Trois familles de chiffrement

- chiffrement symétrique :
  - L'expéditeur et le destinataire connaissent un même nombre, la clé
  - avantages : temps de calcul courts, peu vulnérable
  - inconvénients : comment transmettre la clé de manière sûre ?

Trois familles de chiffrement

- chiffrement symétrique
- chiffrement asymétrique :
  - 2 clés différentes : une clé « publique » pour chiffrer, une clé « privée » pour déchiffrer. On ne peut pas calculer la clé privée à partir de la clé publique.
  - avantage : pas de problème pour transmettre la clé
  - inconvénients : temps de calcul longs pour permettre une bonne protection
  - exemple : RSA (Rivest Shamir Adleman)

Trois familles de chiffrement

- chiffrement symétrique
- chiffrement asymétrique
- chiffrement hybride :
  - chiffrement symétrique pour les informations → temps de calcul intéressant
  - chiffrement asymétrique de la clé du chiffrement symétrique → pas de problème de transmission de la clé

Exemple de chiffrement asymétrique

Expéditeur

Destinataire

Exemple de chiffrement asymétrique

Expéditeur

Destinataire

1. Création simultanée d'une clé **publique** et d'une clé **privée**

2. Envoi de la clé **publique** (sur un canal non protégé)

3. Chiffrement de l'information avec la clé **publique**

4. Envoi du message chiffré avec la clé **publique** (canal non protégé)

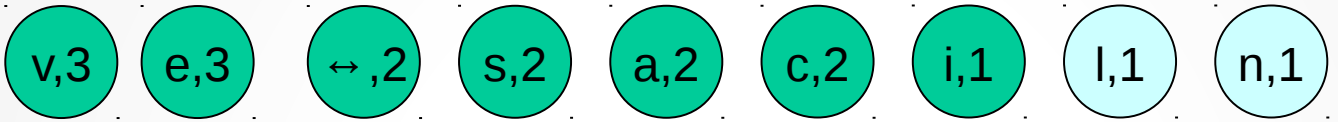
5. Déchiffrement de l'information avec la clé **privée**

Un exemple de compactage statistique : le codage de Huffman

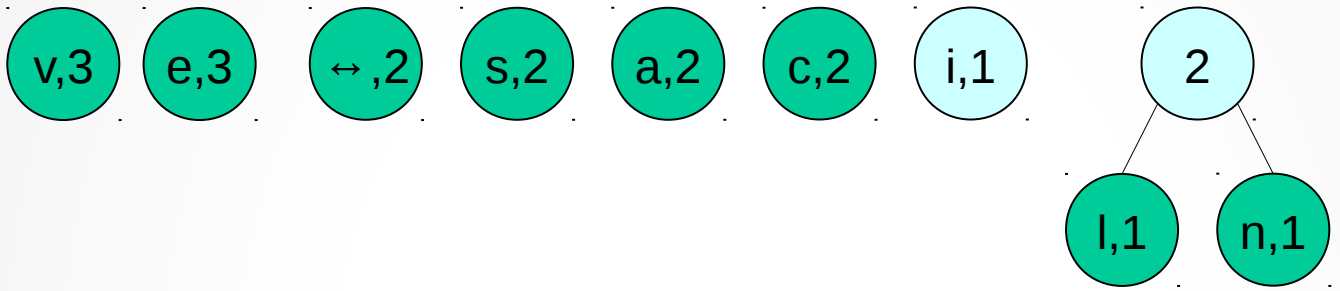
- Exemple : on veut coder la chaîne « vive les vacances ».
  - comptage des effectifs :

  - récupération de deux nœuds de poids minimal, combinaison en un nouveau nœud (le plus faible à droite). Les deux nœuds sont enlevés de l'ensemble des nœuds à traiter.
  - répétition jusqu'à obtenir un seul nœud.

