Département d'informatique et de génie logiciel

Compression de données
IFT-4003/IFT-7023

Notes de cours Implémentation entière du codage arithmétique

Édition Hiver 2012

Mohamed Haj Taieb

Local: PLT 2113

Courriel: mohamed.haj-taieb.1@ulaval.ca

Faculté des sciences et de génie
Département de génie électrique et de génie informatique

UNIVERSITÉ LAVAL

Plan de la présentation

- Implémentation entière:
  - Les fonctions du mapping
  - Implémentation de l'encodeur
  - Exemple d'implémentation de l'encodeur
  - Implémentation du décodeur
  - Exemple d'implémentation du décodeur

Fonctions de mapping (1)

- Fonctions de redimensionnement
  - Nous avons défini préalablement seulement deux fonctions de redimensionnement lorsque le tag se trouve soit dans la moitié inférieure soit dans la moitié supérieure.
  - On rajoute une 3ème fonction lorsque le tag contient le point milieu et qu'il soit de largeur inférieure au quart de l'intervalle original:

E_1:[0, 0.5)→[0,1); $E_1(x)=2x$

E_2:[0.5, 1)→[0,1); $E_2(x)=2(x-0.5)$.

E_3:[0.25, 0.75)→[0,1); $E_3(x)=2(x-0.25)$

Fonctions de mapping (2)

- Comment informer le décoder lorsqu'on applique la fonction de redimensionnement E_3?
  - E_1: envoi de 0
  - E_2: envoi de 1
  - E_3: on procède autrement.
  - Si après E_3 on a E_1 : on envoie 0 1.
  - Si après E_3 on a E_2 : on envoie 1 0.
  - E_3 E_3 E_3 E_1 : 0 0 0 1.
  - E_3 E_3 E_3 E_2 : 1 1 1 0.
- Du coté du décodeur
  - Comme le décodeur imite l'encodeur chaque fois que le tag se trouve dans [0.25, 0.75), E_3 est appliqué.

Implémentation entière de l'encodeur (1)

- Longueur du mot code m
  - Première étape: décider de la longueur du mot code m.
  - Pour une longueur m on représente 2^m valeurs possibles de [0,1).
  - 0→ 00…0 m fois.
  - 1→ 11…1 m fois.
  - 0.5→ 10…0 (m-1) fois.
- Équations de mise à jour
  - Comme on utilise maintenant une arithmétique entière on doit remplacer les valeurs de $F_X(x)$ dans ces deux équations.

[Rappel: équations de mise à jour]

$$u^{(n)} = l^{(n-1)} + (u^{(n-1)} - l^{(n-1)})F_X(x_n)$$

$$l^{(n)} = l^{(n-1)} + (u^{(n-1)} - l^{(n-1)})F_X(x_n - 1)$$

Implémentation entière de l'encodeur (2)

[Rappel: équations de mise à jour]

$$u^{(n)} = l^{(n-1)} + (u^{(n-1)} - l^{(n-1)})F_X(x_n)$$

$$l^{(n)} = l^{(n-1)} + (u^{(n-1)} - l^{(n-1)})F_X(x_n - 1)$$

Quelques définitions

- n_j: nombre de répétition du symbole j dans une séquence de longueur total_count.
- La fonction cumulative devient alors: $F_X(k) = \frac{\sum_{j=1}^{k} n_j}{total\_count}$
- On définie le compteur cumulatif : $cumul\_count(k) = \sum_{j=1}^{k} n_j$
- Les équations de mise à jour deviennent:

$$u^{(n)} = l^{(n-1)} + \left\lfloor (u^{(n-1)} - l^{(n-1)} + 1) \times \frac{cumul\_count(x_n)}{total\_count} \right\rfloor - 1$$

$$l^{(n)} = l^{(n-1)} + \left\lfloor (u^{(n-1)} - l^{(n-1)} + 1) \times \frac{cumul\_count(x_n - 1)}{total\_count} \right\rfloor$$

$$l^{(k)}, u^{(k)} \in [\underbrace{00\cdots0}_{m \text{ fois}}, \underbrace{11\cdots1}_{m \text{ fois}}) = [0, 2^m)$$

