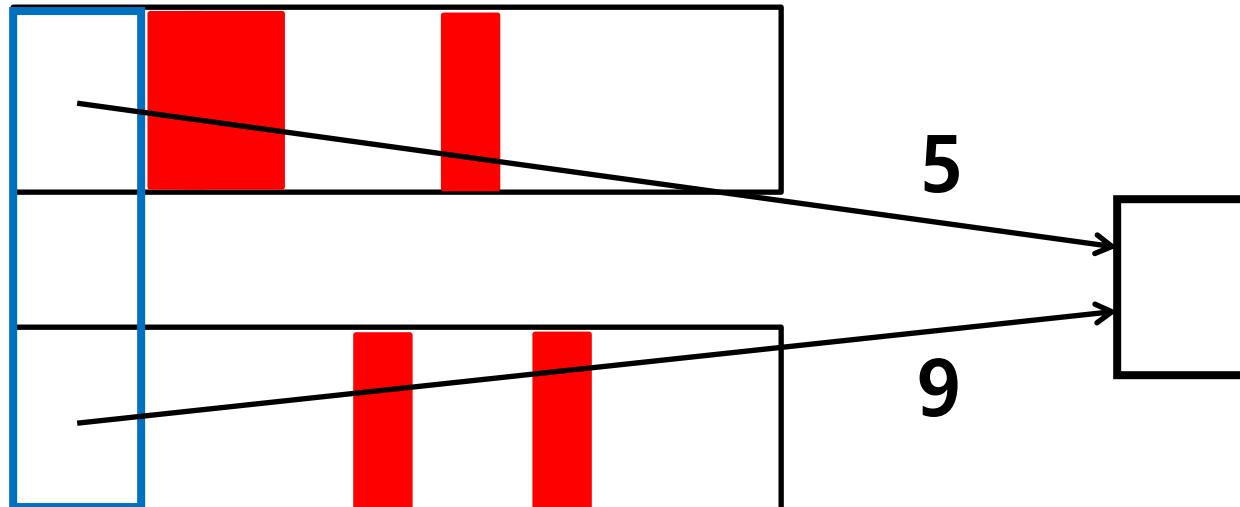
0s₁ + 9s₂ (daca numai s₂ este corect)

Nimic (Dacă nici unul nu este corect)

Destinația trebuie să cunoască ce combinație a recepționat



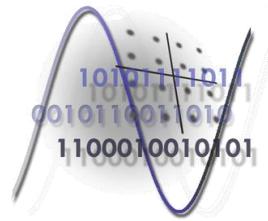
Utilizează *run length encoding*



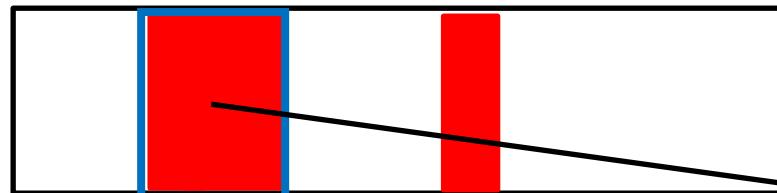
Pachetele originale

Pachetul codat

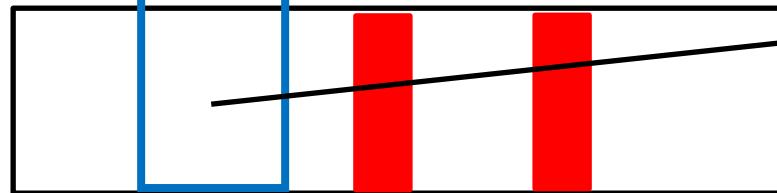
Destinația trebuie să cunoască ce combinație a recepționat



