Département d'informatique et de génie logiciel

Compression de données
IFT-4003/IFT-7023

Notes de cours
Codage de Huffman

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

- Codage de Huffman:
  - Codage de Shannon-Fano
  - Procédure de construction des codes de Huffman
  - Extension des codes de Huffman
  - Code de Huffman non binaires
  - Codes de Huffman adaptatif
  - Codes de Golomb

Codage de Shannon-Fano

- Construction d'un code préfixe basé sur la théorie de Shannon.
- Code développé en 1960 par Claude E. Shannon (MIT) et Robert M. Fano (Laboratoires de Bell).
- Assignation du code selon la probabilité de chaque symbole.
- Algorithme simple avec des performances élevées.
- Cependant c'est un code sous-optimal en terme de longueur moyenne des mot code.
- A partir de ce code un étudiant gradué a développé un autre code assurant l'optimalité: David A. Huffman.
- Exemple d'utilisation: format ZIP.

Algorithme de Shannon-Fano

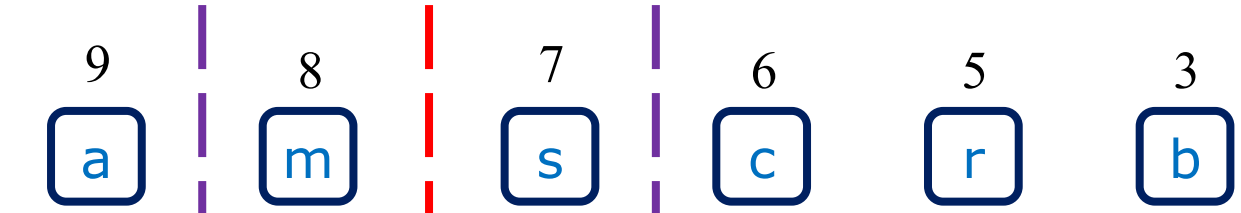
1. Détermination des probabilités de chacun des symboles soit par mesure ou soit par estimation.
2. Ordonner les symboles selon leurs probabilité d'apparence croissant ou décroissant.
3. Diviser l'ensemble des symboles en deux sous-groupes ayant une différence de probabilité minimale.
4. Assigner un '0' pour le premier sous-groupe et un '1' pour le second sous-groupes.
5. Réitérer à la 3ème étape en subdivisant les sous-groupes.
6. Condition d'arrêt: tous sous-groupes sont formés d'un singleton.

Exemple: codage de Shannon-Fano (1)

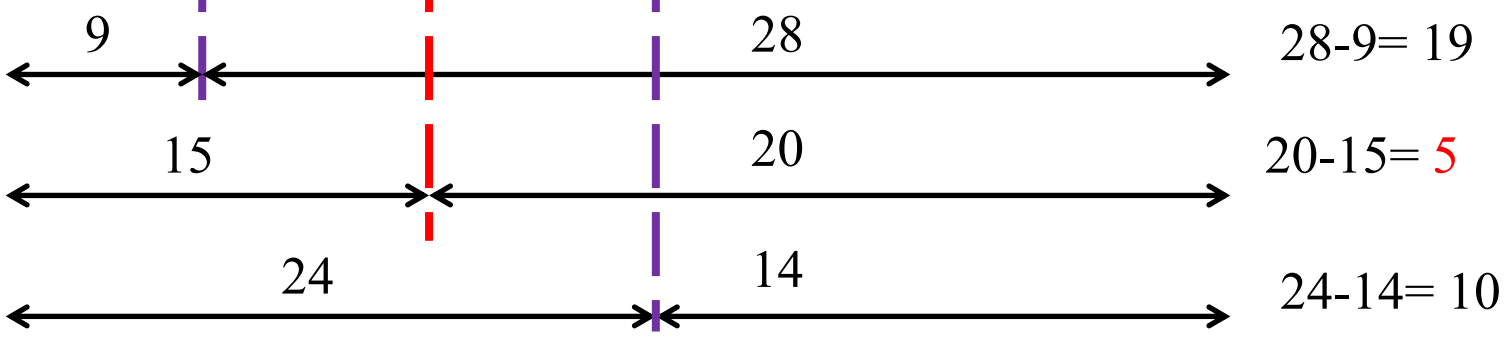
Étape 1:
calcul des fréquences

Lettres	a	b	c	m	r	s
Fréquence	9	3	6	8	5	7

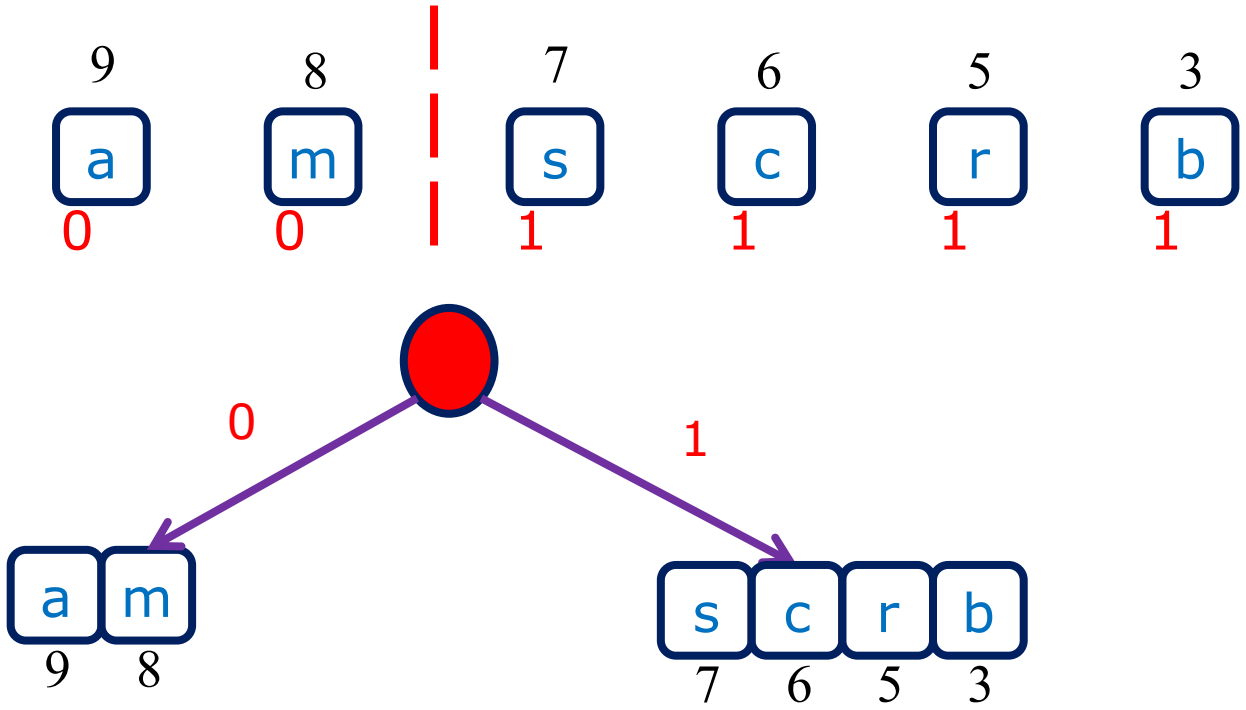
Étape 2:
ordonner les fréquences



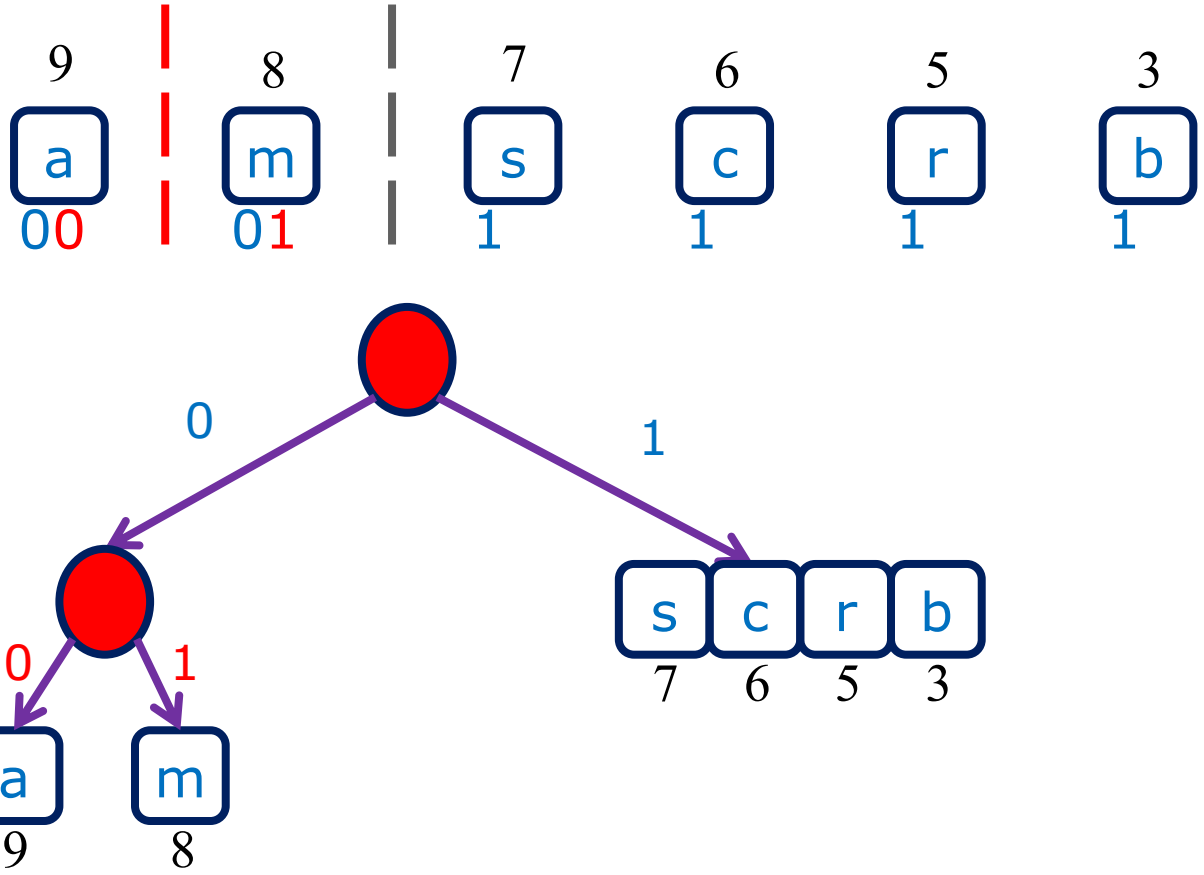
Étape 3:
Division en groupes de fréquences rapprochées