Gérer les effets de l'arithmétique entière

Redimensionnement avec l'arithmétique entière (1)

- Redimensionnement E_1 et E_2= simple décalage
  - $l^{(n)}$ et $u^{(n)}$ dans la moitié supérieure: MSB de $l^{(n)}$ et $u^{(n)}$ est le même=1.
  - Application de E_2: décalage en dehors du MSB pour $l^{(n)}$ et $u^{(n)}$ et décalage en dedans par 0 pour $l^{(n)}$ et 1 pour $u^{(n)}$.
  - Exemple m=6: Application de E_2
  - $l^{(n)}$ =33=100001 et $u^{(n)}$ =54=110110
  - $l^{(n)}$ =~~1~~000010 = 2 et $u^{(n)}$=~~1~~101101 =45
  - Exemple m=6: Application de E_1
  - $l^{(n)}$ =1=000001 et $u^{(n)}$ =22=010110
  - $l^{(n)}$ =~~0~~000010 = 2 et $u^{(n)}$=~~0~~101101 =45

Redimensionnement avec l'arithmétique entière (2)

- Détection du redimensionnement E_3
  - Quant le tag ne se trouve ni dans la moitié inférieure ni supérieure et le tag se trouve dans $[0.25, 0.75)x2^m$, on applique le redimensionnement E_3.
  - $\rightarrow 0.25x2^m \leq l^{(n)} < 0.5x2^m$ et $0.5x2^m \leq u^{(n)} < 0.75x2^m$.
  - $0.25x\ 10\ldots0 \leq l^{(n)} < 0.5x10\ldots0 \rightarrow 0010\ldots0 \leq l^{(n)} < 010\ldots0$
  - $0.5x\ 10\ldots0 \leq l^{(n)} < 0.75x10\ldots0 \rightarrow 010\ldots0 \leq u^{(n)} < 0110\ldots0$
  - $0.75x.10\ldots0=(2^{-1}+2^{-2})2^m=2^{m-1}+2^{m-2}+0x2^{m-3}+\ldots+0x2^0=110\ldots0$
  - Avec une représentation en m bits on obtient:
  - $010\ldots0 \leq l^{(n)} < 10\ldots0 \rightarrow l^{(n)} = \mathbf{01}x\ldots x$
  - $10\ldots0 \leq u^{(n)} < 110\ldots0 \rightarrow u^{(n)} = \mathbf{10}x\ldots x$
  - Pour utiliser E_3 il faut juste vérifier les 2 MSBs.

Redimensionnement avec l'arithmétique entière (3)

- Procédure de redimensionnement E_3
  - $E_3(l^{(n)})=2*(l^{(n)}-0.25*2^m)=2*(l^{(n)}-2^{m-2})$
  - $E_3(l^{(n)})=2*(\mathbf{01}x\ldots xx-010\ldots0)=2*(00x\ldots xx)=0x\ldots xx0$

NB: Multiplication par 2=décalage vers la gauche part ajout de 0.

  - $E_3(u^{(n)})=2*(u^{(n)}-0.25*2^m)+1=2*(u^{(n)}-2^{m-2})+1$
  - $E_3(u^{(n)})=2*(\mathbf{10}x\ldots xx-010\ldots0)+1=2*(01x\ldots xx)+1=$ décalage vers la gauche part ajout de 1= 1x…xx1.

E_3 : (1) Complémenter le second MSB (2) décaler à gauche.