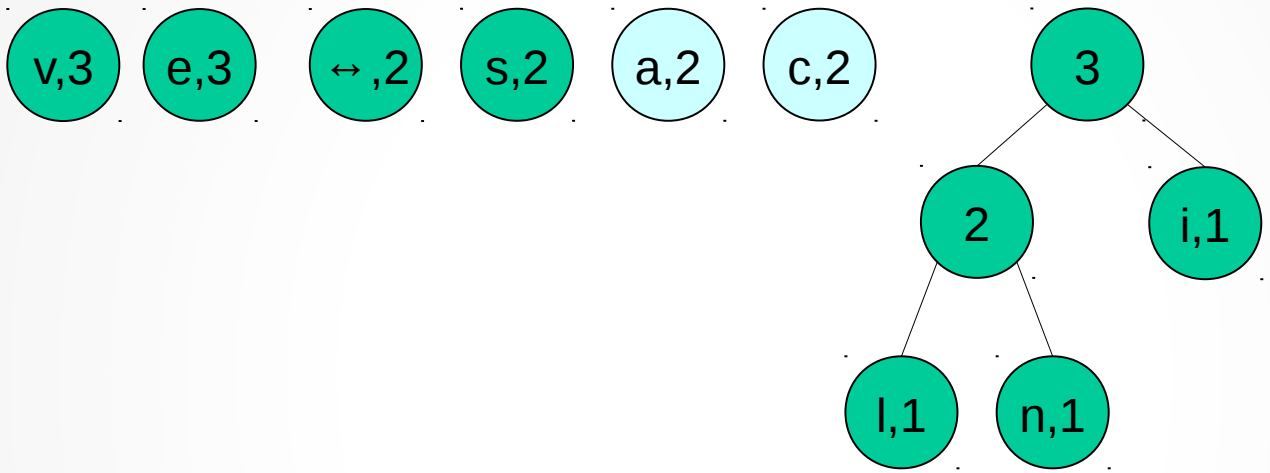
Huffman en action



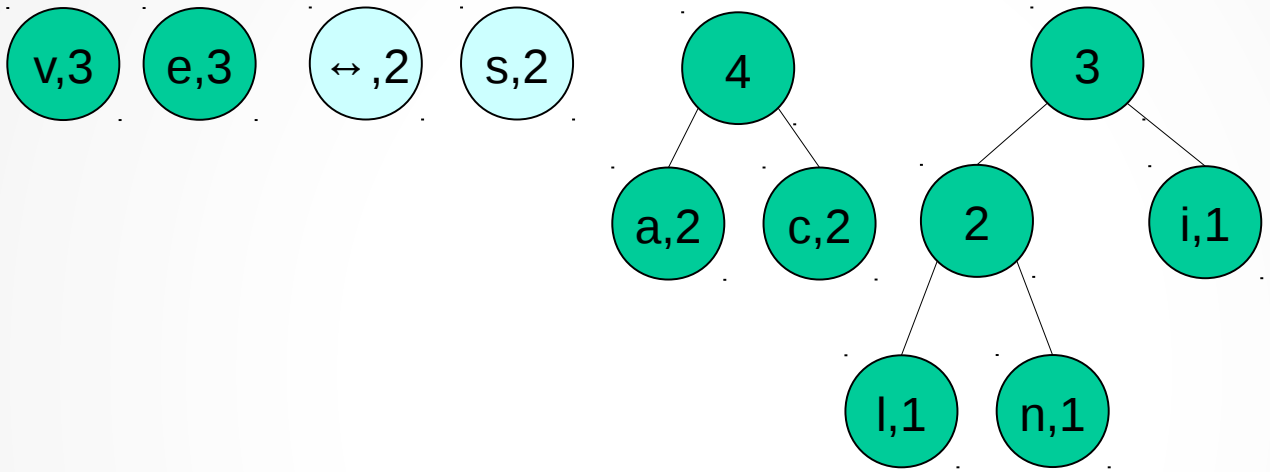
Huffman en action



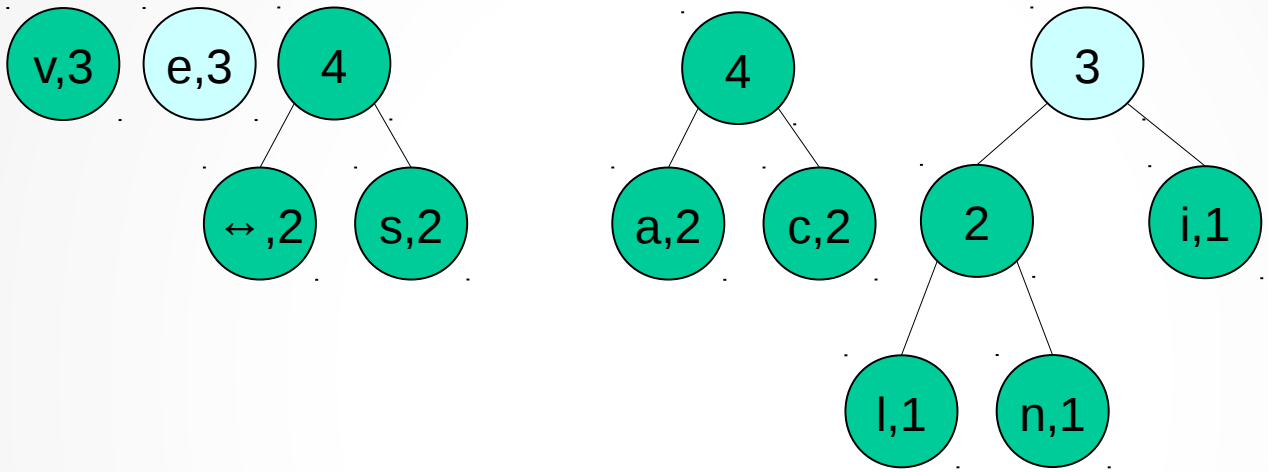
Huffman en action



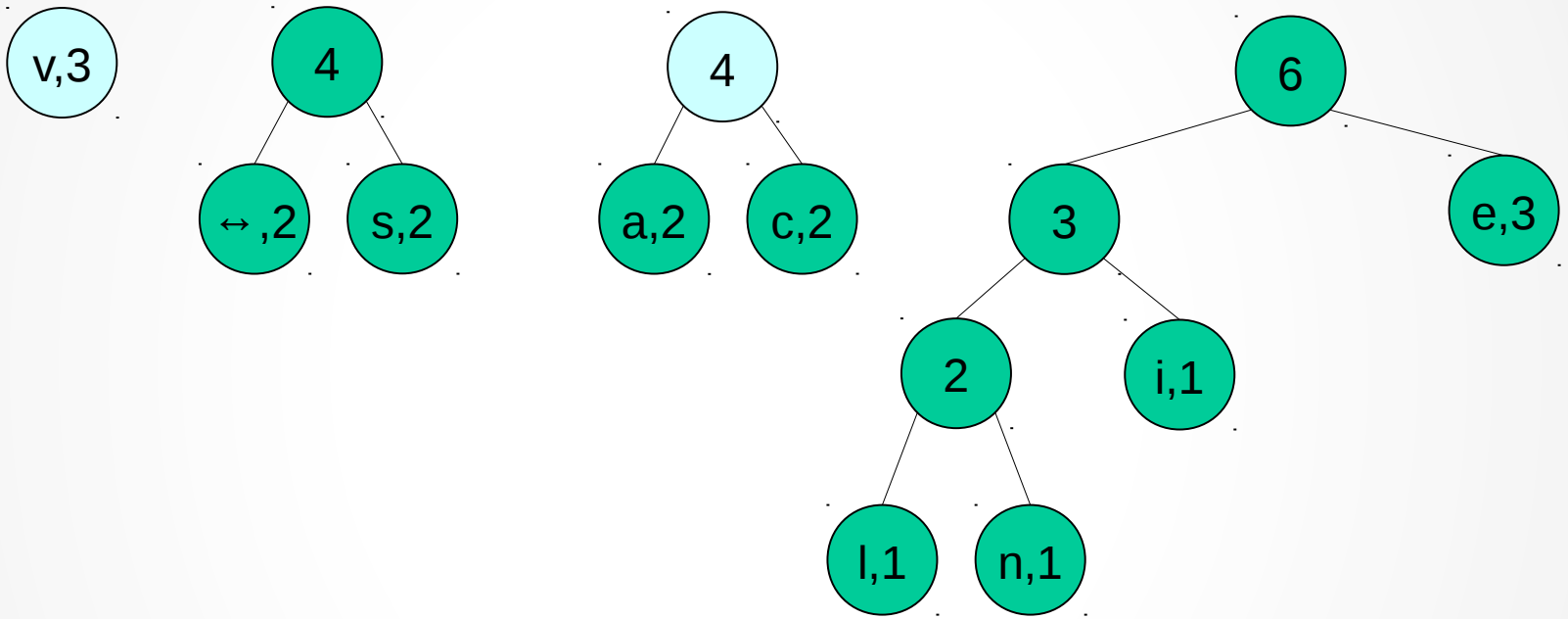
