Les données sont D[1]

Étude de méthodes de compression sans pertes

Thomas BAGREL

Lycée Henri POINCARÉ, Nancy

TIPE session 2018

Aperçu

Régularités et gains

- Théorie
- `zip` recursif

Composantes de la compression

Codage

- Problème résolu
- Inefficacité de Huffman – pourquoi

Régularités et gains

Théorie

Compression des données sans pertes

Régularités et gains

Théorie

Compression des données sans pertes

- exploiter les régularités des données

Régularités et gains

Théorie

Compression des données sans pertes

- exploiter les régularités des données
- données aléatoires : pas de gain

Régularités et gains

Théorie

Compression des données sans pertes

- exploiter les régularités des données
- données aléatoires : pas de gain

Théorème. (Entropie de SHANNON)

$$\mathrm{H(S)} = -\sum_{i=1}^{n} p_i \log_2(p_i)$$

$\mathrm{H(S)}$: *nb. de bits moyen par symbole de la source*

Régularités et gains

Théorie

Compression des données sans pertes

- exploiter les régularités des données
- données aléatoires : pas de gain

Théorème. (Entropie de SHANNON)

$$\mathrm{H(S)} = -\sum_{i=1}^{n} p_i \log_2(p_i)$$

$\mathrm{H(S)}$: *nb. de bits moyen par symbole de la source*

- une fois les données compressées (dérivées) une fois, plus aucune régularité

Régularités et gains

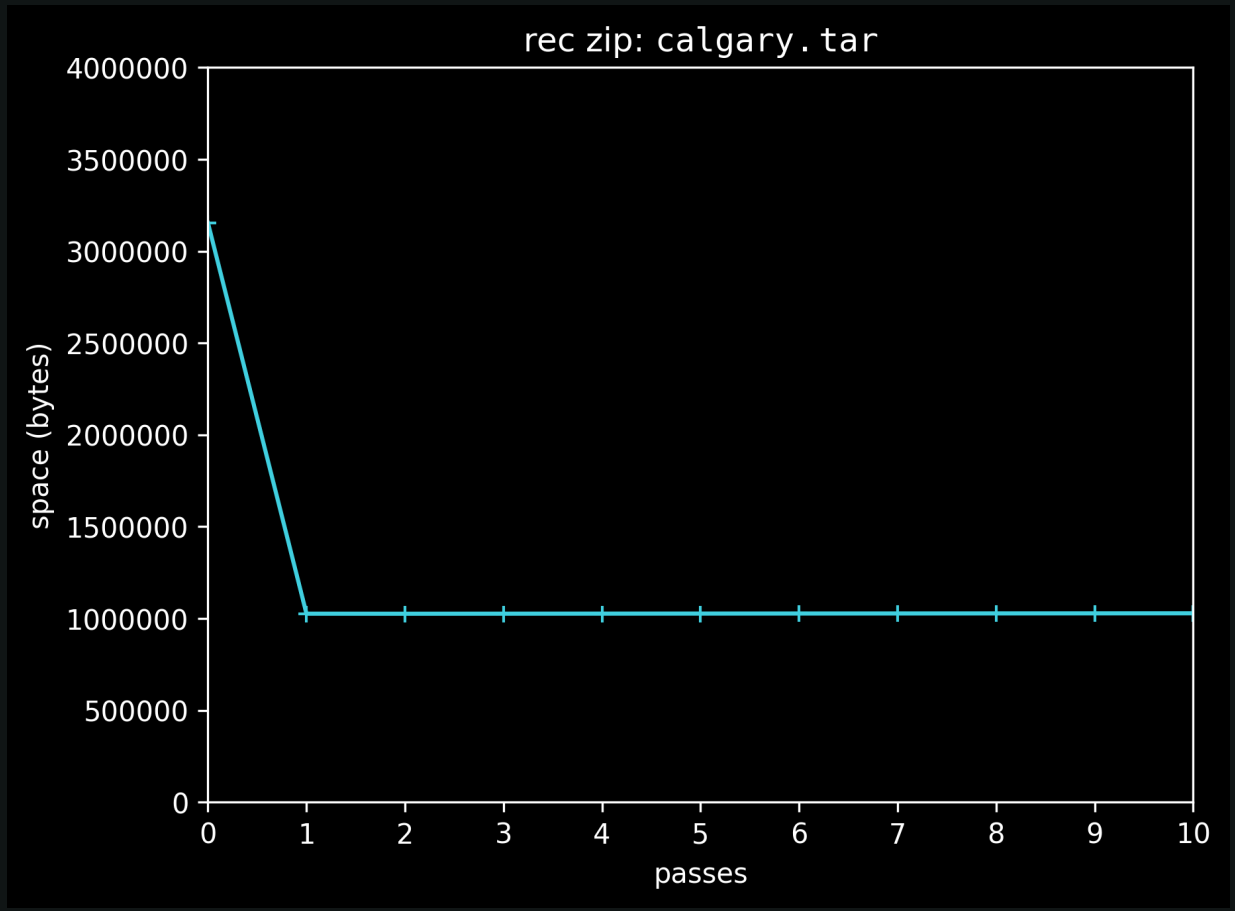
zip recursif

Expérience : compresser récursivement un fichier avec le même algorithme (zip)

Régularités et gains

zip recursif

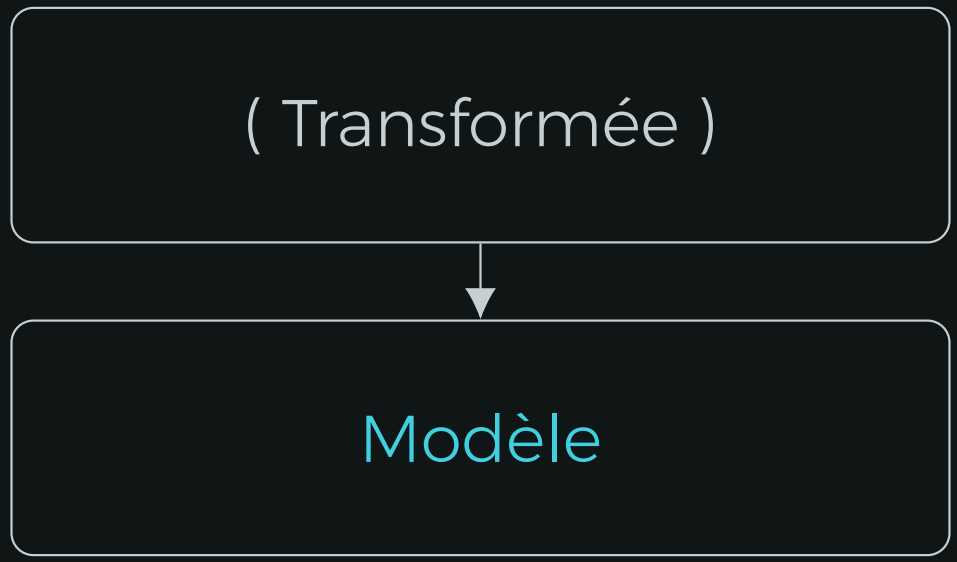
Expérience : compresser récursivement un fichier avec le même algorithme (zip)