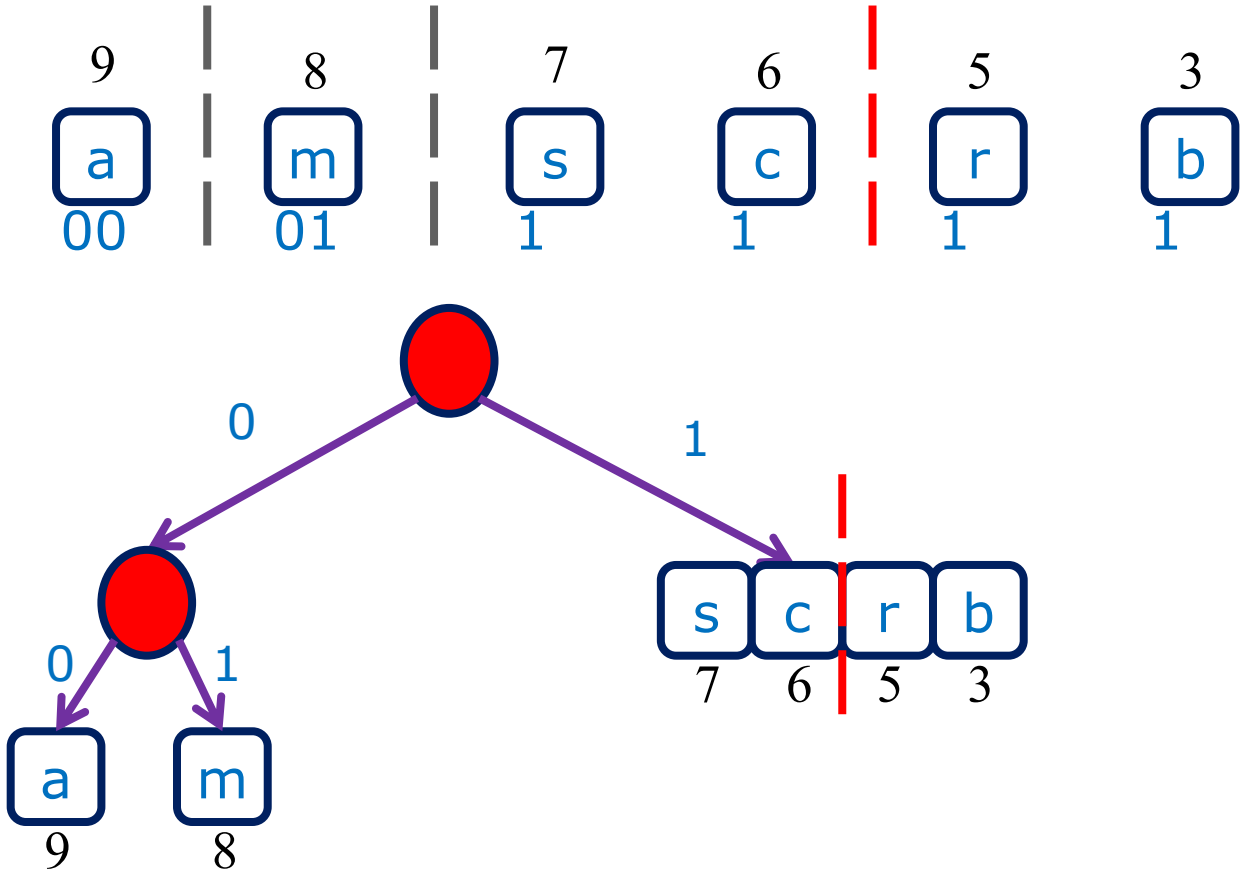
Exemple: codage de Shannon-Fano (2)



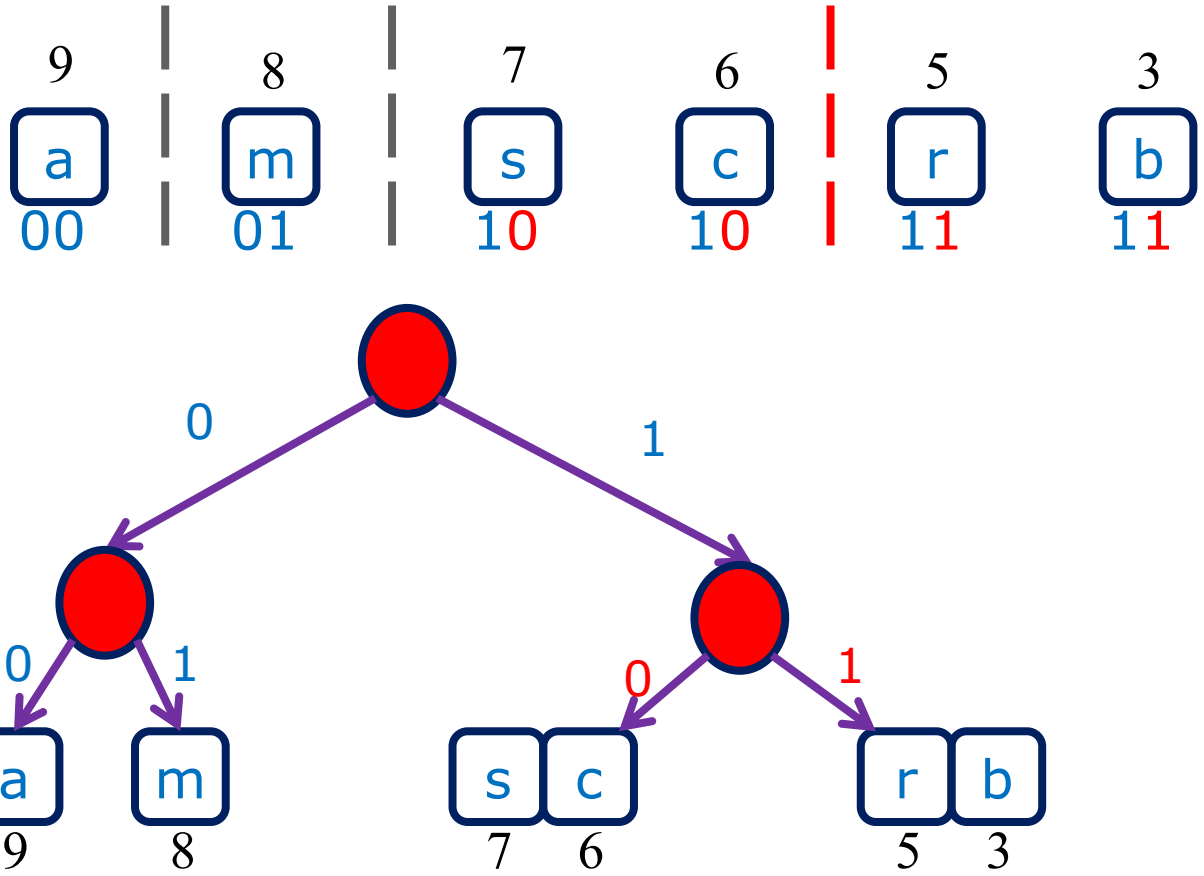
Exemple: codage de Shannon-Fano (3)



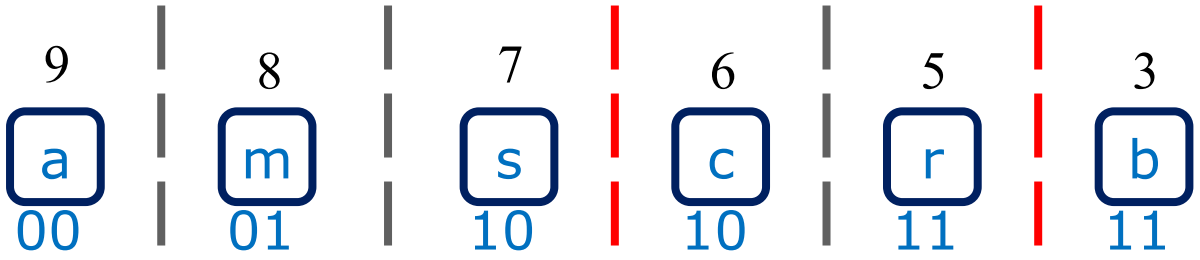
Exemple: codage de Shannon-Fano (4)