Huffman en action



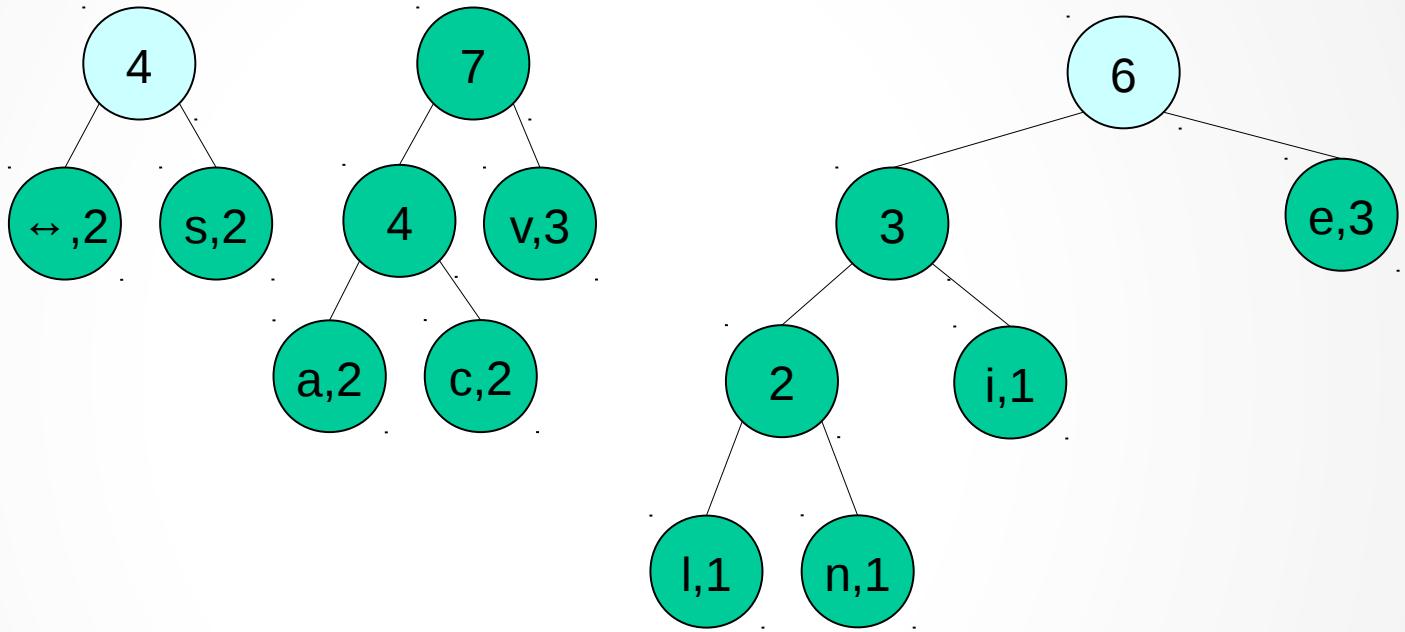
Huffman en action



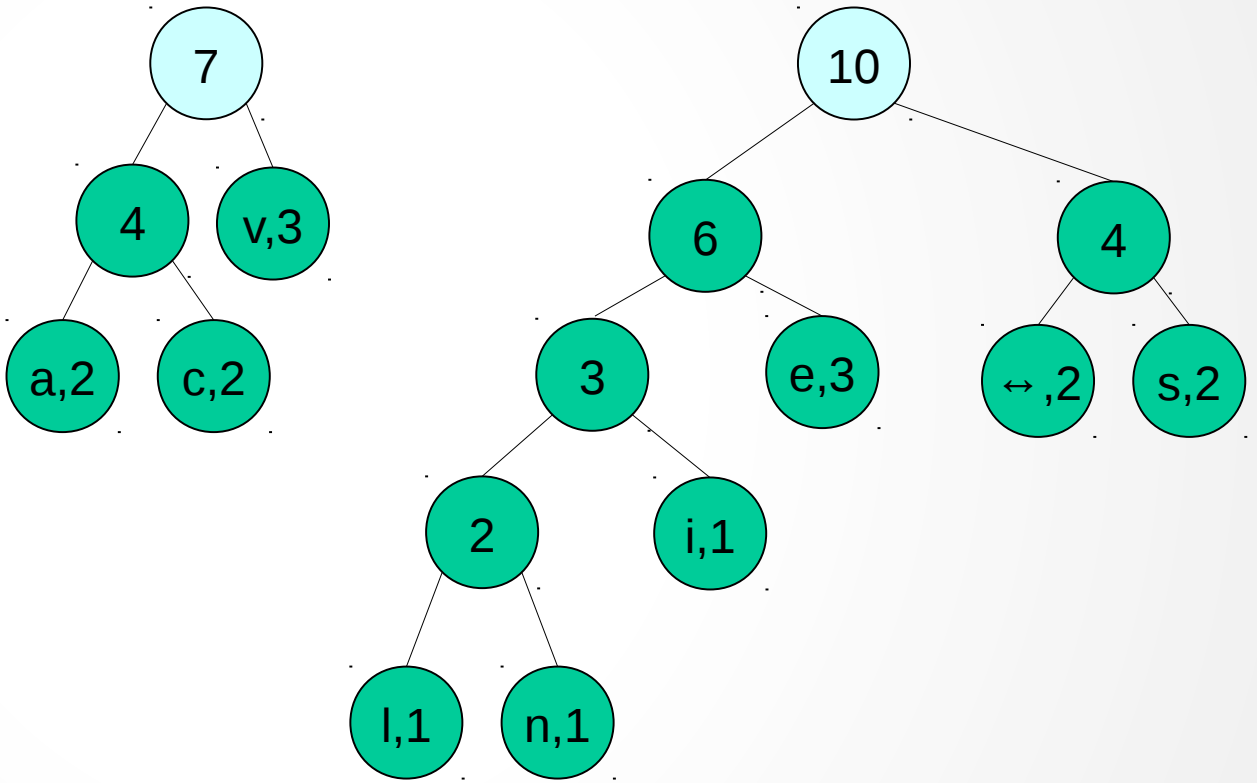
Huffman en action



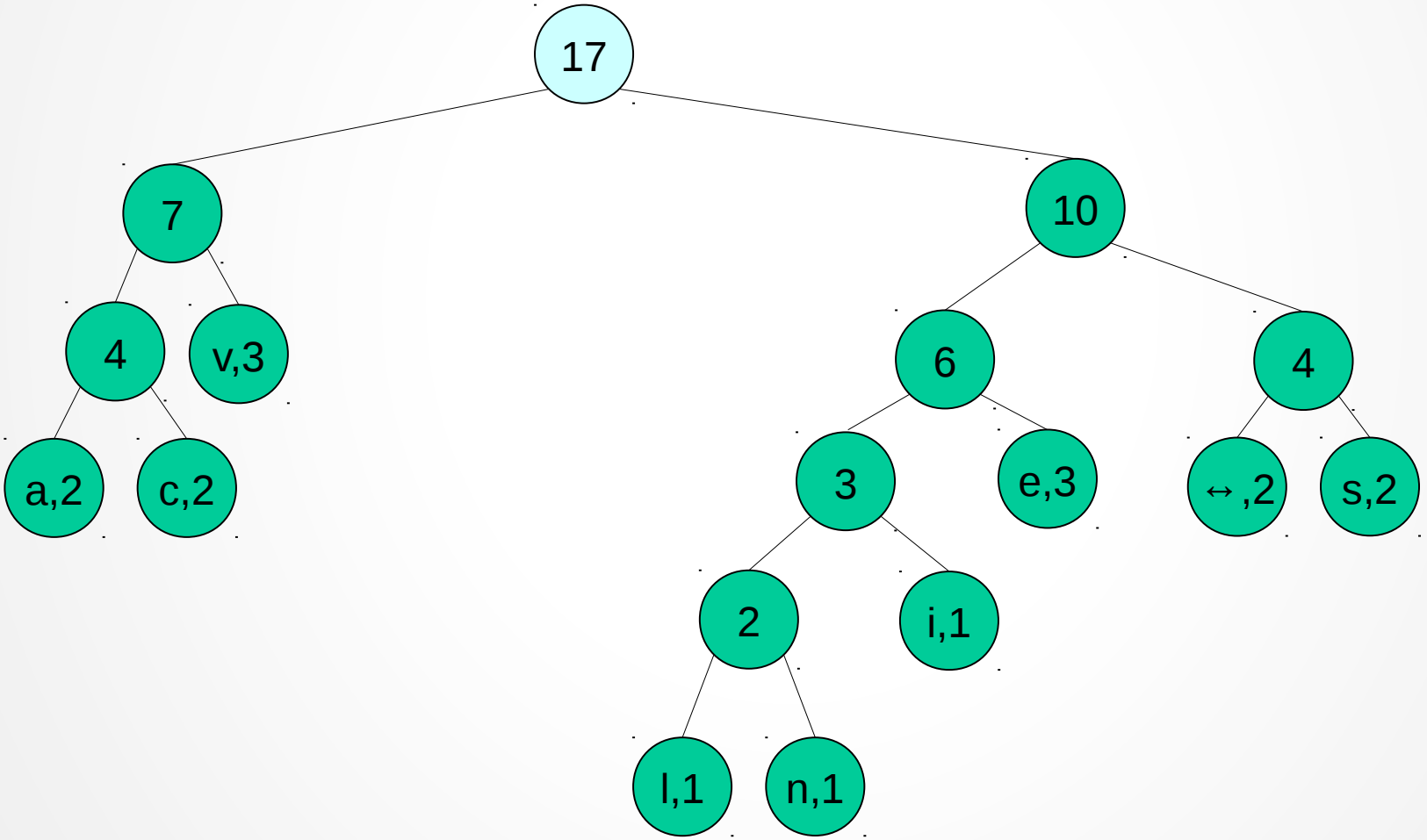
Huffman en action



Huffman en action