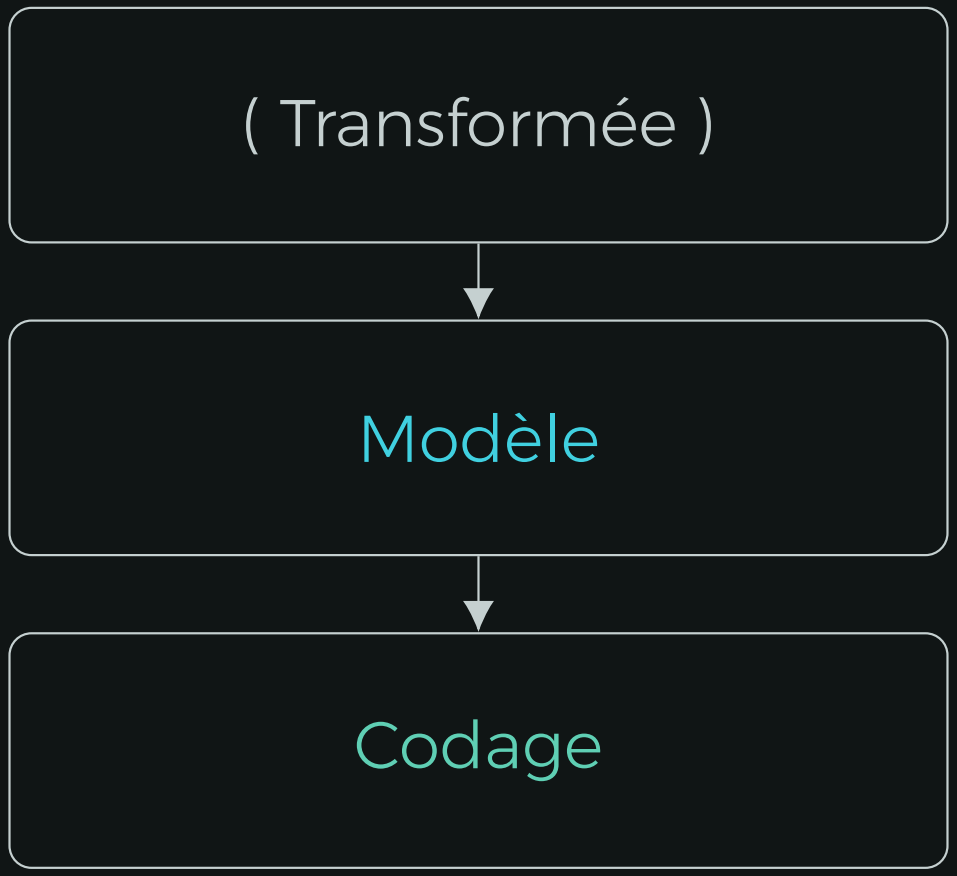
Composantes de la compression

(Transformée)

Composantes de la compression



Composantes de la compression



Codage

Problème résolu

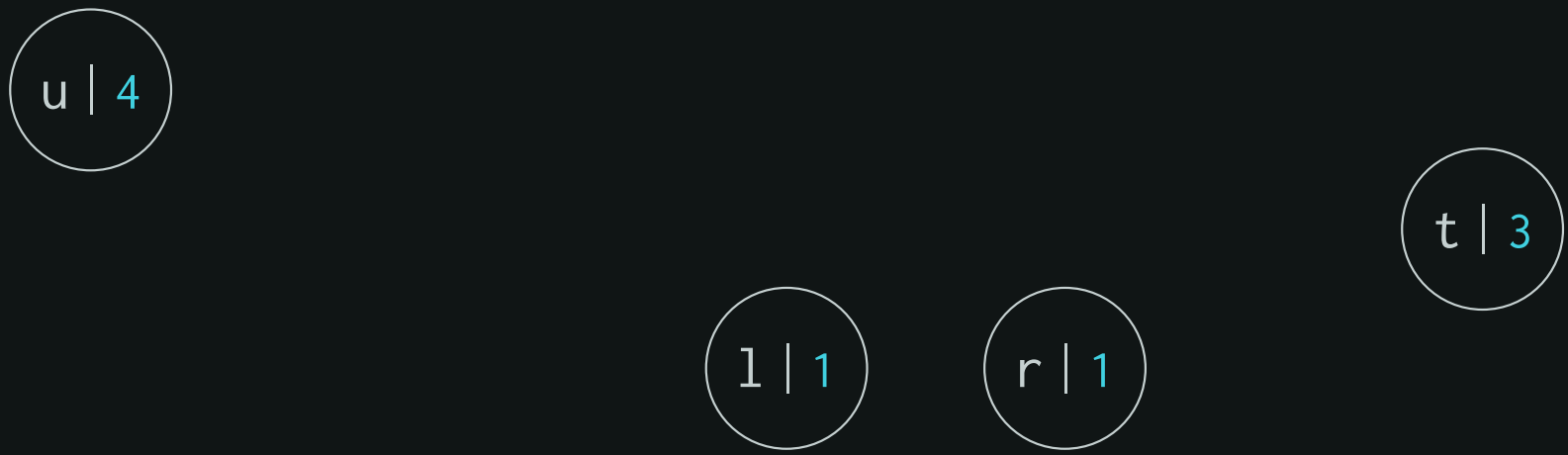
Le codage, contrairement aux apparences, est un problème résolu

