Codificación Huffman

Estructuras de Datos

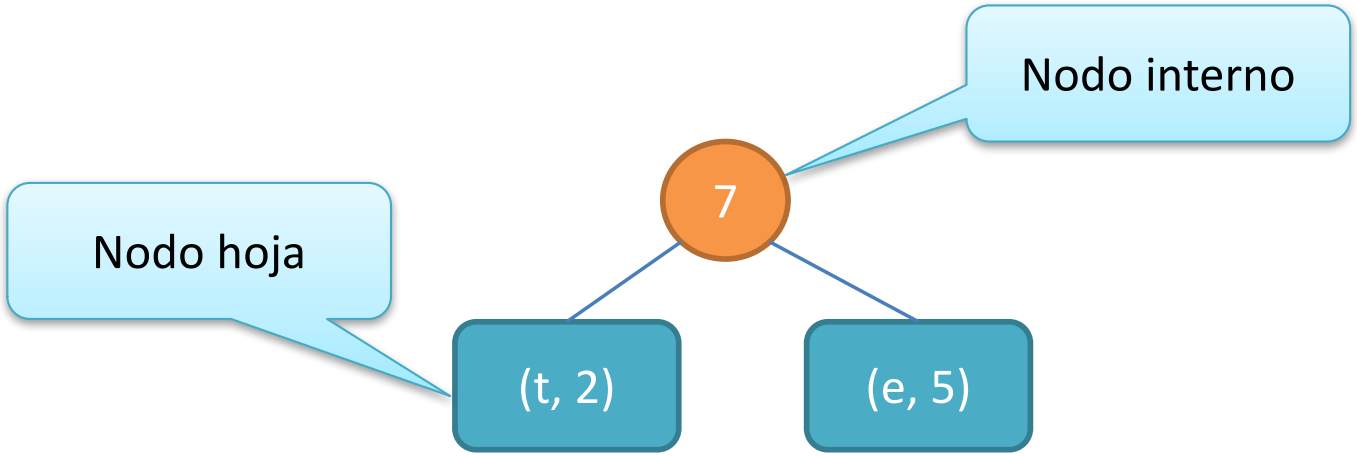
Calcular la frecuencia de aparición

- Para el mensaje “abracadabra” tenemos las siguientes frecuencias o pesos

Carácter	Frecuencia (Peso)
a	5
b	2
c	1
d	1
r	2

Árboles de Huffman

- Un árbol de Huffman es un árbol binario:
  - Los nodos internos contienen pesos
  - Los nodos hoja contienen un carácter y su peso

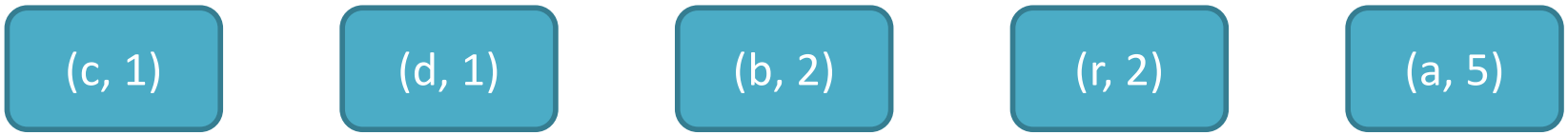


Construcción de árboles de Huffman

- Los árboles de Huffman se construyen de abajo a arriba, desde las hojas hasta la raíz
- Se parte de una colección de ***n*** árboles hoja
- Se extraen los 2 árboles de menor peso
- Se inserta la mezcla (suma de pesos) de ambos árboles, obteniendo una colección de ***n-1*** árboles
- El proceso continúa hasta obtener una colección con un solo árbol de Huffman