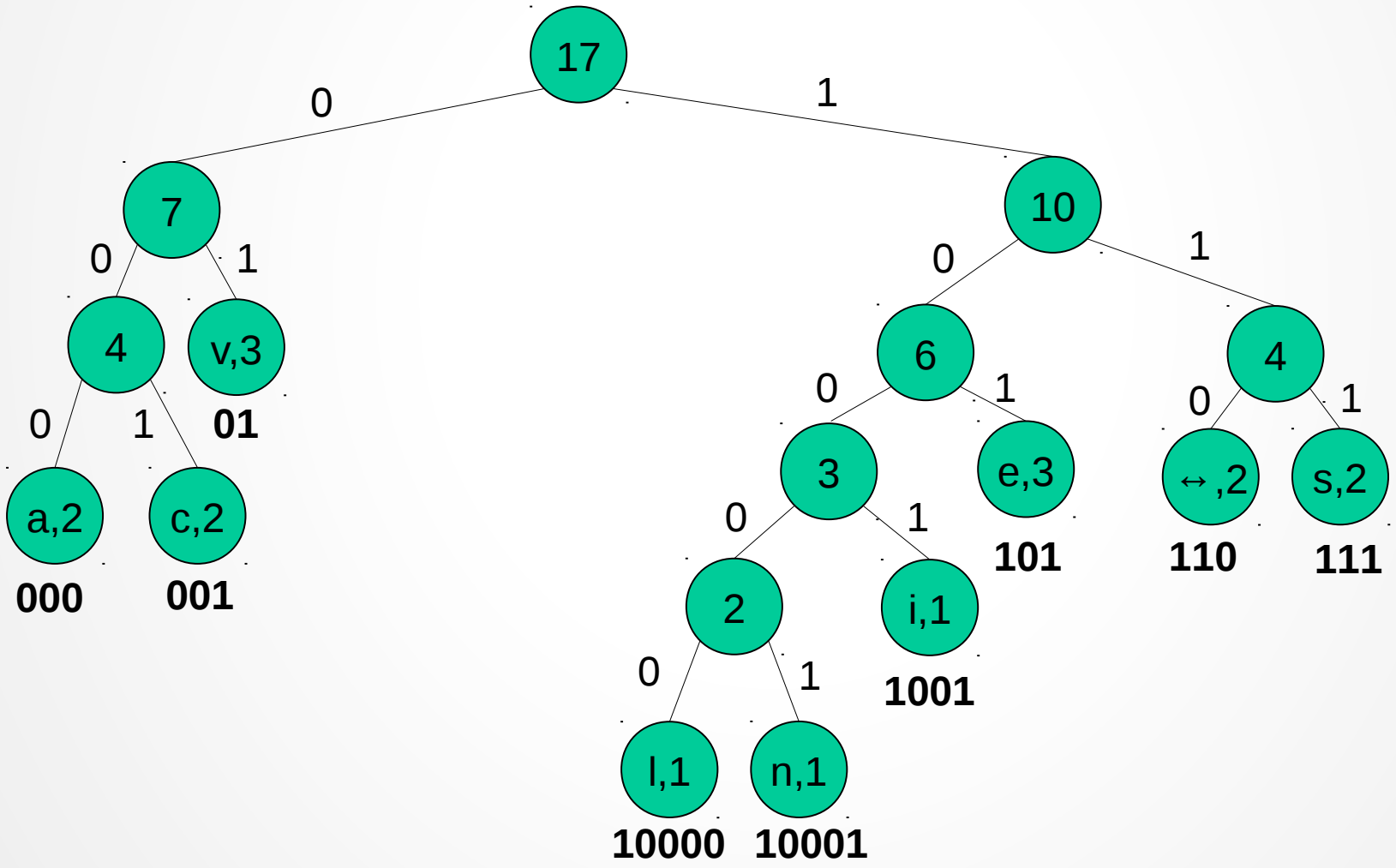
Huffman en action



Huffman : codage des caractères

- On assigne des codes aux branches, 0=gauche, 1=droite
- Le chemin parcouru de la racine à la feuille d'un caractère donne son code