- Exemple m=6: Application de E_3
- $l^{(n)}=23=010111=\mathbf{01}x\ldots x$ et $u^{(n)}=46=101110=\mathbf{10}x\ldots x$
- $l^{(n)}$:010111 (1)→ 000111 (2)→ 001110=14=2(23-0.25x64)
- $u^{(n)}$:101110 (1)→ 111110 (2)→ 111101=61=2(46-0.25x64)+1

Implémentation entière de l'encodeur [Algorithme]

```
l=00…0, u=11…1, e3_count=0
repeat
  x=get_symbol
  l=l+ ⌊(u-l+1)xCC(x-1)/TC⌋          // lower bound update
  u=l+ ⌊(u-l+1)xCC(x)/TC⌋-1          // upper bound update
  while(MSB(u)==MSB(l) OR E3(u,l)) // MSB(u)=MSB(l)=0 ➔ E1 rescaling
    if(MSB(u)==MSB(l))             // MSB(u)=MSB(l)=1 ➔ E2 rescaling
      send(MSB(u))
      l = (l<<1)+0                 // shift left, set LSB to 0
      u = (u<<1)+1                 // shift left, set LSB to 1
      while(e3_count>0)
        send(!MSB(u))              // encode accumulated E3 rescalings
        e3_count--
      endwhile
    endif
    if(E3(u,l))                    // perform E3 rescaling & remember
      l = (l<<1)+0
      u = (u<<1)+1
      complement MSB(u) and MSB(l)
      e3_count++
    endif
  endwhile
until done
```

Exemple: Implémentation entière de l'encodeur (1)