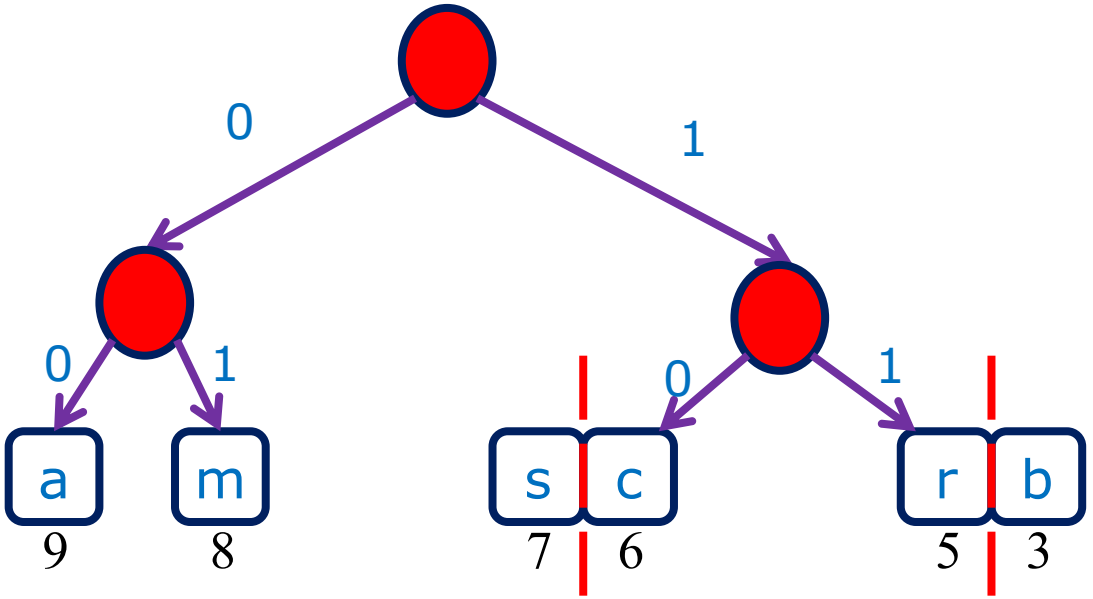


Exemple: codage de Shannon-Fano (5)

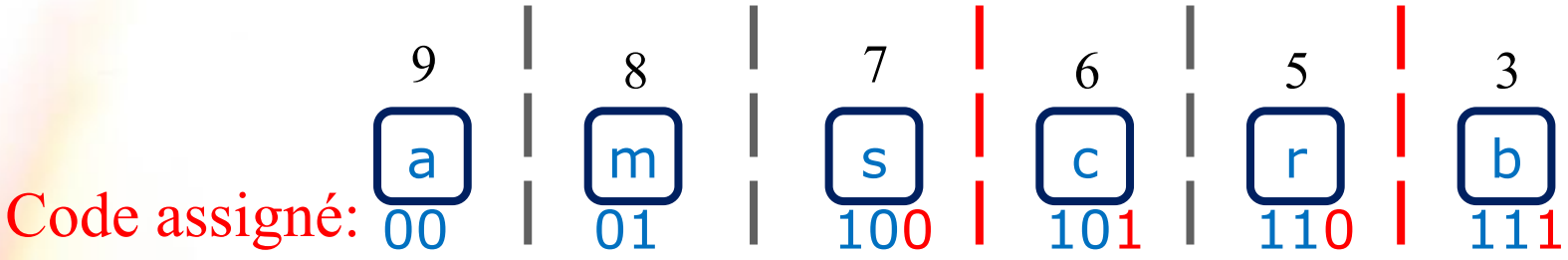


Exemple: codage de Shannon-Fano (6)

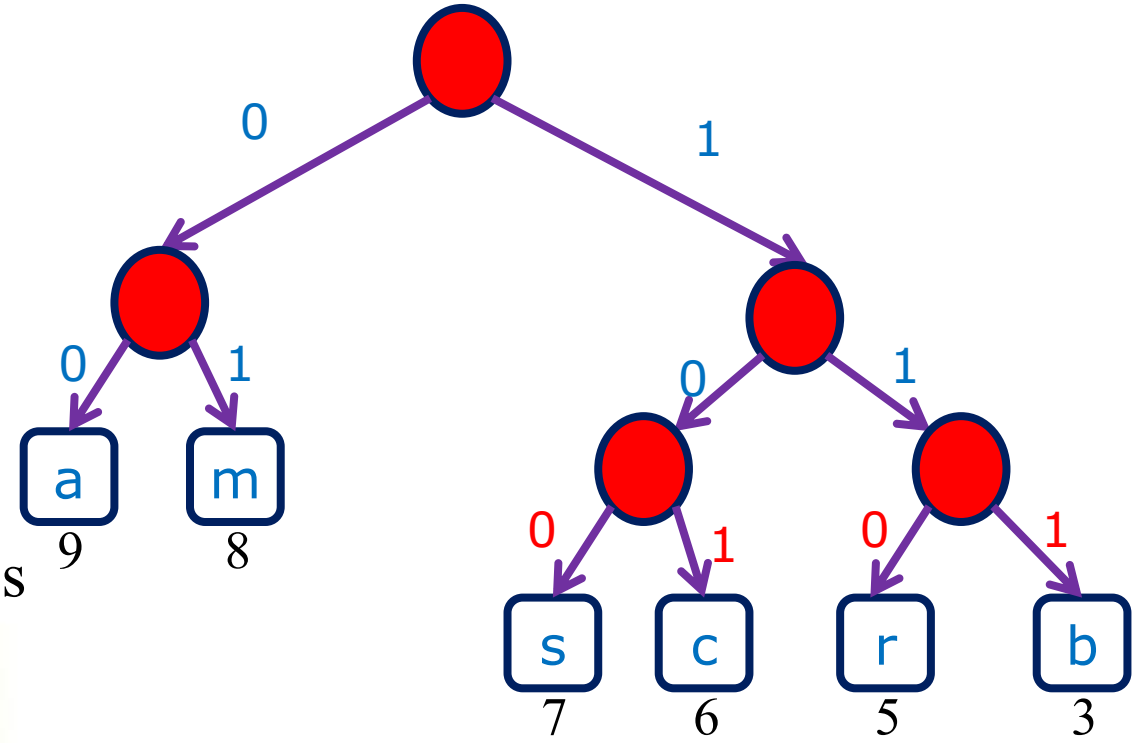




Exemple: codage de Shannon-Fano (7)



Condition d'arrêt:
Sous groupes formés par des singletons



Remarques: codage de Shannon-Fano

Lettres	a	b	c	m	r	s
Fréquence	9	3	6	8	5	7
Code	00	111	101	01	110	100
Probabilité	0.23684	0.078947	0.15789	0.21053	0.13158	0.18421

Longueur moyenne du code: 2.5526

Entropie de la source: 2.5096

Code de Huffman

- Le code de Shannon-Fano ne permette pas d'obtenir un code optimale.
- Le code de Huffman est code presque aussi simple que le code de Shannon-Fano.
- Le code permet d'avoir un code à préfixe aussi.
- Le code de Huffman est optimal et il est basé sur deux observation:
  - Dans un code optimal, on assigne moins de bits aux symboles les plus fréquents et plus de bits au symboles les moins fréquents.
  - Dans un code optimal, les deux moins fréquents symboles ont la même longueur.

Longueur du code de Huffman (1)

- La longueur moyenne de se code peut atteindre l'entropie de la source (code optimal). Cependant il faut que chaque mot code puisse être représenté par un nombre entier de bits.
- Ceci étant possible si les probabilités des symboles sont des puissances négatives de 2, i.e.: 2^{-1}, 2^{-2} ..
- Longueur moyenne d'un code de Huffman:

$$H(S) \leq l_{moy} < H(S) + 1$$

- Pour démontrer ce point on utilise l'inégalité de ***Kraft-McMillan***. Soit C un code uniquement décodable formé par K mots code de longueurs $l_{i=1..K}$ alors:

$$\sum_{i=1}^{K} 2^{-l_i} \leq 1$$

Longueur du code de Huffman (2)

- Soit un code 1 qui est uniquement décodable et un code 2 qui ne l'est pas. Vérifions l'inégalité de Kraft-McMillan:

Lettres	a	b	c	m	r	s
Code 1	00	111	101	01	110	100
Code 2	00	11	10	01	110	100

- Code 1: $\sum_{i=1}^{K} 2^{-l_i} = 2^{-2} + 2^{-3} + 2^{-3} + 2^{-2} + 2^{-3} + 2^{-3} = 1 \leq 1$
- Code 2: $\sum_{i=1}^{K} 2^{-l_i} = 2^{-2} + 2^{-2} + 2^{-2} + 2^{-2} + 2^{-3} + 2^{-3} = 1.25 > 1$

Longueur du code de Huffman (3)

- Nous avons dit que la longueur moyenne d'un code de Huffman:

$$H(S) \leq l_{moy} < H(S) + 1$$

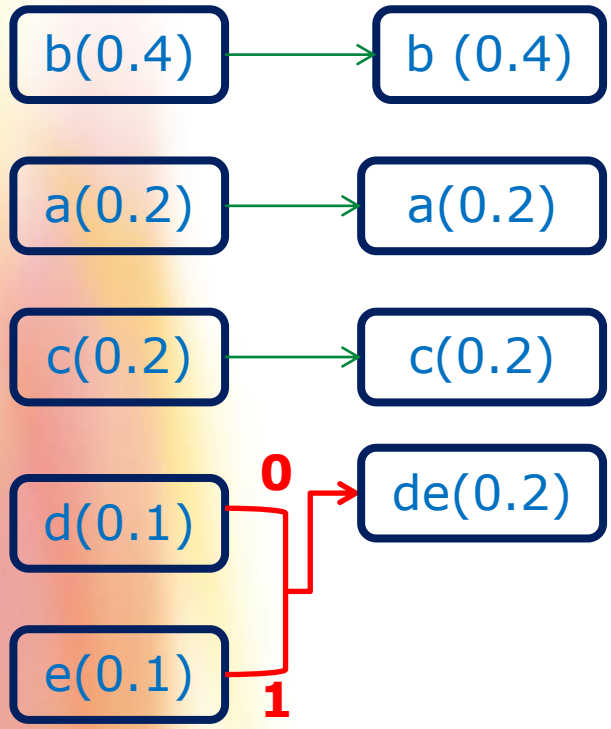
- La bande supérieure est en fait un peu lousse. En effet soit une liste de symboles tel que la probabilité maximale d'apparition d'un d'eux est p_{max}. Alors on a:

$$H(S) \leq l_{moy} < H(S) + p_{\max}, \quad p_{\max} \geq 0.5$$

$$H(S) \leq l_{moy} < H(S) + p_{\max} + 0.086, \quad p_{\max} < 0.5$$

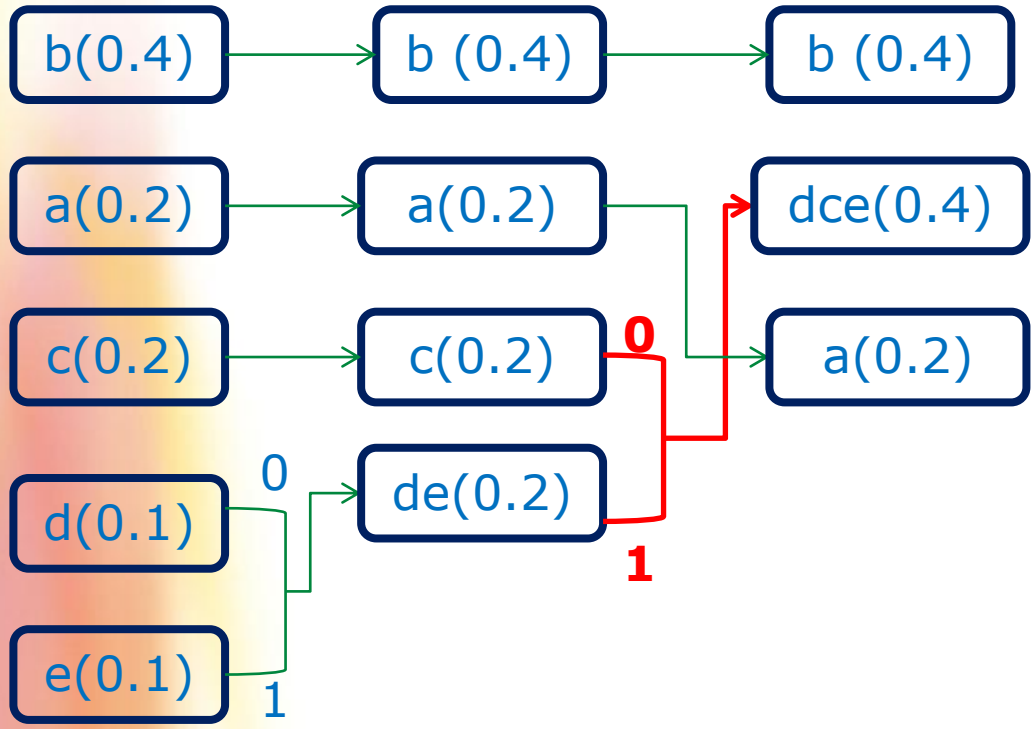
Génération d'un code de Huffman (1)

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code				**0**	**1**



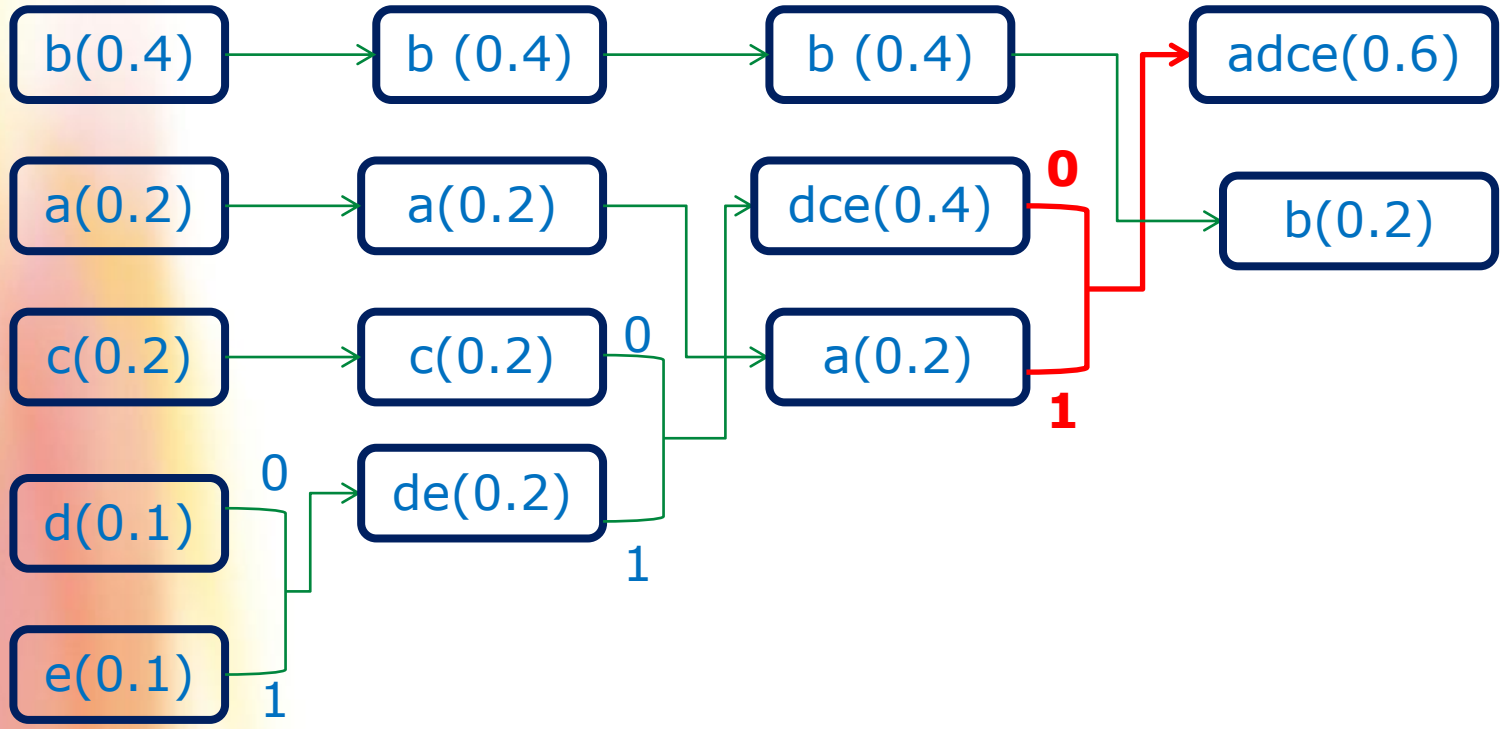
Génération d'un code de Huffman (2)

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code			**0**	**1**0	**1**1



Génération d'un code de Huffman (3)

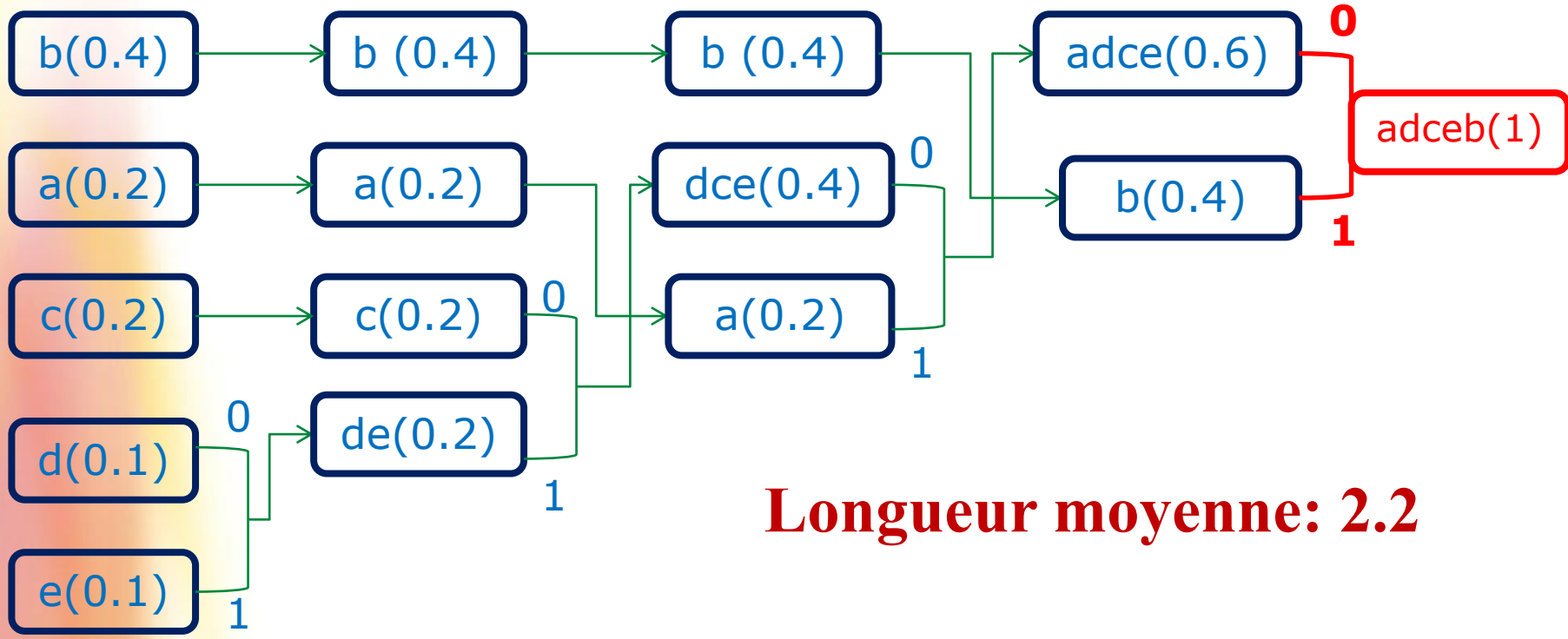
Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	**1**		**0**0	**0**10	**0**11



Mohamed Haj Taieb, Département de génie électrique et de génie informatique

Génération d'un code de Huffman (4)