Utilizează *run length encoding*

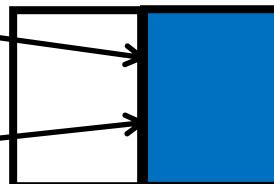


0



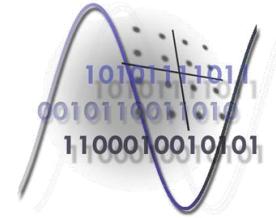
9

Pachetele originale

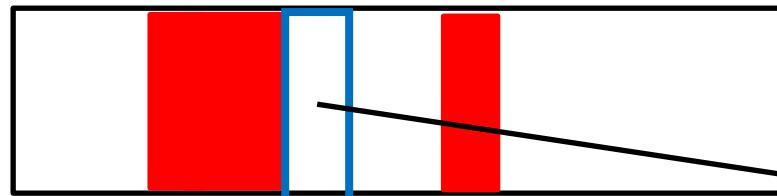


Pachetul codat

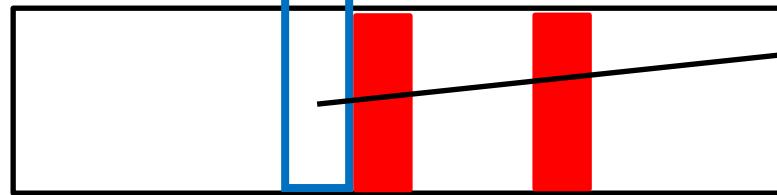
Destinația trebuie să cunoască ce combinație a recepționat



Utilizează *run length encoding*

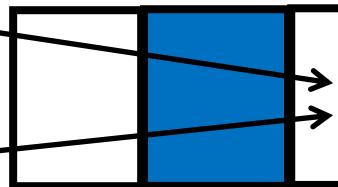


5



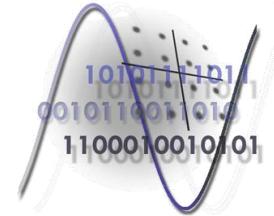
9

Pachetele originale

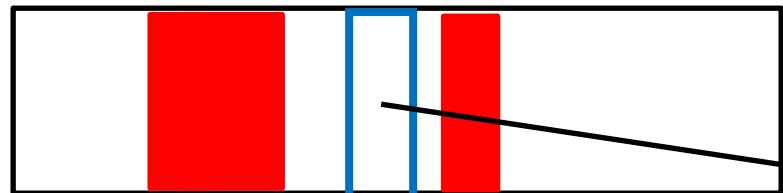


Pachetul codat

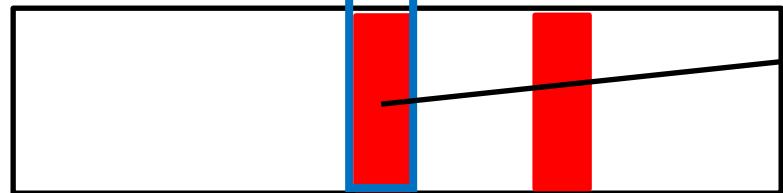
Destinația trebuie să cunoască ce combinație a recepționat