- Remarques :
  - aucun code n'est préfixe d'un autre
  - les caractères les plus répandus (= de poids maximal) se retrouvent en haut de l'arbre avec un codage court

Huffman en action



Huffman : compactage

- D'où la représentation de la chaîne :

v	i	v	e	_	l	e	s	_	v	a	c	a	n	c	e	s
01	1001	01	101	110	10000	101	111	110	01	000	001	000	10001	001	101	111

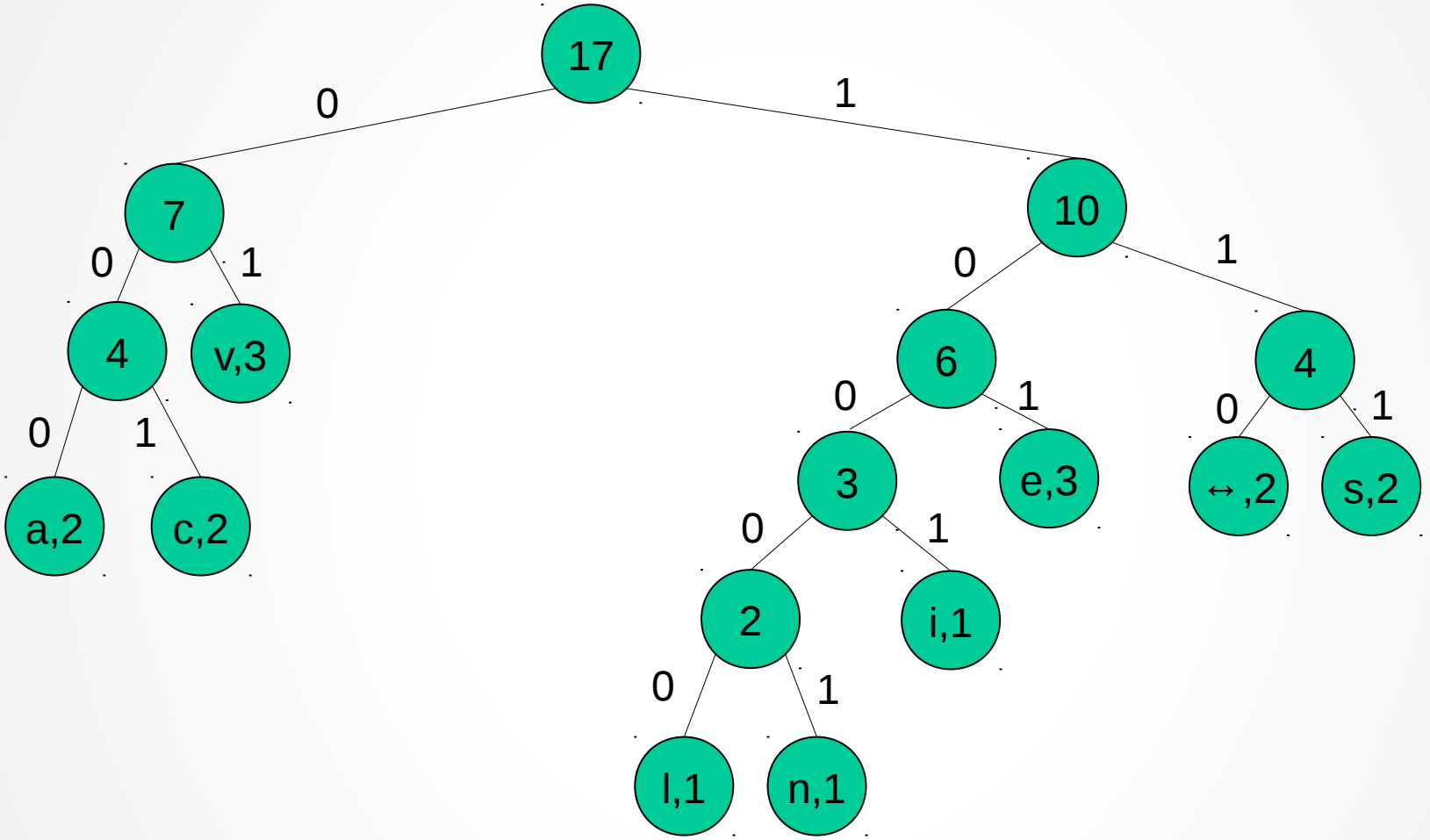
0110010110 11101000 01011111 10010000 01000100 01001101 111