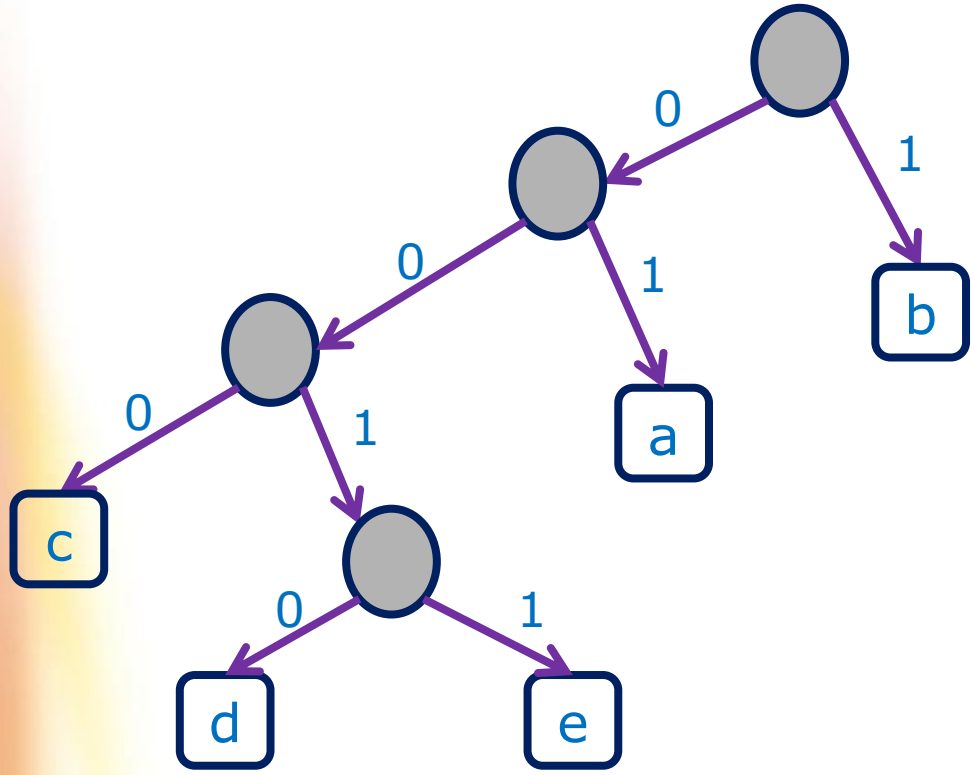
Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	**0**1	**1**	**0**00	**0**010	**0**011



Longueur moyenne: 2.2

Génération d'un code de Huffman avec un arbre

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	01	1	000	0010	0011



Remarques: Code Huffman (1)

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	01	1	000	0010	0011
Longueur	2	1	3	4	4

- Longueur moyenne du code: l=2.2 bits/symbole
- Entropie de la source: H= 2.1219 bits /symbole
- Redondance du code: l-H= 0.078072 bits /symbole
- Efficacité du code: $$\xi = \frac{H\ \ X}{lmoy} = \frac{2.1219}{2.2} = 96{,}45\%$$
- Redondance>0 car les probabilités ne sont pas des puissances négative de 2.
- Variance de la longueur du code: 1.36

Remarques: Code Huffman (2)

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	01	1	011	0110	0111
Longueur	2	1	3	4	4

- Longueur moyenne du code: l=2.2 bits/symbole
- Entropie de la source: H= 2.1219 bits /symbole

$$H(X) \le l_{moy} < H(X) + p_{\max} + 0.087$$
$$2.1219 \le 2.2 < 2.1219 + 0.4 + 0.087$$
$$2.1219 \le 2.2 < 2.6089$$

- Kraft-McMillan: $\sum_{i=1}^{K} 2^{-l_i} = 2^{-2} + 2^{-2} + 2^{-3} + 2^{-4} + 2^{-4} = 1 \le 1$

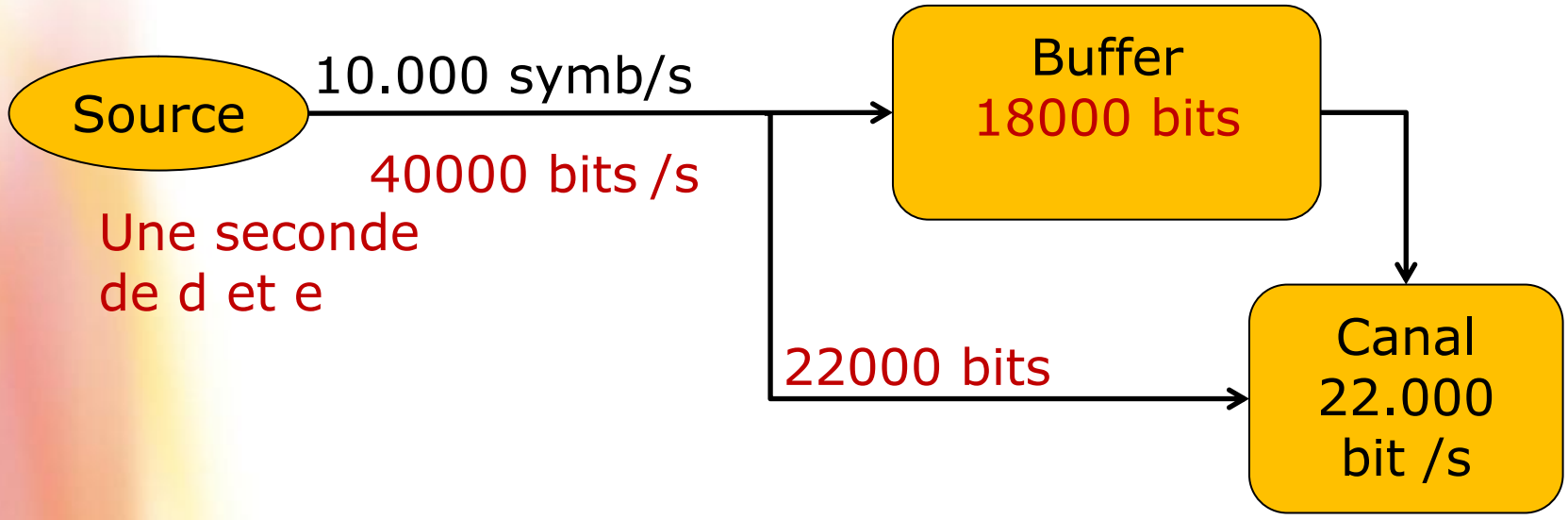
Mohamed Haj Taieb, Département de génie électrique et de génie informatique

Code de Huffman à variance minimale

- Dans la plupart des applications de transfert de données, le taux de transmission est fixe même si on utilise des codes à longueur variable.
- Pour gérer l'envoi d'une séquence à longueur variable avec un taux, on utilise une mémoire tampon pour amortir la variation du débit.
- La variabilité de la longueur du code complique le design de la mémoire tampon. Il faut alors minimiser cette variabilité.
- **Exemple:**
  - Soit une source qui génère une source à un rythme de 10000 symboles/seconde
  - Soit un canal avec un taux de transmission fixe 22000 bits /seconde.

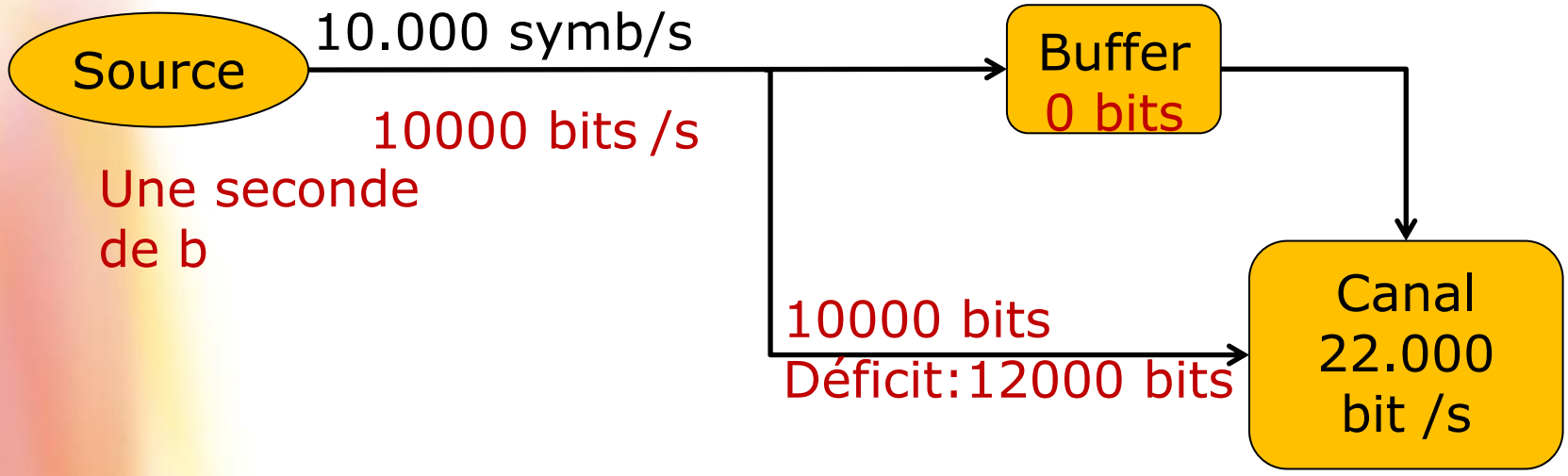
Inconvénient de la variance élevée

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	01	1	011	0110	0111
Longueur	2	1	3	4	4



Inconvénient de la variance élevée

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	01	1	011	0110	0111
Longueur	2	1	3	4	4