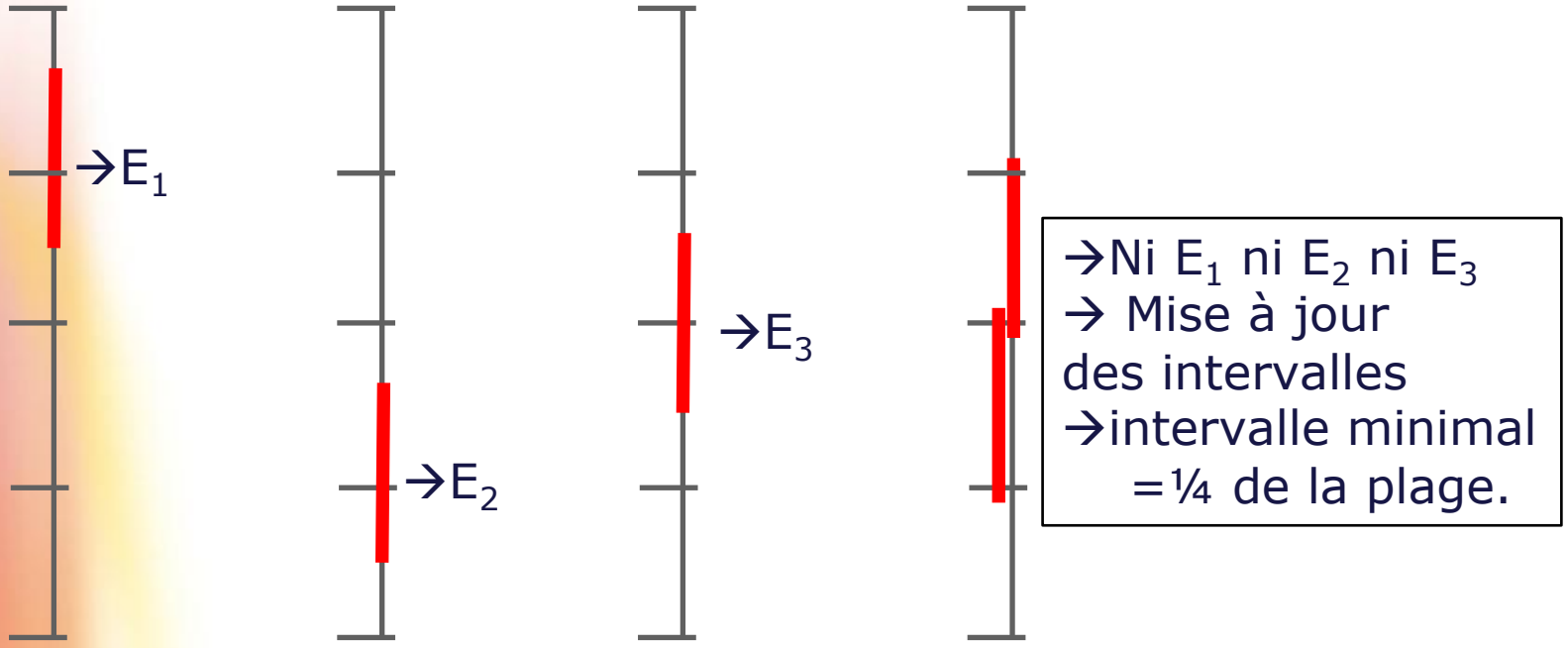
- Encodage de la séquence: **1 3 2 1**

Count(1)=40 Count(2)=1 Count(3)=9 Total_count=50	Cumul(0)=0 Cumul(1)=40 Cumul(2)=41 Cumul(3)=50	Scale3=0

- Choix de la longueur m:
- Cumul(1) et cumul(2) diffère seulement de 1.
- Il faut un m assurant une résolution de 1 pour distinguer entre tous les sous-intervalles.
- Cependant après la mise à jour l'intervalle peut être divisé par 4→ il faut alors une résolution de 1/4.
- Nombre de pts= plage/résolution=50x4=200pts→m=8

Exemple: Implémentation entière de l'encodeur (2)

- Longueur minimal d'un intervalle
  - Nombre de pts= plage/résolution=50x4=200pts→m=8
  - $2^m=256>200$



Exemple: Implémentation entière de l'encodeur (3)

- Encodage de la séquence: **1** 3 2 1 [m=8] [Scale3=0]
  - Initialisation: $l^{(0)}=(00000000)_2$ et $u^{(0)}=(11111111)_2$
  - Mise à jour de l'intervalle du tag:

$$u^{(1)} = l^{(0)} + \left\lfloor (u^{(0)} - l^{(0)} + 1) \times \frac{cumul\_count(x_1)}{total\_count} \right\rfloor - 1$$

$$u^{(1)} = 0 + \left\lfloor (255 - 0 + 1) \times \frac{40}{50} \right\rfloor - 1 = 204 - 1 = 203 = (11001011)_2$$

$$l^{(1)} = l^{(0)} + \left\lfloor (u^{(0)} - l^{(0)} + 1) \times \frac{cumul\_count(x_1 - 1)}{total\_count} \right\rfloor$$

$$l^{(1)} = 0 + \left\lfloor (255 - 0 + 1) \times \frac{0}{50} \right\rfloor = 0 = (00000000)_2$$

  - → Pas de redimensionnement.

[Envoi=]

Exemple: Implémentation entière de l'encodeur (4)

- Encodage de la séquence: 1 **3** 2 1 [m=8] [Scale3=0]
  - Mise à jour de l'intervalle du tag:

$$u^{(2)} = 0 + \left\lfloor (203 - 0 + 1) \times \frac{50}{50} \right\rfloor - 1 = 203 = (11001011)_2$$

$$l^{(2)} = 0 + \left\lfloor (203 - 0 + 1) \times \frac{41}{50} \right\rfloor = 167 = (10100111)_2$$

  - Redimensionnement E_2: [167, 203) inclus dans la moitié supérieure [ou encore: MSB de $l^{(2)}$ et $u^{(2)}$ est le même et égal à 1→ E_2 → Comme scale3=0 envoi du MSB=**1**]
  - $l^{(2)}$ = 167 =10100111 et $u^{(2)}$ =203=11001011
  - $l^{(2)}$ = ~~1~~01001110 = 78 et $u^{(2)}$=~~1~~10010111 = 151

[Envoi=**1**]

Exemple: Implémentation entière de l'encodeur (5)