Utilizează *run length encoding*

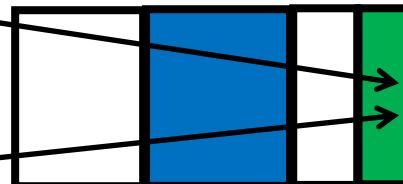


5



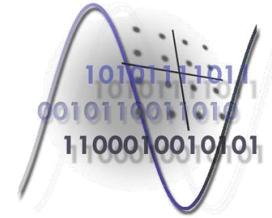
0

Pachetele originale

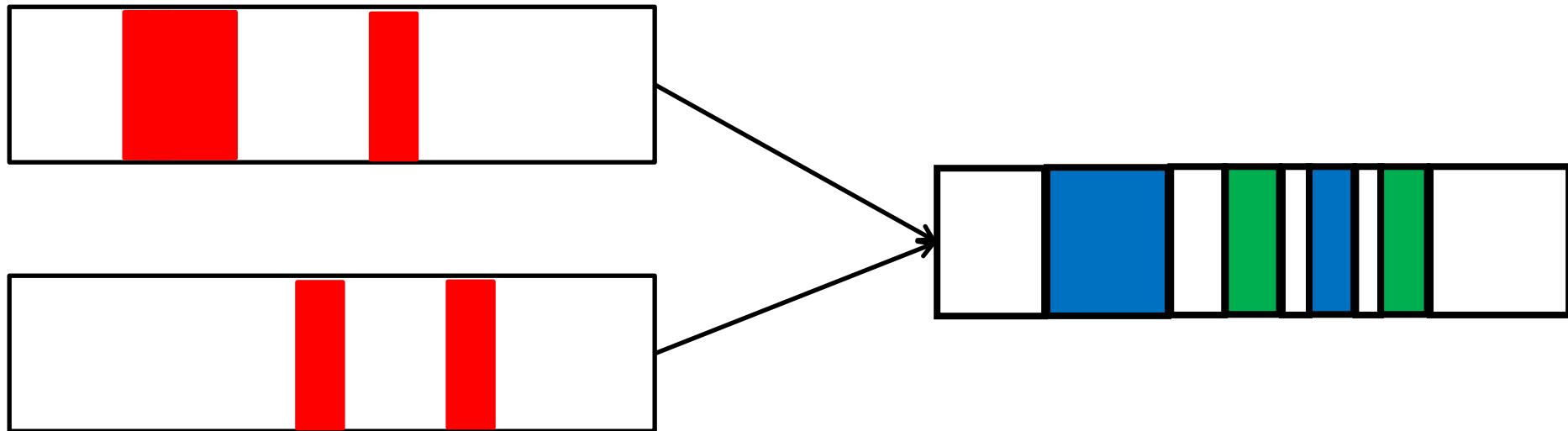


Pachetul codat

Destinația trebuie să cunoască ce combinație a recepționat



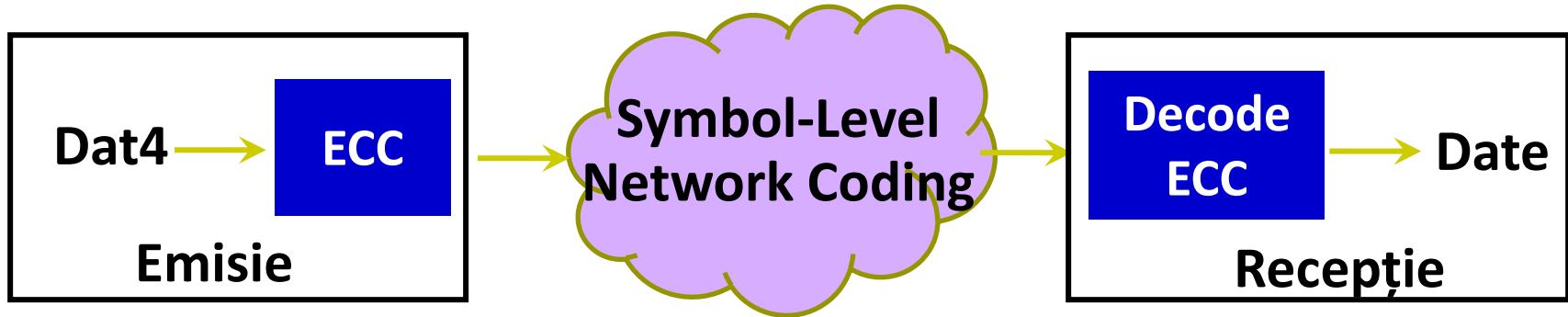
Utilizează *run length encoding*



Run length encoding descrie eficient combinațiile posibile

Ruterele pot să retrasmă biții eronați chiar și în cazul unui grad ridicat de încredere

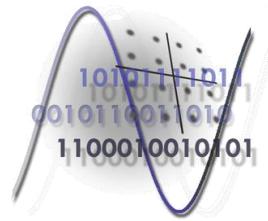
MIXIT are capabilitate de corecție E2E!