→ codage sur 7 octets, au lieu de 17 au début

Huffman : décompression

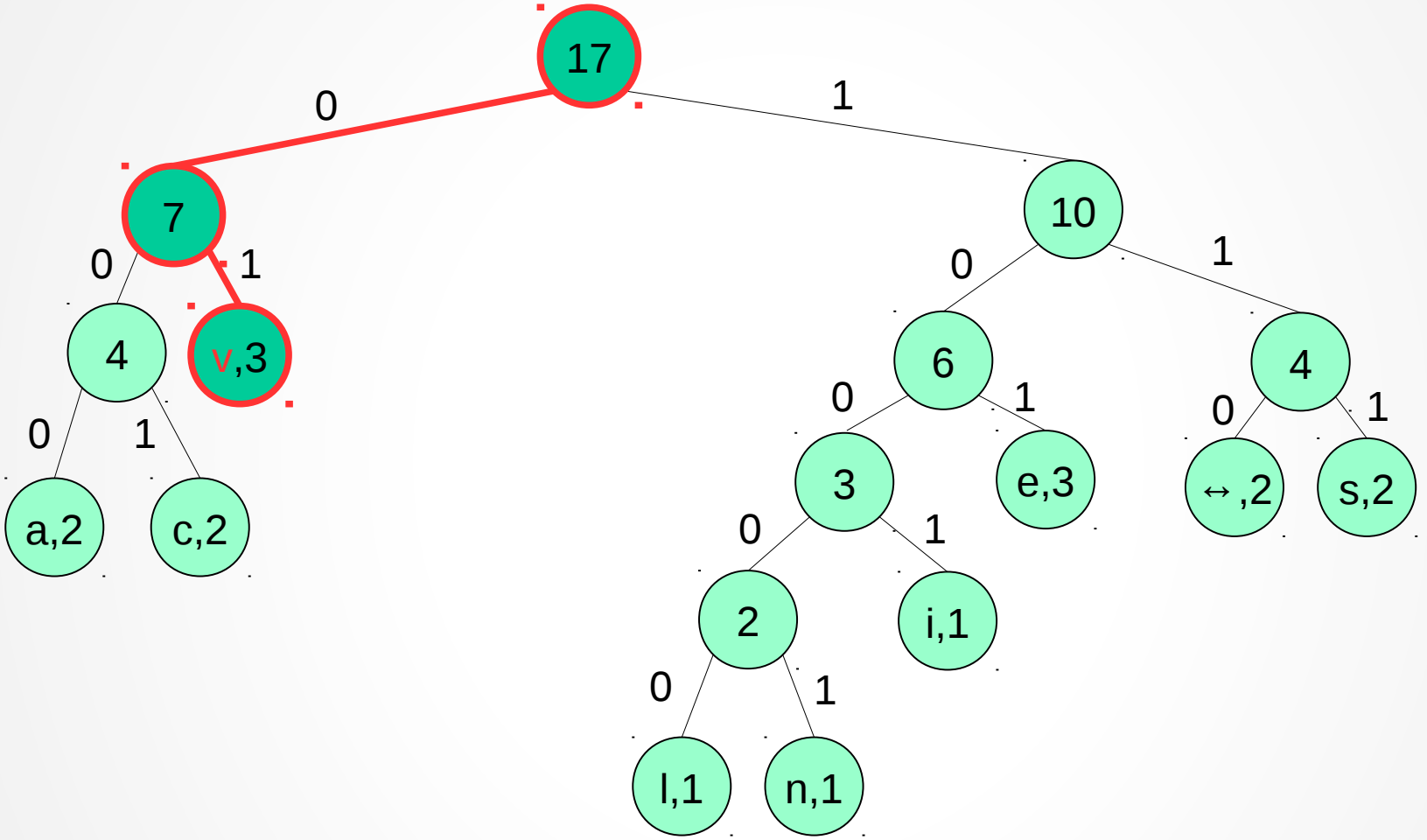
- Pour décompacter, on parcourt l'arbre en fonction des bits lus dans le fichier compacté :
  1. on part de la racine de l'arbre
  2. à chaque 0, on se dirige à gauche
  3. à chaque 1, on se dirige à droite
  4. lorsqu'on arrive sur une feuille, on renvoie le caractère correspondant et on repart à l'étape 1
- La correspondance caractère / code binaire est stockée en tête du fichier compacté (encombrement négligeable pour un fichier volumineux) sauf si on utilise une correspondance existante.