- Si les p_i sont connus, la limite de compression théorique est donnée par SHANNON
- HUFFMAN permet d'approcher cette limite
- Le codage arithmétique l'atteint

Codage

Problème résolu

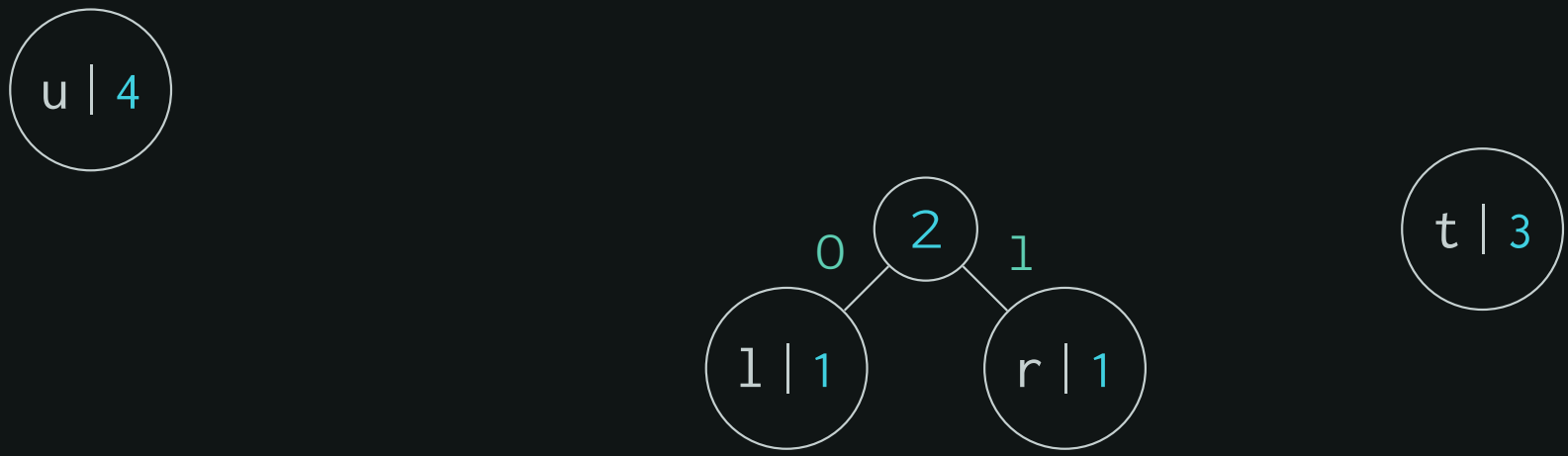
Le codage arithmétique sera détaillé plus loin.
Codage d'HUFFMAN pour `turlututu`