Capabilitate (ECC Error Correcting Code)

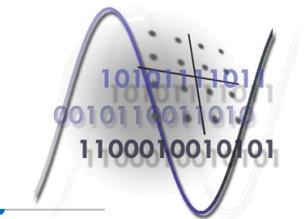
1. Ruterele sunt transparente pentru ECC
2. Capabilitatea Optimă de corecție
3. Rateless

PHY + LL	Straturile superioare receptioneaza numai simbolurile corecte
Network	Retransmite numai simbolurile corecte

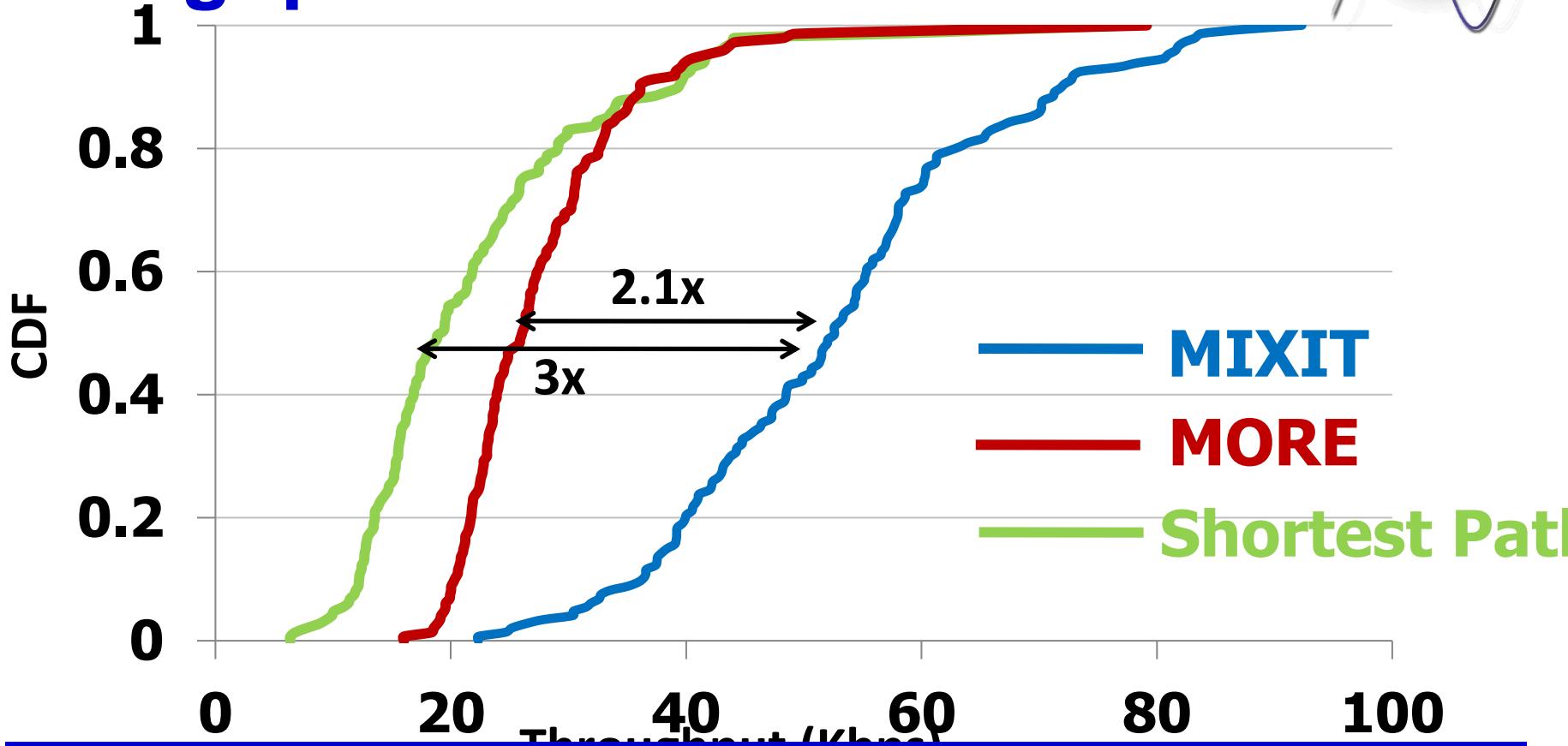


Evaluarea

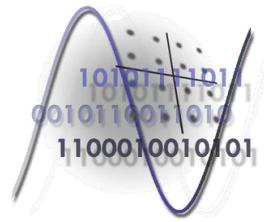
- Implementare în GNURadio pe SDR și USRP
- Zigbee (IEEE 802.15.4) link layer
- 25 noduri
- Scenarii de referință:
 1. Shortest path routing
 2. MORE: rutare oportunistă la nivel de pachete-MORE