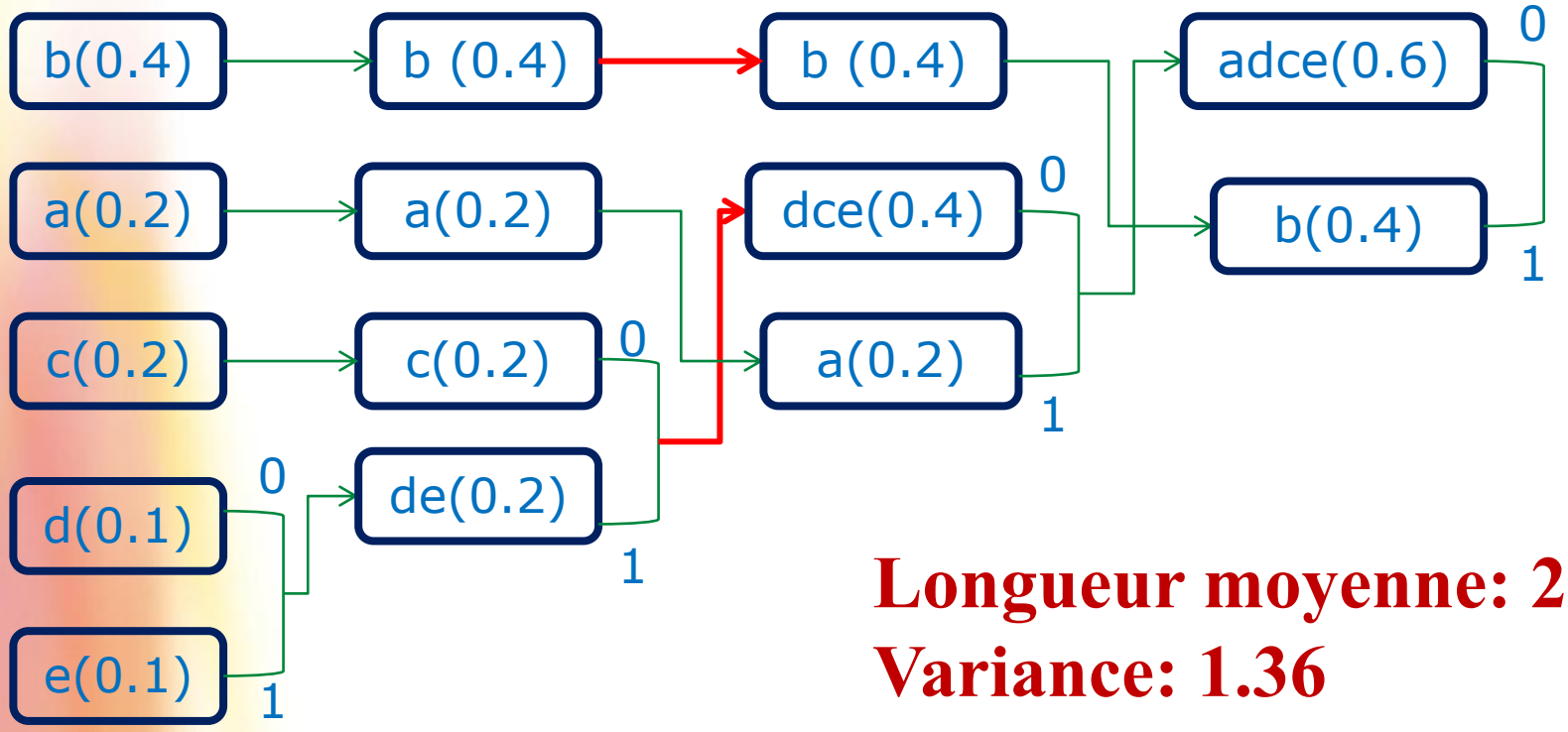


Génération du code de Huffman à variance minimale

- Lorsque la variance de la longueur du code est élevée la mémoire tampon à l'encodage doit être large: 18000 bits.
- Le déficit de transmission peut atteindre 12000 bits.
- I faut concevoir un code de Huffman tout en minimisant la variance.
- C'est quasiment la même procédure de codage.
- Lorsqu'il y a une égalité dans les probabilités placer l'ensemble formé par le plus de lettre en haut.
- Dans l'exemple qu'on a présenté avant la variance est de 1.36. On va construire un code de Huffman à variance minimale de 0.24.