Codage

Problème résolu

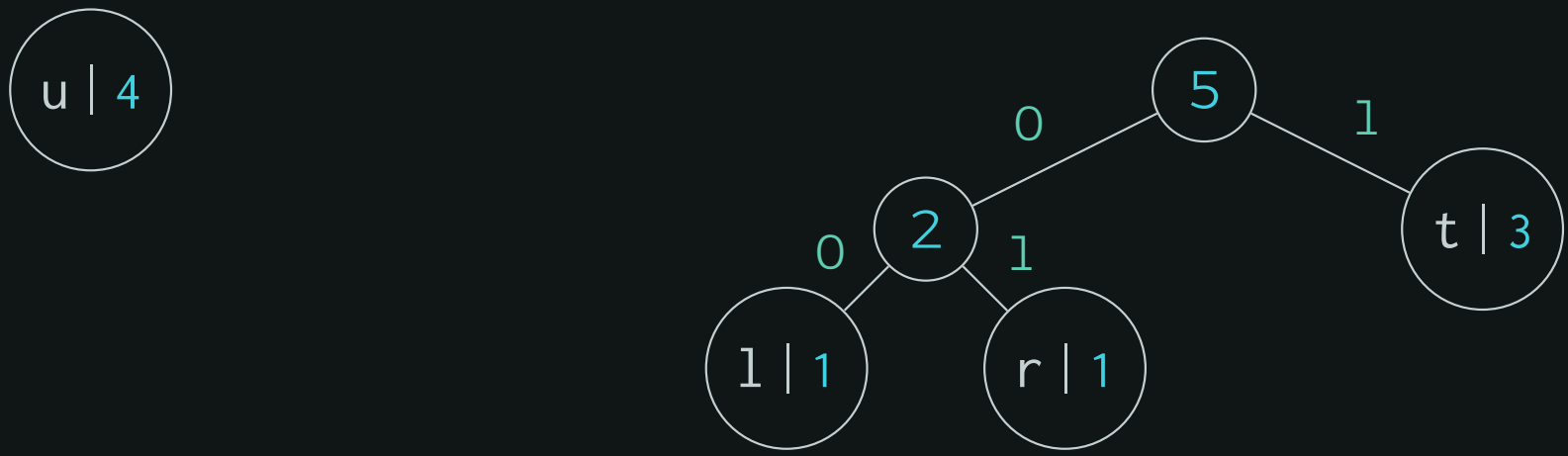
Le codage arithmétique sera détaillé plus loin.
Codage d'HUFFMAN pour `turlututu`



Codage

Problème résolu

Le codage arithmétique sera détaillé plus loin.
Codage d'HUFFMAN pour `turlututu`

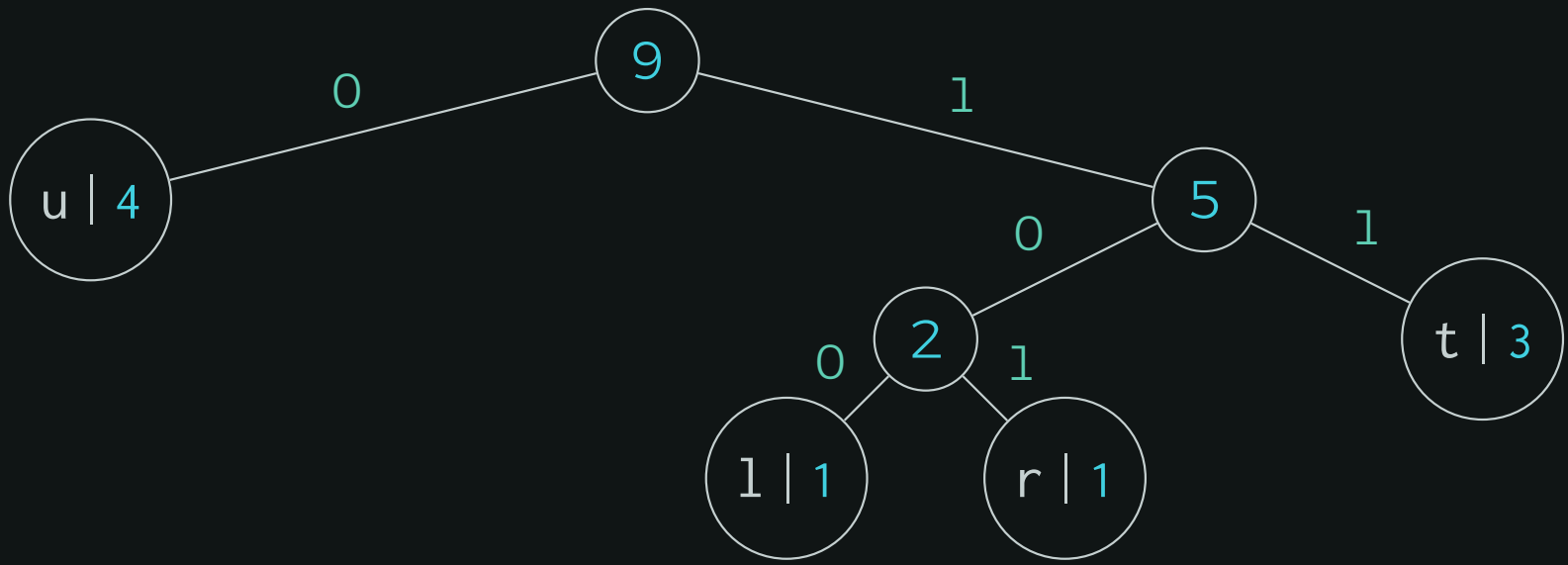


Codage

Problème résolu

Le codage arithmétique sera détaillé plus loin.