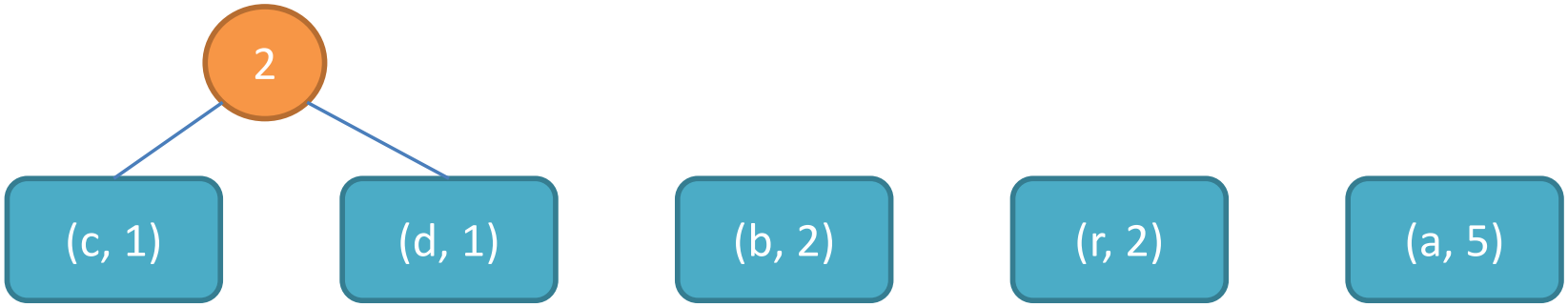
Colección inicial de árboles hoja

- Para “abracadabra” tenemos 5 árboles hoja, uno por cada carácter del mensaje



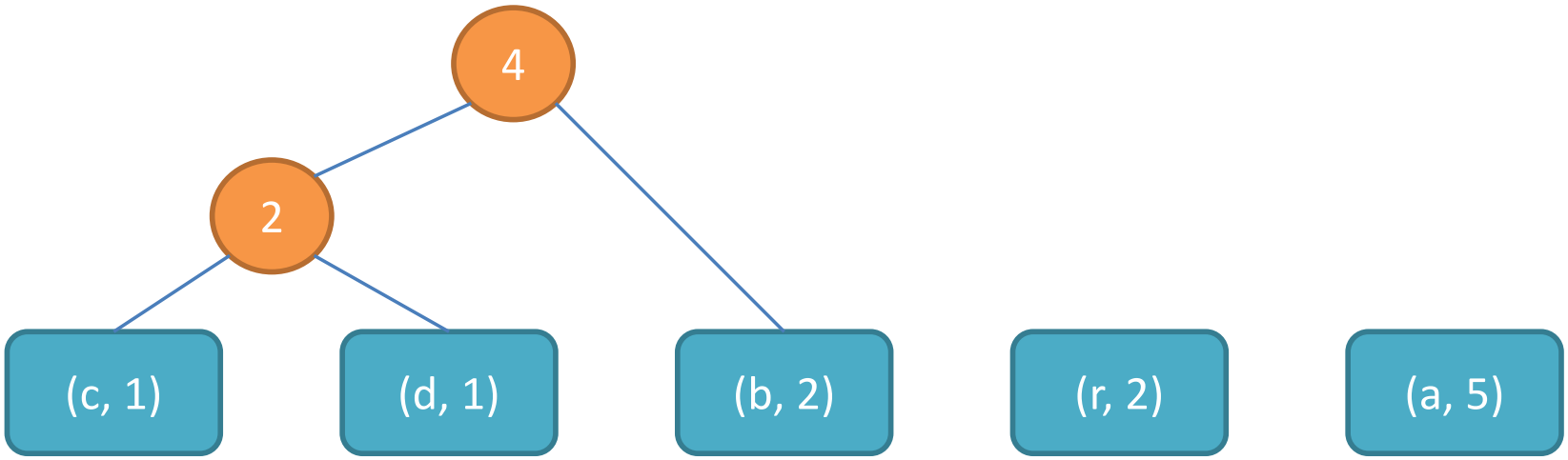
Reducción por mezcla (I)

- Se extraen los 2 árboles de menor peso y se inserta su mezcla (se suman los pesos)
- Esta colección contiene 4 árboles