Throughput



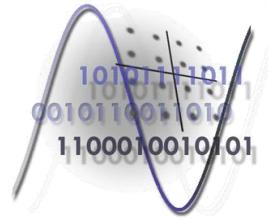
Throughput: 3x - SPR, 2x - MORE



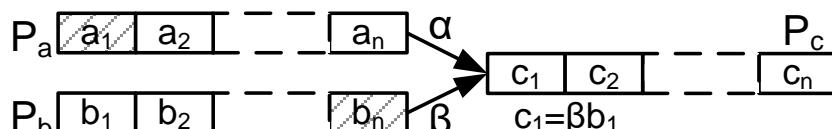
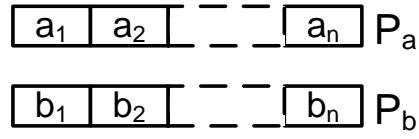
MIXIT

- Componenta de bază a arhitecturii MIXIT este un cod de tip NC la nivel de simbol care funcționează ca și un cod corector de erori de tip rateless.
- se încearcă simplificarea soluțiilor pentru două probleme majore:
 - Identificarea simbolurilor recepționate de către fiecare nod pentru a se preveni transmisii duplicate – poate fi o sarcină de coordonare foarte complexă a nodurilor.
 - Prin utilizarea codurilor NC la nivel de simbol, routerele transmit combinații liniare ale simbolurilor recepționate, reducându-se astfel probabilitatea de a trimite informații duplicate și se elimină necesitatea coordonării nodurilor
- Chiar dacă routerele trimit numai simboluri care au fost decodate cu grad ridicat de încredere există probabilitatea de a se trimite simboluri greșite. Codurile NC la nivel de simbol acționează și ca coduri corectoare de erori rateless, asigurându-se o redundanță adaptivă pentru corecția simbolurilor eronate

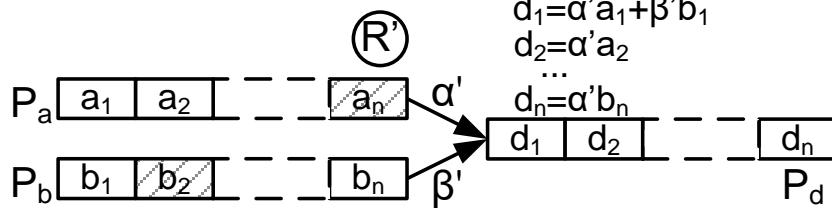
MIXIT



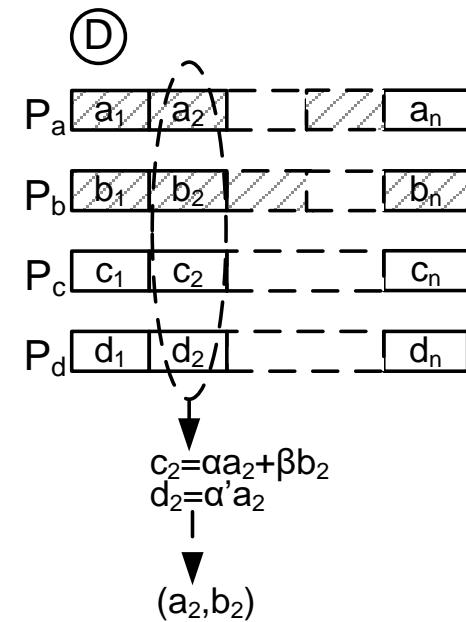
(S)