- Encodage de la séquence: 1 **3** 2 1 [m=8] [Scale3=**1**]
  - $l^{(2)}$ = 01001110 et $u^{(2)}$=10010111: 2 MSBs→ E_3
  - Redimensionnement E_3:
  - $l^{(2)}$ = 01001110 (1)→ 00001110 (2)→ 00011100 = 28
  - $u^{(2)}$ =10010111 (1)→ 11010111 (2)→ 10101111 = 175
  - Scale3+; [$l^{(2)}$,$u^{(2)}$)=[28,175)→Aucun redimensionnement.
- Encodage de la séquence: 1 3 **2** 1 [m=8] [Scale3=1]
  - Mise à jour de l'intervalle du tag:

$$u^{(3)} = 28 + \left\lfloor (175 - 28 + 1) \times \frac{41}{50} \right\rfloor - 1 = 28 + \left\lfloor 148 \times \frac{41}{50} \right\rfloor - 1 = 148 = (10010100)_2$$

$$l^{(3)} = 28 + \left\lfloor (175 - 28 + 1) \times \frac{40}{50} \right\rfloor = 28 + \left\lfloor 148 \times \frac{40}{50} \right\rfloor = 146 = (10010010)_2$$

[Envoi=1]

Exemple: Implémentation entière de l'encodeur (6)

- Encodage de la séquence: 1 3 **2** 1 [m=8] [Scale3=1]
  - $l^{(3)}$ = 10010010 et $u^{(3)}$=10010100
  - Le même MSB=1→ E_2 → envoi MSB=1 [Envoi=1 **1**]
  - Redimensionnement E_2:
  - $l^{(3)}$ = 10010010 → ~~1~~00100100 = 36
  - $u^{(3)}$ = 10010100 → ~~1~~00101001 = 41
  - Comme Scale3=1 (1)→ On envoie !MSB=!1=**0** [Envoi=1 1 **0**]
    (2)→ décrémentation de Scale3=0 [Scale3=0]
  - Redimensionnement E_1: car le même MSB=0
  - $l^{(3)}$ = 00100100 → ~~0~~01001000 = 72
  - $u^{(3)}$ = 00101001 → ~~0~~01010011 = 83
  - Comme Scale3=0→ envoi MSB=**0** [Envoi=1 1 0 **0**]

Exemple: Implémentation entière de l'encodeur (7)

- Encodage de la séquence: 1 3 **2** 1 [m=8] [Scale3=0]
  - $l^{(3)}$ = 01001000 et $u^{(3)}$ = 01010011
  - Redimensionnement E_1: car le même MSB=0
  - $l^{(3)}$ = 01001000 → ~~0~~10010000 = 144
  - $u^{(3)}$ = 01010011 → ~~0~~10100111 = 167
  - Comme Scale3=0→ envoi MSB=**0**
  - Redimensionnement E_2: car le même MSB=1
  - $l^{(3)}$ = 10010000 → ~~1~~00100000 = 32
  - $u^{(3)}$ = 10100111 → ~~1~~01001111 = 79
  - Comme Scale3=0→ envoi MSB=**1**

[Envoi=1 1 0 0 **0**]

[Envoi=1 1 0 0 0 **1**]

Exemple: Implémentation entière de l'encodeur (8)

- Encodage de la séquence: 1 3 **2** 1 [m=8] [Scale3=**1**]
  - $l^{(3)}$ = 00100000 et $u^{(3)}$ = 01001111
  - Redimensionnement E_1: car le même MSB=0
  - $l^{(3)}$ = 00100000 → ~~0~~01000000 = 64
  - $u^{(3)}$ = 01001111 → ~~0~~10011111 = 159
  - Comme Scale3=0→ envoi MSB=**0** [Envoi=1 0 0 0 1 **0**]
  - Redimensionnement E_3: Voir les 2 premiers MSBs
  - $l^{(3)}$ = 01000000 (1)→ 00000000 (2)→~~0~~00000000 = 0
  - $u^{(3)}$ = 10011111 (1)→ 11011111 (2)→~~1~~10111111 = 191
  - Scale3++
  - Aucun redimensionnement [intervalle> ¼x256=64]

[Envoi=1 1 0 0 0 1 **0**]

Exemple: Implémentation entière de l'encodeur (9)

- Encodage de la séquence: 1 3 2 **1** [m=8] [Scale3=1]
  - $l^{(3)}$ = 00000000=0 et $u^{(3)}$ = 10111111=191

$$u^{(4)} = 0 + \left\lfloor (191 - 0 + 1) \times \frac{40}{50} \right\rfloor - 1 = \left\lfloor 192 \times \frac{40}{50} \right\rfloor - 1 = 152 = (10011000)_2$$

$$l^{(4)} = 0 + \left\lfloor (191 - 0 + 1) \times \frac{0}{50} \right\rfloor = 0 = (00000000)_2$$