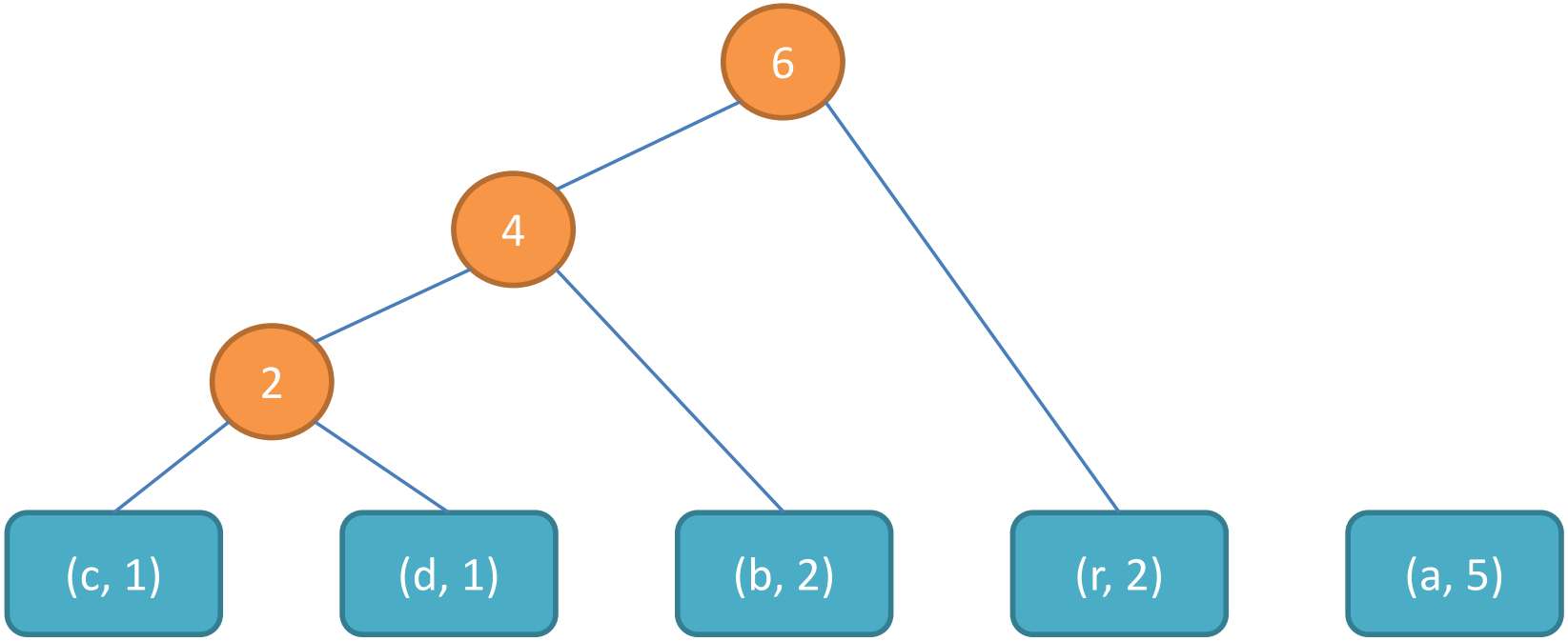
Reducción por mezcla (II)

- Se extraen los 2 árboles de menor peso y se inserta su mezcla (se suman los pesos)
- Esta colección tiene 3 árboles