Génération d'un code de Huffman

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	01	1	011	0110	0111

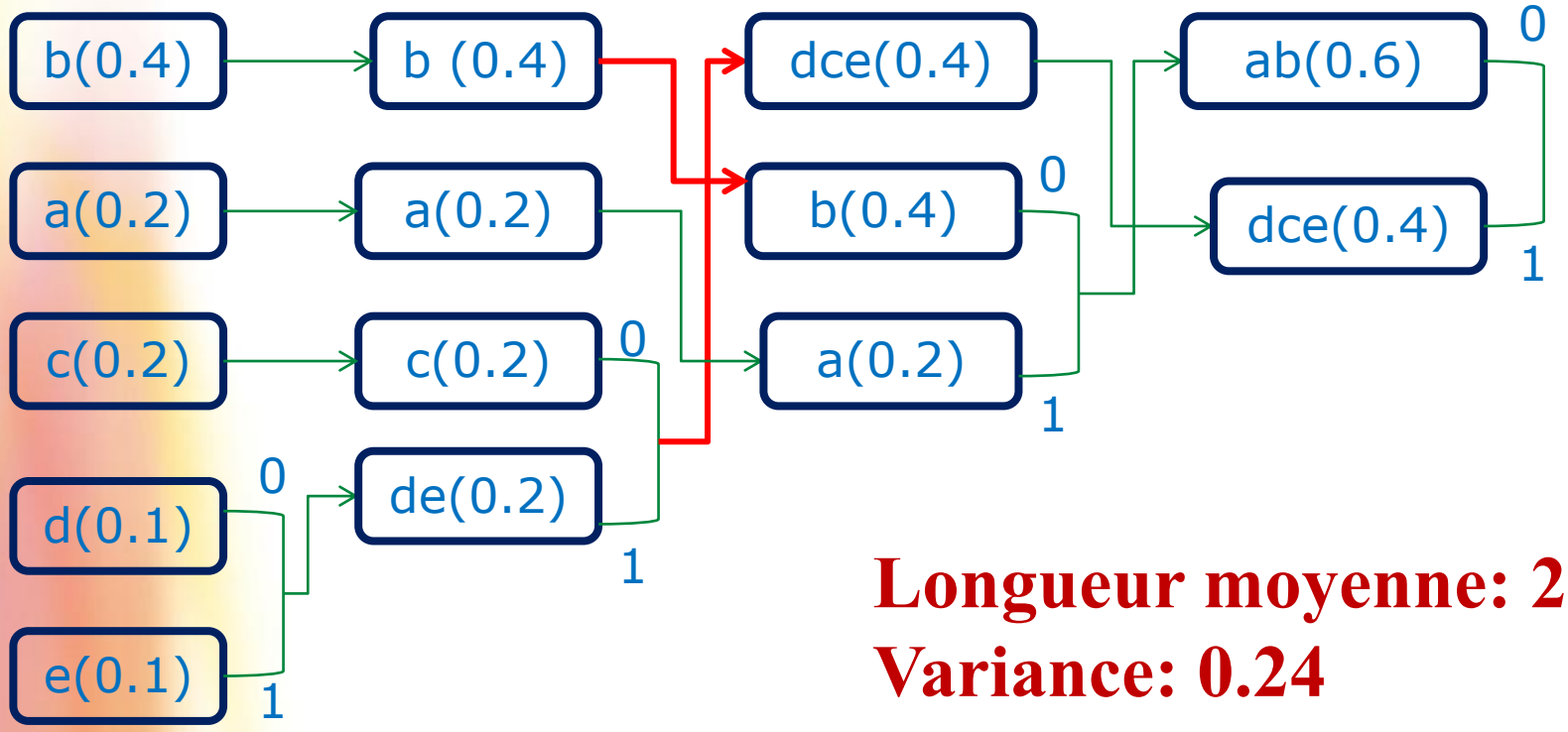


Longueur moyenne: 2.2

Variance: 1.36

Génération d'un code de Huffman à variance minimale

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	01	00	10	110	111

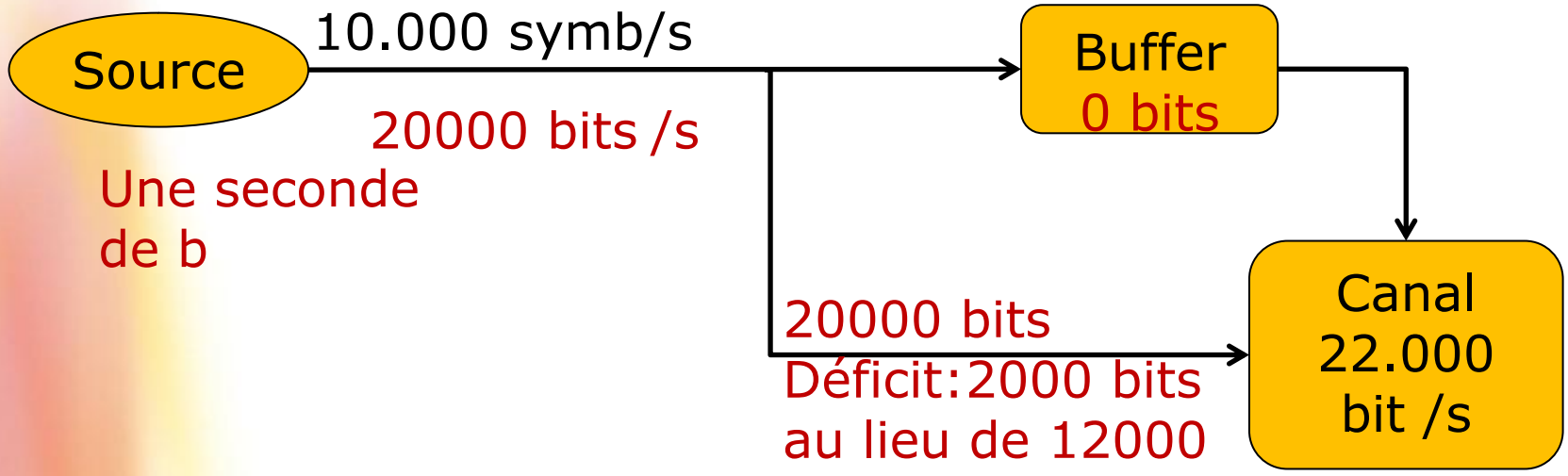


Longueur moyenne: 2.2

Variance: 0.24

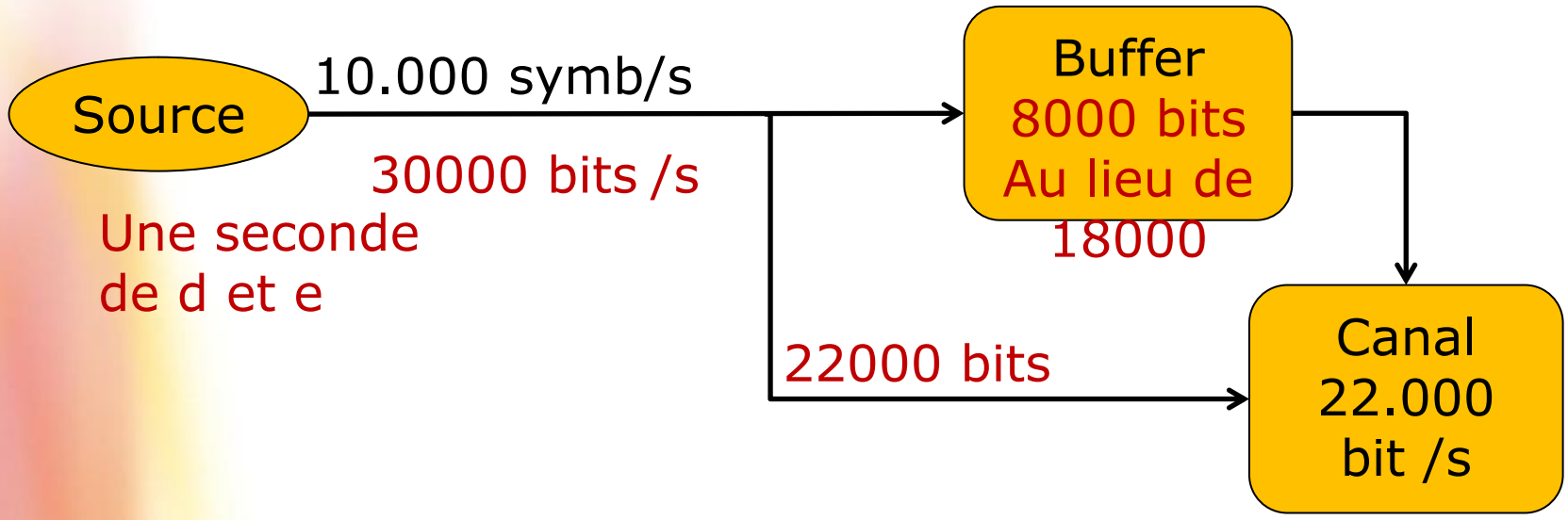
Minimisation de la variance

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	01	00	10	110	111



Minimisation de la variance

Lettres	a	b	c	d	e
Probabilité	0.2	0.4	0.2	0.1	0.1
Code	01	00	10	110	111



Extension des codes de Huffman

- Dans les application où la taille de l'alphabet est large, la valeur de p_{max} est relativement faible. Ainsi le code est performant vu que sa longueur moyenne est proche de l'entropie:

$$H(X) \leq l_{moy} < H(X) + p_{\max} + 0.087$$
$$p_{\max} \ll 1 \Rightarrow l_{moy} \approx H(X)$$

- Cependant pour un alphabet de taille faible et une probabilité de symbole débalancée et biaisée valeur de p_{max} devient élevée et la longueur moyenne risque d'être élevée.
- Dans cette situation l'efficacité du code risque de devenir faible

Inefficacité du code de Huffman

- Soit le code de Huffman de la source suivante:

Lettres	a	b	c
Probabilité	0.8	0.02	0.18
Code	0	11	10

- Entropie de la source: H=0.816 bit/symbole
- l_{moy}=1.2 bit/symbole
- Redondance=1.2-0.816=0.384 bit/symbole=47% de l'entropie
- Le code nécessite 47% plus de bits que le code optimal.
- Efficacité du code

$$\xi = \frac{H}{lmoy} = \frac{0.816}{1.2} = 68\%$$

Regroupement des symboles (1)

- Soit une source S qui émet des éléments indépendants d'un alphabet A={a_1, a_2,..,a_m}. L'entropie de la source est:

$$H(S) = -\sum_{i=1}^{m} P(a_i) \log_2 P(a_i)$$

- Comme indiqué précédemment, on peut générer un code de Huffman avec un débit *R* tel que:

$$H(S) \leq R < H(S) + 1$$

- Ici le débit *R* dénote le nombre de bits/symbole (ou encore la longueur moyenne).
- Les système de communication *R* réfère au débit de transmission exprime en bits/seconde.

Regroupement des symboles (2)

- Si l'on encode la même source en assignant un mot-code à l'ensemble de m symboles.
- $A=\{a_1, a_2,..,a_m\}$ =>
- $A^{(n=2)}=\{a_1a_1, a_1a_2,.., a_1a_m, a_2a_1, a_2a_2,.., a_2a_m, \dots , a_ma_1, a_ma_2,.., a_ma_m\}$
- Pour l'encodage de la source $S^{(n)}$, il faut un débit $R^{(n)}$ tel que:

$$H(S^{(n)}) \leq R^{(n)} < H(S^{(n)}) + 1$$

- Le débit $R^{(n)}$ est le débit nécessaire pour encoder n symboles et R est le débit nécessaire pour encoder un seul symbole.

$$R^{(n)} = nR \quad \Rightarrow \quad H(S^{(n)}) \leq nR < H(S^{(n)}) + 1$$

$$\frac{H(S^{(n)})}{n} \leq R < \frac{H(S^{(n)})}{n} + 1$$

Regroupement des symboles (3)

$$H(S^{(n)}) \leq nR < H(S^{(n)}) + 1$$

$$\frac{H(S^{(n)})}{n} \leq R < \frac{H(S^{(n)})}{n} + \frac{1}{n}$$

Et comme les éléments de la séquence sont générés indépendamment on a: [démonstration comme exercice]

$$H(S^{(n)}) = nH(S)$$

$$\frac{nH(S)}{n} \leq R < \frac{nH(S)}{n} + \frac{1}{n}$$

$$\boxed{H(S) \leq R < H(S) + \frac{1}{n}}$$

Sans regroupement de symboles nous avions:

$$H(S) \leq R < H(S) + 1$$

Regroupement des symboles (4)

- Sans regroupement de symboles nous avions:

$$H(S) \leq R < H(S) + 1$$

- Avec un encodage de la source utilisant des blocks plus longs le débit R s'approche de plus en plus de l'entropie:

$$\boxed{H(S) \leq R < H(S) + \frac{1}{n}}$$

- Le plus que n augmente le plus le code est plus efficace mais plus complexe.

Exemple code de Huffman étendu (1)

- Reconsidérons la source de l'exemple précédent:

Lettres	a	b	c
Probabilité	0.8	0.02	0.18
Code	0	11	10

- H=0.816 bit/symbole
- l_{moy}=1.2 bit/symbole
- Redondance=0.384
- Efficacité du code: $\xi = 68\%$

- Génération de mots-codes formés de deux symboles.
- L'alphabet étendu contient 3^2=9 éléments comportant toutes les combinaisons possibles:

  {aa, ab, ac, ba, bb, ac, ca, cb, cc}

Exemple code de Huffman étendu (2)

- L'encodage de Huffman de la source étendue donne:

Lettres	aa	ab	ac	ba	bb	bc	ca	cb	cc
Proba.	0.64	0.016	0.144	0.016	0.0004	0.0036	0.144	0.00036	0.0324
Code	0	1010 1	11	10100 0	10100 101	10100 11	10	101001 00	1011

- **Code de Huffman**
- H=0.816 bit/symbole
- l_{moy}=1.2 bit/symbole
- Redondance=0.384
- Efficacité du code: $\xi = 68\%$

- **Code de Huffman étendu**
- l_{moy}=1.7228 bit/2symbs
- l_{moy}=1.7228/2=0.8614 bit/symb
- Redondance=0.045
- Efficacité du code:$\xi = 94.73\%$

Inconvénients des codes de Huffman étendus

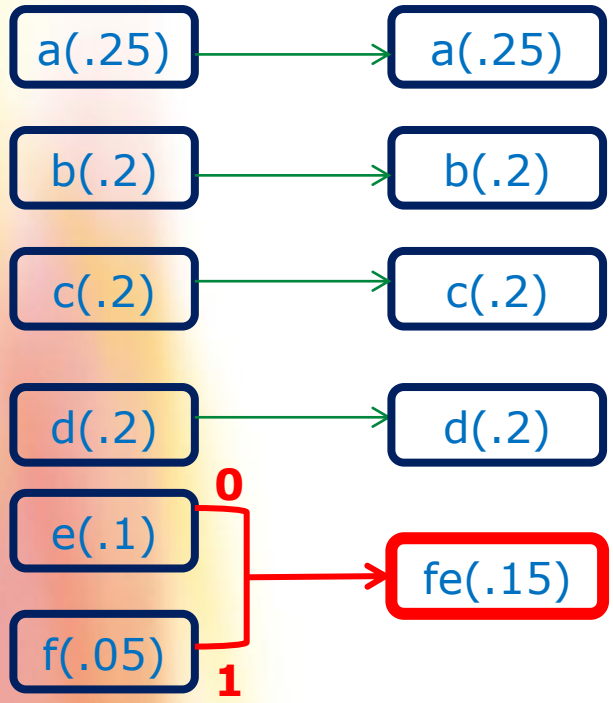
- Dans l'exemple précédent on a vu qu'avec un regroupement de 2 symboles on a pu atteindre une longueur du code très proche de l'entropie.
- Le code de Huffman peut être étendu encore plus et considérer un nouveau alphabet formé par un regroupement de 3, 4, … n symboles donc 3^3, 3^4 , … ,3^n éléments dans l'alphabet.
- Cependant cette croissance exponentielle de la taille de l'alphabet rend le système complexe:
  - Exemple: pour un code d'ASCII de longueur 3 on obtient un alphabet de taille $256^3 = 2^{24} = 16$ Mb
- Dans cette situation, il faut utiliser d'autres méthodes comme le *codage arithmétique*.