- Fin de la séquence:
  - Séquence envoyée= 1 1 0 0 0 1 0
  - Envoi du tag $\in [l^{(4)}, u^{(4)})$ → choix: tag= $l^{(4)}$ = 00000000
  - Sauf que Scale3=1: on envoie MSB=0 de $l^{(4)}$ puis on envoie !MSB=1 enfin on continue avec le restant des 0s.

[Envoi=1 1 0 0 0 1 0 **0 1 0 0 0 0 0 0 0**]

Implémentation entière du décodeur [Algorithme]

```
Initialize l, u, t                          // t = first m bits
repeat
  k=0
  while( ⌊((u-l+1)×TC-1)/(u-l+1)⌋ ≥ CC(k)
    k++
  x = decode_symbol(k)
  l=l+ ⌊(u-l+1)×CC(x-1)/TC⌋
  u=l+ ⌊(u-l+1)×CC(x)/TC⌋-1
  while(MSB(u)==MSB(l) OR E3(u,l))
    if(MSB(u)==MSB(l))                      // Perform E1/E2 rescaling of l,u,t
      l = (l<<1)+0
      u = (u<<1)+1
      t = (u<<1)+next_bit
    endif
    if(E3(u,l))                             // Perform E3 rescaling of l,u,t
      l = (l<<1)+0
      u = (u<<1)+1
      t = (u<<1)+next_bit
      complement MSB(u), MSB(l), MSB(t)
    endif
  endwhile
until done
```

Exemple: Implémentation entière décodeur (1)

- Décodage de la séquence: **1 1 0 0 0 1 0 0** 1 0 0 0 0 0 0 0
  - On regroupe les m=8 premiers bits pour former le tag.
  - t=11000100=196
  - On initialise : l=00000000=0 et u=11111111=255

$$\text{On calcule:}\begin{cases}\left\lfloor \dfrac{(t-l+1)\times total\_count-1}{u-l+1}\right\rfloor=\left\lfloor \dfrac{197\times 50-1}{256}\right\rfloor=38\\ cumul(0)\le 38<cumul(1)\Rightarrow 1\end{cases}$$

$$\text{Mise à jour:}\begin{cases}u=0+\left\lfloor (255-0+1)\times\dfrac{40}{50}\right\rfloor-1=203=(11001011)_2\\ l=0+\left\lfloor (255-0+1)\times\dfrac{0}{50}\right\rfloor=0=(00000000)_2\end{cases}$$

Cumul(0)=0
Cumul(1)=40
Cumul(2)=41
Cumul(3)=50

➢ Séquence décodée: **1**

Exemple: Implémentation entière décodeur (2)

- Décodage de la séquence: **1 1 0 0 0 1 0 0** 1 0 0 0 0 0 0 0
  - l=**00**000000 et u=**11**001011→ aucun mapping

$$\text{On calcule:}\begin{cases}\left\lfloor \dfrac{(t-l+1)\times total\_count-1}{u-l+1}\right\rfloor=\left\lfloor \dfrac{197\times 50-1}{203}\right\rfloor=48\\ cumul(2)\le 48<cumul(3)\Rightarrow 3\end{cases}$$

$$\text{Mise à jour:}\begin{cases}u=0+\left\lfloor (203-0+1)\times\dfrac{50}{50}\right\rfloor-1=203=(11001011)_2\\ l=0+\left\lfloor (203-0+1)\times\dfrac{41}{50}\right\rfloor=167=(1010011)_2\end{cases}$$

Cumul(0)=0
Cumul(1)=40
Cumul(2)=41
Cumul(3)=50

➢Séquence décodée: 1 **3**

Exemple: Implémentation entière décodeur (3)