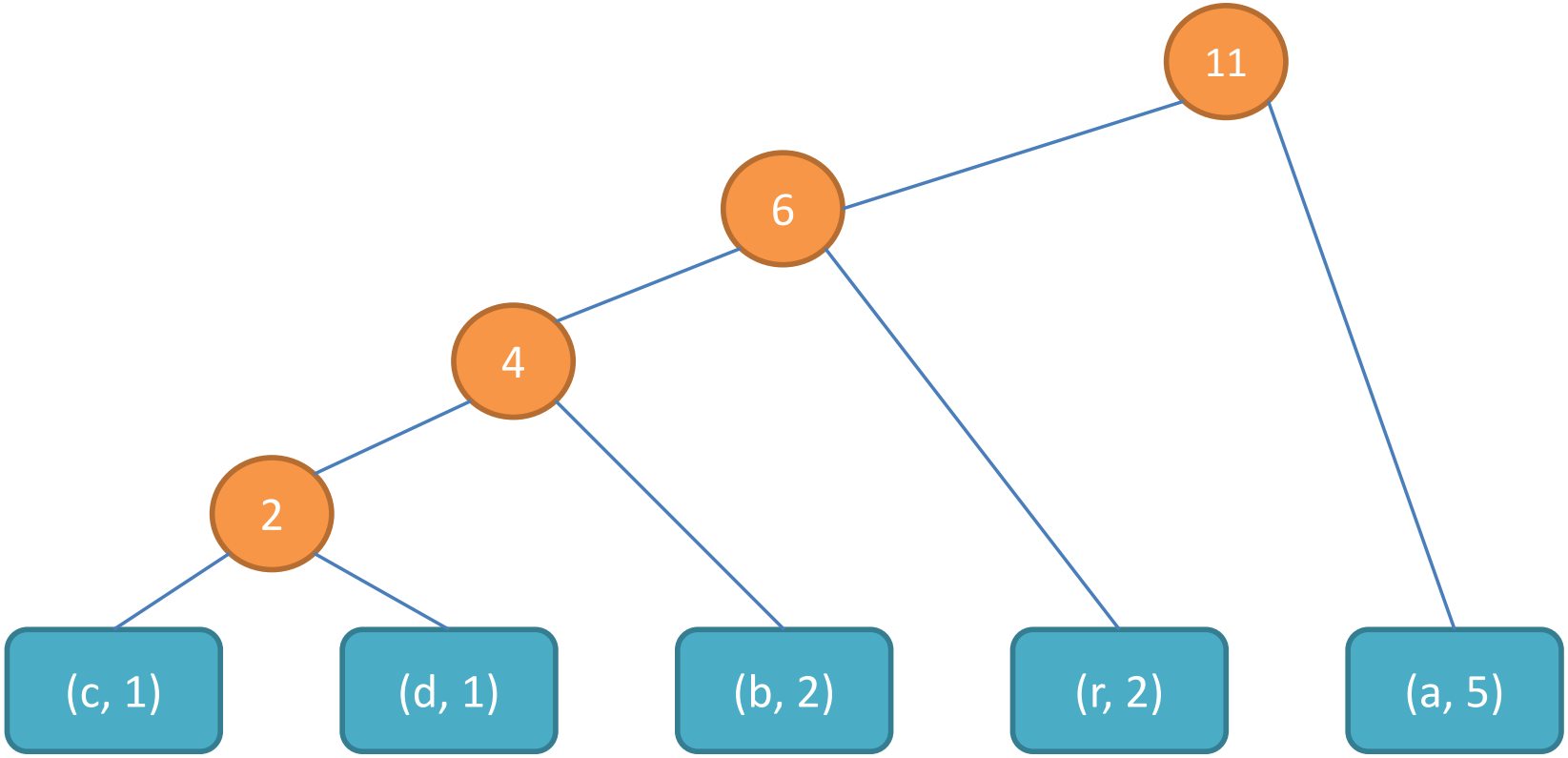
Reducción por mezcla (III)

- Repetimos el proceso…



Reducción por mezcla (y IV)