Huffman en action




```
01100101101110100001011111100100000100010001001101111 = ?
```

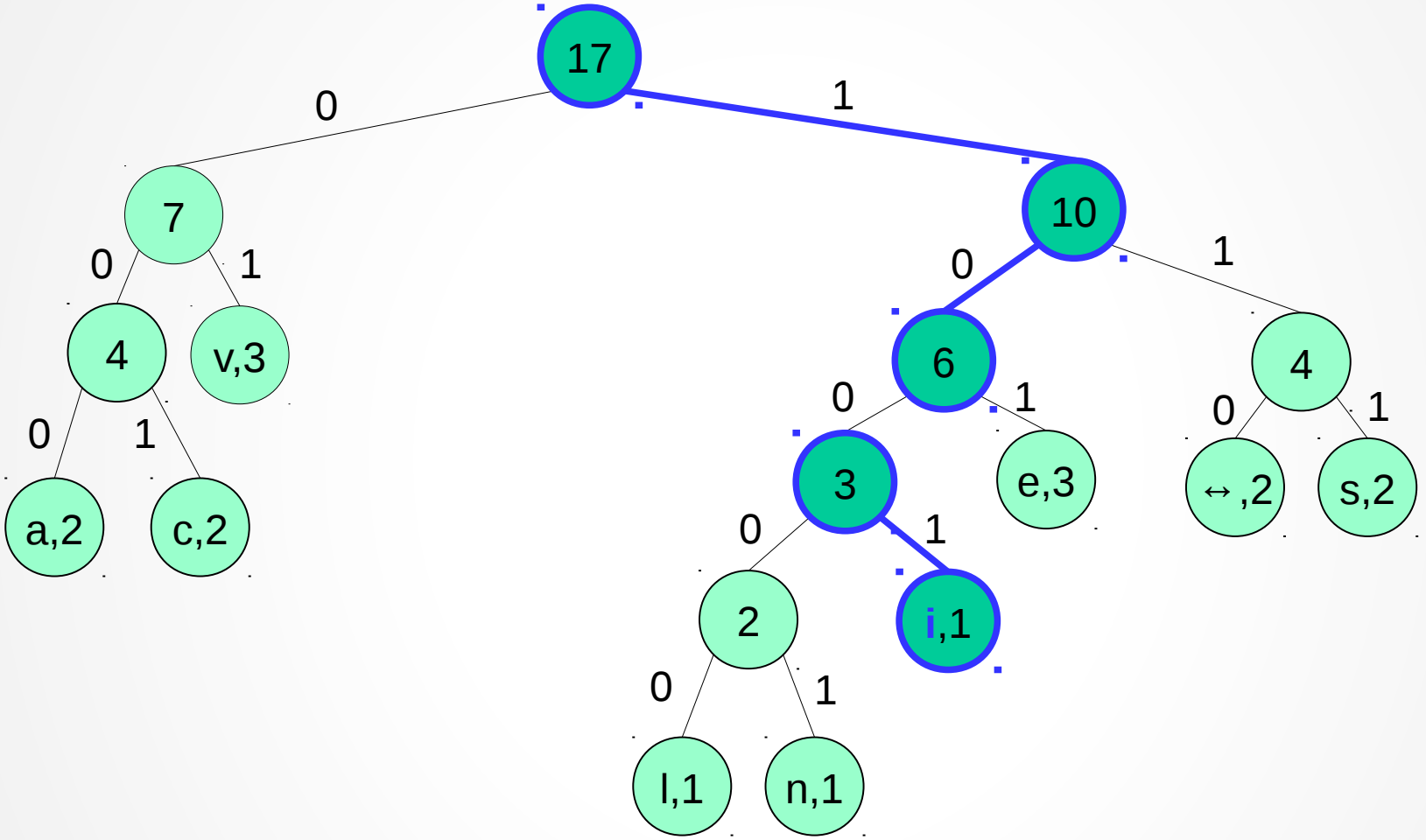
Huffman en action



01100101101110100001011111100100000100010001001101111

v

Huffman en action



0110010110111010000101111110010000010001000100111011111

v **i** ...

Merci pour votre attention