Codage d'HUFFMAN pour `turlututu`

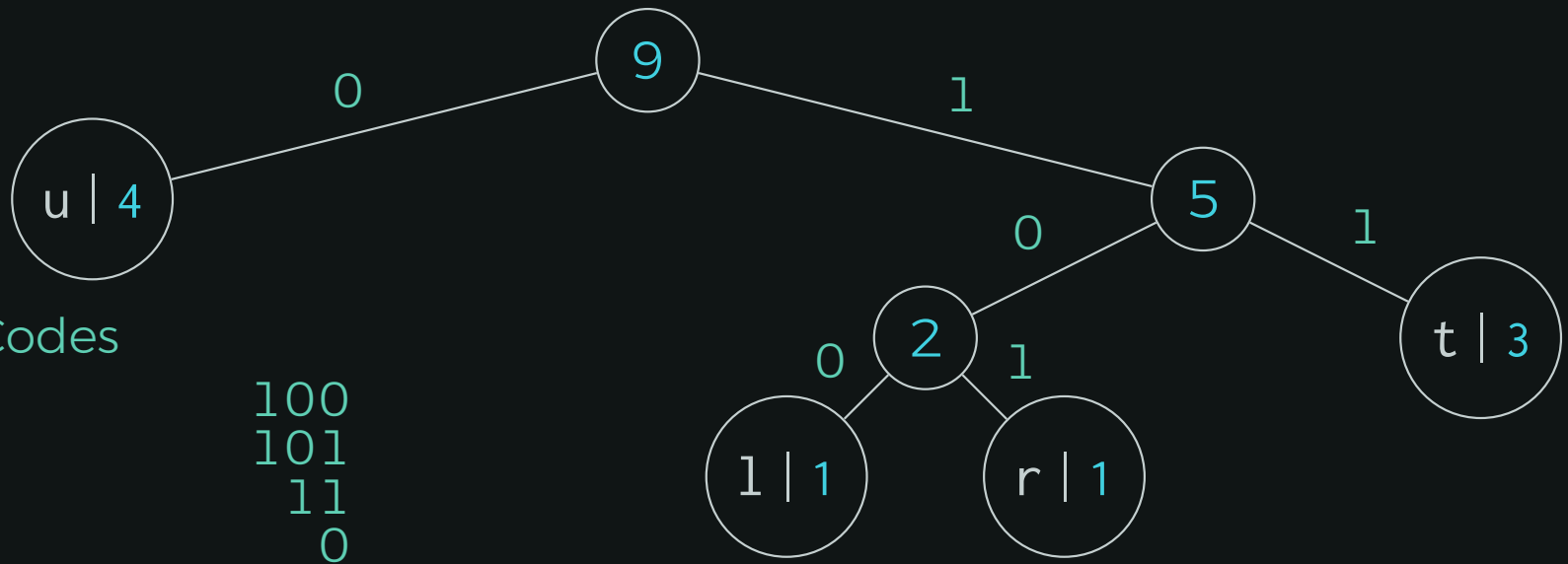


Codage

Problème résolu

Le codage arithmétique sera détaillé plus loin.