- Hasta obtener un único árbol de Huffman



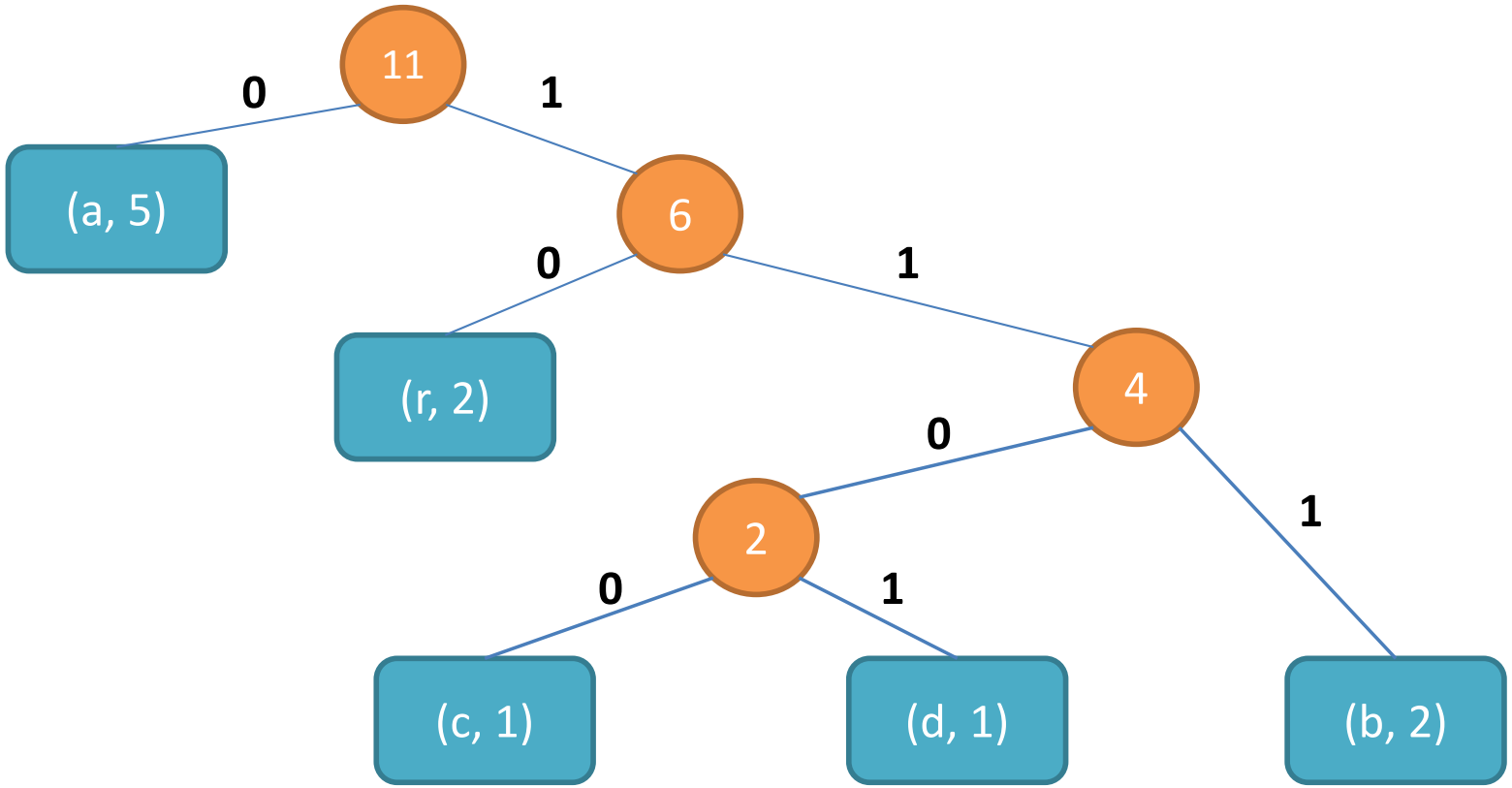
El código de Huffman no es único

Depende de:

- la política de inserción y eliminación de la colección, que rompe empates entre árboles del mismo peso
- la mezcla de árboles, que decide qué árbol queda a izquierda/derecha

La rotación de un árbol de Huffman también es un árbol de Huffman

Otro árbol para “abracadabra”