Code de Huffman M-aire

- Le codage de Huffman non binaire est basé sur les mêmes principes que le codage binaire avec une petite différence:
  - On assigne moins de bits aux symboles les plus fréquents et plus de bits aux symboles les moins fréquents.
  - Les ~~deux~~ M' (M ou M-1 ou …3 ou 2) moins fréquents symboles ont la même longueur.
- Détermination de M'
- M' désigne le nombre de symbole à regrouper en premier lieu avant de commencer à construire l'arbre.
- Soit un alphabet de longueur n. La valeur de M' est déterminé selon la fonction suivante:
- M'=(M-1)+[n mod(M-1)] (i.e; M=3 et n=6 => M'=2+0=2)

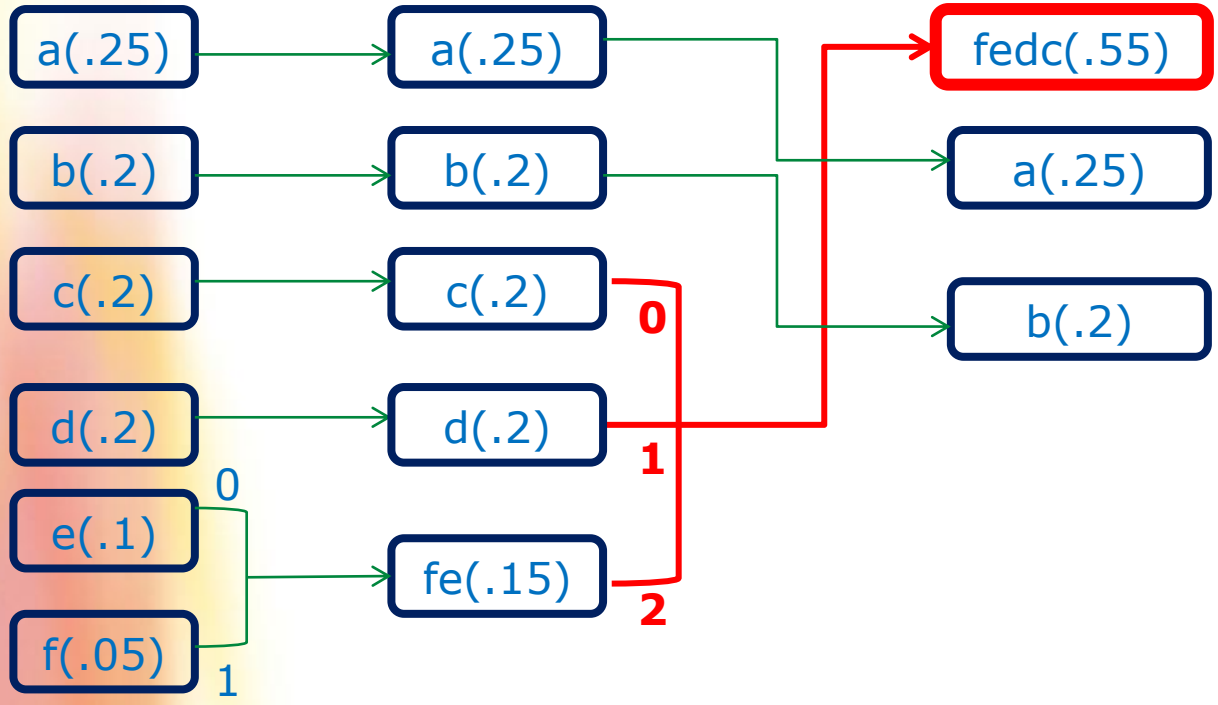
Construction d'un Code de Huffman ternaire (1)

Lettres	a	b	c	d	e	f
Probabilité	0.25	0.20	0.20	0.20	0.10	0.05
Code					**0**	**1**



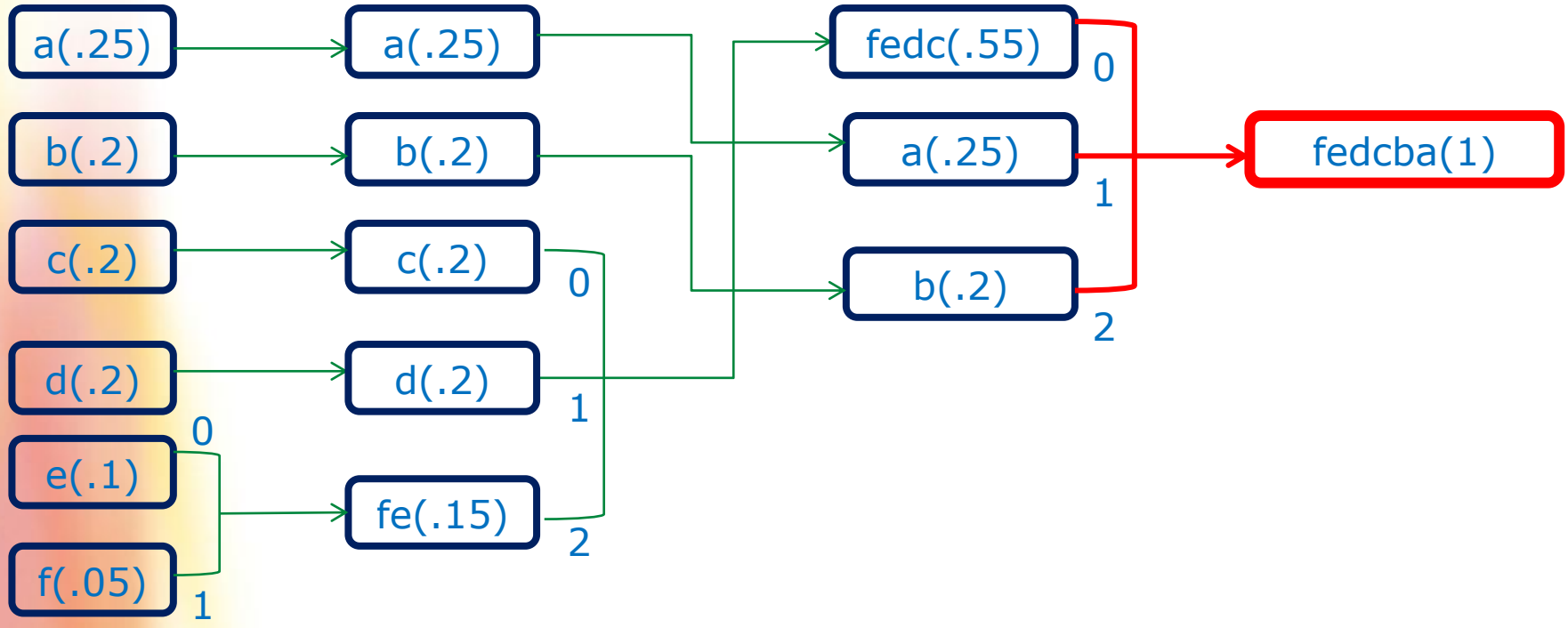
Construction d'un Code de Huffman ternaire (2)

Lettres	a	b	c	d	e	f
Probabilité	0.25	0.20	0.20	0.20	0.10	0.05
Code			**0**	**1**	**2**0	**2**1



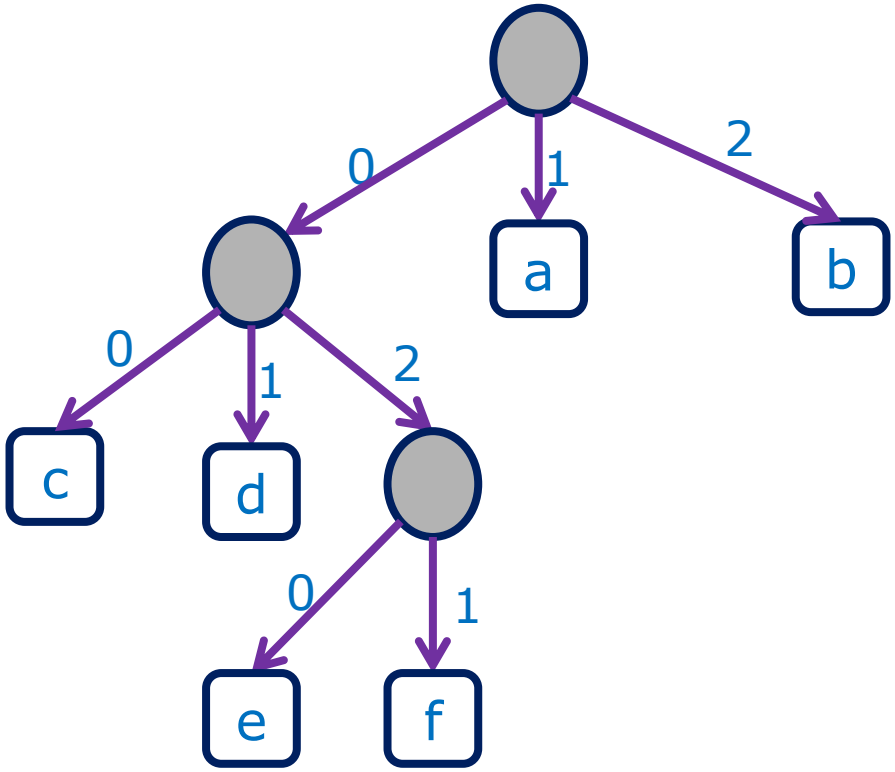
Construction d'un Code de Huffman ternaire (3)

Lettres	a	b	c	d	e	f
Probabilité	0.25	0.20	0.20	0.20	0.10	0.05
Code	**1**	**2**	**0**0	**0**1	**0**20	**0**21



Construction d'un Code de Huffman ternaire [Arbre]

Lettres	a	b	c	d	e	f
Probabilité	0.25	0.20	0.20	0.20	0.10	0.05
Code	1	2	00	01	020	021



Code de Huffman Adaptatif