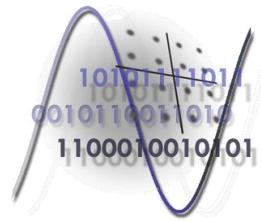


$$\begin{aligned} c_1 &= \beta b_1 \\ c_2 &= \alpha a_2 + \beta b_2 \\ \dots \\ c_n &= \alpha a_n \end{aligned}$$



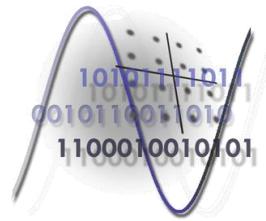
$$\begin{aligned} d_1 &= \alpha' a_1 + \beta' b_1 \\ d_2 &= \alpha' a_2 \\ \dots \\ d_n &= \alpha' b_n \end{aligned}$$





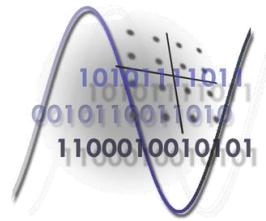
MIXIT

- *Componentele arhitecturii. Caracteristici*
 - **Sursa:** Nodul sursă transmite fișierele în grupuri de K pachete. Sursa creează combinații lineare aleatoare ale celor K pachete din grup și transmite pachetele codate
 - La fiecare pachet codat trebuie atașat un antet separat care arată, simbolurile care au fost combinate pentru formarea pachetului codat, respectiv cum s-a realizat procesul de codare
 - Antetul conține de asemenea și lista de relee/rutere (numiți și forwarderi)

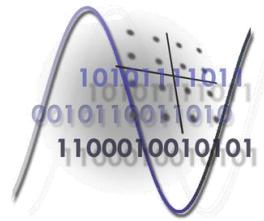


● Releele:

- Când un nod recepționează un pachet verifică dacă el este în lista de relee pentru pachetul respectiv (forwarders list). Dacă această condiție este îndeplinită nodul verifică dacă pachetul recepționat este inovativ, adică conține informație nouă
- retele creează combinații liniare ale simbolurilor corect recepționate, aceste combinații vor fi transmise mai departe. Astfel pachetele transmise vor conține numai simbolurile corecte
- În antetul noului pachet codat trebuie să fie inclusi coeficienții de codare globali



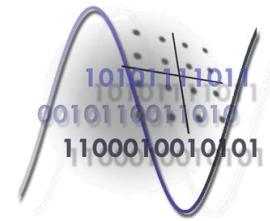
- **Destinația:** Trebuie să recupereze simbolurile originale din simbolurile codate recepționate utilizând algoritmi de decodare standard.
 - Odată ce simbolurile originale sunt recuperate destinația le reassemblează în pachetele originale și trimit un semnal ACK către sursă pentru a permite generarea următorului grup de pachete.
 - Semnalele ACK sunt transmise utilizând o rutare clasică bazată pe selecția celei mai bune căi.
 - Se poate asocia o prioritate mărită pentru pachetele ACK și se pot proteja aceste pachete cu coduri FEC.



În ce situații ajută MIXIT?

- Simulații/studii efectuate arată că se pot obține câștiguri de throughput între 1.2 și 8.
 - Din studii rezultă că se obțin câștiguri reduse de throughput în situația în care majoritatea legăturilor wireless sunt bimodale, adică fie sunt foarte bune, cu probabilitate redusă de eroare pe simbol, fie sunt foarte proaste, având o probabilitate ridicată de eroare pe simbol.
- Utilizând tehnica MIXIT se pot obține câștiguri de throughput de aproximativ de 4 ori relativ la alte metode de rutare oportunistă, dacă nu se consideră antetul mai mare al pachetelor MIXIT.
- În realitate acest câștig trebuie considerat o limită superioară.

Cerințe cerute de aplicarea practică a NC în rețele wireless multihop



- *Capacitatea de a face față la un trafic variabil și un mediu dinamic*
- *Operații de Broadcast cu evitarea coliziunilor*
- *Algoritmi de codare și de decodare de complexitate redusă*
- *Se poate utiliza în mod transparent cu TCP*
 - recuperarea pachetelor pierdute
 - reordonarea pachetelor