- Décodage de la séquence: 1 1 0 0 0 1 0 0 **1** 0 0 0 0 0 0 0
  - l=**1**0100011 et u=**1**1001011→ mapping E2(l, u, t)
  - l=0100110 u=10010111 t=10001001
- Décodage de la séquence: 1 1 0 0 0 1 0 0 1 **0** 0 0 0 0 0 0
  - l=**01**000110 et u=**10**010111→ mapping E3(l, u, t)
  - l=00001100=28 u=10101111=175 t=10010010=146

$$\left\{\left\lfloor \frac{(t-l+1)\times total\_count-1}{u-l+1} \right\rfloor = 40 \Rightarrow cumul(1) \le 40 < cumul(2) \Rightarrow 2\right.$$

$$\begin{cases} u = 28 + \left\lfloor (175-28+1)\times \frac{41}{50} \right\rfloor - 1 = 148 = (10010100)_2 \\ l = 28 + \left\lfloor (175-28+1)\times \frac{40}{50} \right\rfloor = 146 = (10010010)_2 \end{cases}$$

➢Séquence décodée: 1 3 **2**

Cumul(0)=0
Cumul(1)=40
Cumul(2)=41
Cumul(3)=50

Exemple: Implémentation entière décodeur (4)

- Décodage de la séquence: 1 1 0 0 0 1 0 0 1 0 **0 0 0 0 0 0**
  - l=10010010=146 u=10010100=148 t=10010010=146
  - On va effectuer E2 E1 E1 E2 E1 sur l, u et t.
  - t=l et les bits restants sont tous des 0s→ t restera=l.
  - E2(E1(E1(E2(E1(l))))))=01000000
  - E2(E1(E1(E2(E1(u))))))=10011111
  - E3: l=00000000=0 → t=l=0
  - E3: u=10111111=191

$$\left\lfloor \frac{(t-l+1)\times total\_count-1}{u-l+1} \right\rfloor = \left\lfloor \frac{49}{192} \right\rfloor = 0 \Rightarrow cumul(0) \le 0 < cumul(1) \Rightarrow 1$$

Cumul(0)=0
Cumul(1)=40
Cumul(2)=41
Cumul(3)=50

➢Séquence décodée: 1 3 2 **1**

Comparaison codage arithmétique vs codage de Huffman (1)

- Longueur moyenne:
  - Codage arithmétique $H(X) \leq l_A < H(X) + \frac{2}{m}$
  - Codage de Huffman étendu avec un groupement de m symboles: $H(X) \leq l_H < H(X) + \frac{1}{m}$
  - Huffman étendu: construction de k^m mots-code.
  - Exemple pour un alphabet de k=16 élément et un m=20 → construction de 16^{20} mots-code.
  - Pour le code arithmétique il n'est pas nécessaire de construire tout les mots-code→ m peut prendre des valeurs très élevés dépassant 20.
  - On peut s'approcher de l'entropie peu importe la source.

Comparaison codage arithmétique vs codage de Huffman (2)

- Longueur moyenne:
  - Si l'alphabet est relativement large et les probabilités sont réparties le code de Huffman s'approche encore plus de l'entropie:

$$H(X) \leq l_H < H(X) + 0.086 + p_{\max}$$

  - Si les probabilités sont des puissances négatives de 2 le code de Huffman sans regroupement est optimal et le code arithmétique ne peut pas faire mieux même si m est élevée.
  - C'est plus facile d'adapter le code arithmétique au changement des statistiques en comptant l'occurrence des éléments de l'alphabet.

Codage arithmétique adaptatif

- Séparation de la modélisation et du codage:
  - Pas d'information disponible à priori sur la source.
  - On initialise par un compteur=1 pour chaque symbole.
  - Après chaque encodage d'une lettre son compteur est incrémenté.
  - Au décodeur le compteur est mis à jour après chaque décodage.
  - La taille des mots-code dépend de total_count.
  - Or total_count varie lors de la lecture des symboles.
  - On a vu que m est choisi tel que 4 x total_count$\leq 2^m$.
  - Donc si total_cout > 2^{m-2}. on décrémente les plus anciennes observations.