Codage d'HUFFMAN pour `turlututu`



Codes

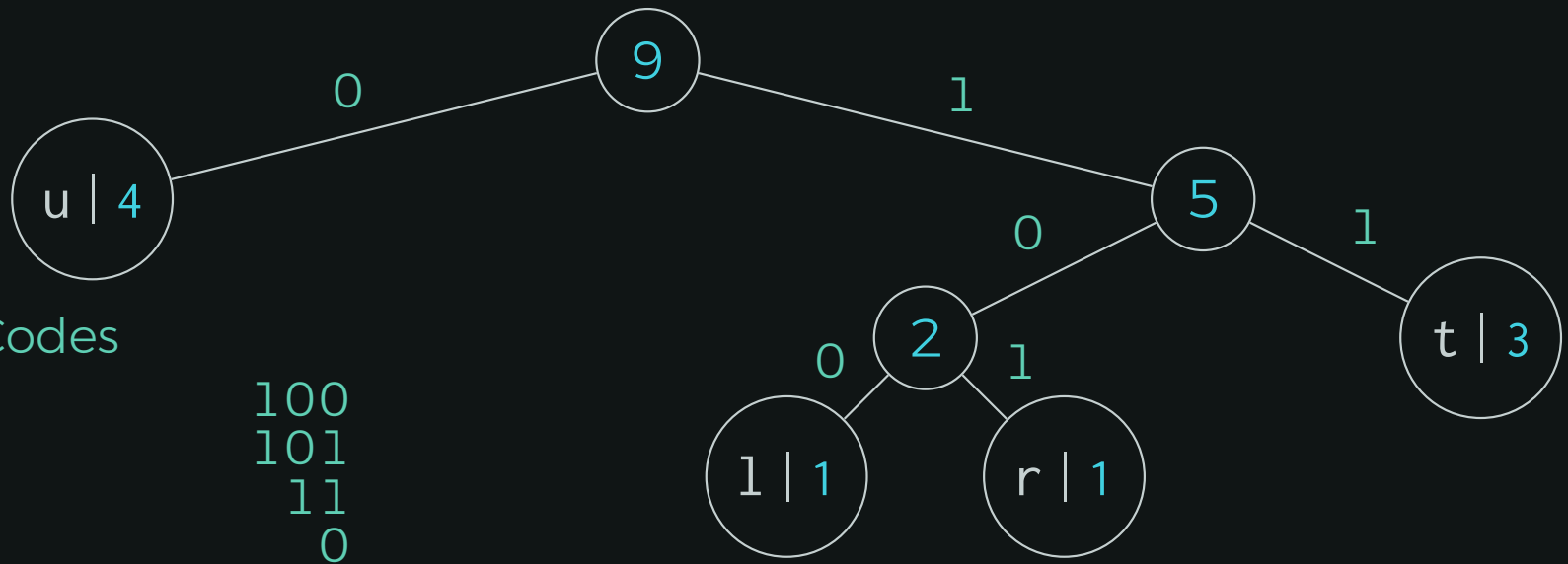
l	100
r	101
t	11
u	0

Codage

Problème résolu

Le codage arithmétique sera détaillé plus loin.

Codage d'HUFFMAN pour `turlututu`



Codes

l	100
r	101
t	11
u	0

Codage

Inefficacité de Huffman – pourquoi

En pratique, efficacité de 30 %

Codage

Inefficacité de Huffman – pourquoi

En pratique, efficacité de 30 %

En appliquant directement Shannon aux fréquences d'apparition, on commet des erreurs

Codage

Inefficacité de Huffman – pourquoi

En pratique, efficacité de 30 %

En appliquant directement Shannon aux fréquences d'apparition, on commet des erreurs

- un symbole n'est pas indépendant des précédents

Codage

Inefficacité de Huffman – pourquoi

En pratique, efficacité de 30 %

En appliquant directement Shannon aux fréquences d'apparition, on commet des erreurs

- un symbole n'est pas indépendant des précédents
- par exemple, en Français, q$\rightarrow$u est plus fréquent que q$\rightarrow$z

Codage

Inefficacité de Huffman – pourquoi

En pratique, efficacité de 30 %

En appliquant directement Shannon aux fréquences d'apparition, on commet des erreurs

- un symbole n'est pas indépendant des précédents
- par exemple, en Français, q$\rightarrow$u est plus fréquent que q$\rightarrow$z
- en quelque sorte, on oublie le caractère lipschitzien de nos données

Il faut donc un modèle