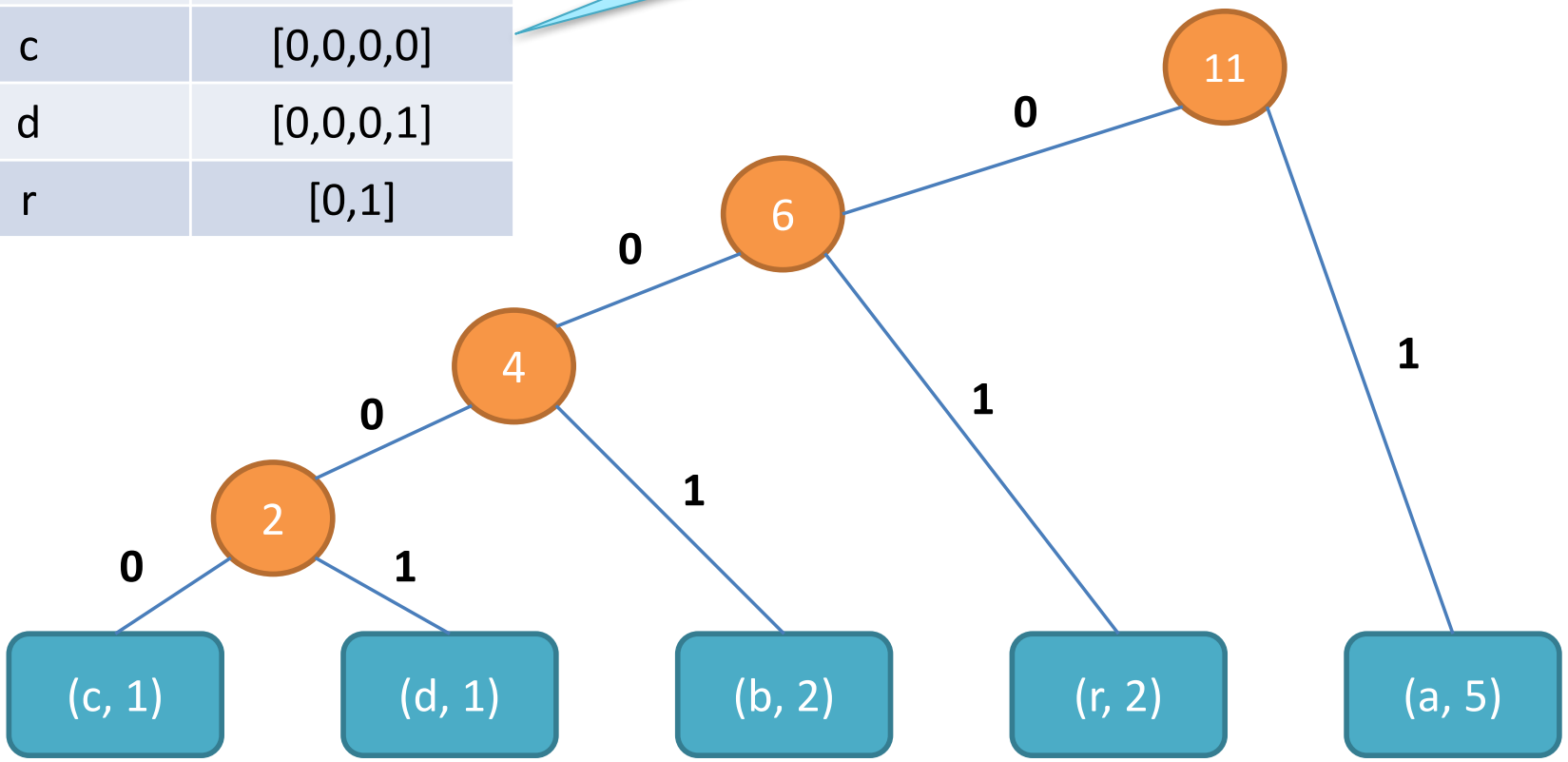
Código de Huffman (I)

- El código de un carácter ***x*** se obtiene recorriendo la rama que va desde la raíz hasta la hoja que contiene a ***x***
- Al descender por la izquierda se añade un ***0*** al código, al descender por la derecha un ***1***
- El número de bits del código de ***x*** es igual a la profundidad a la que se encuentra ***x***

Código de Huffman (y II)

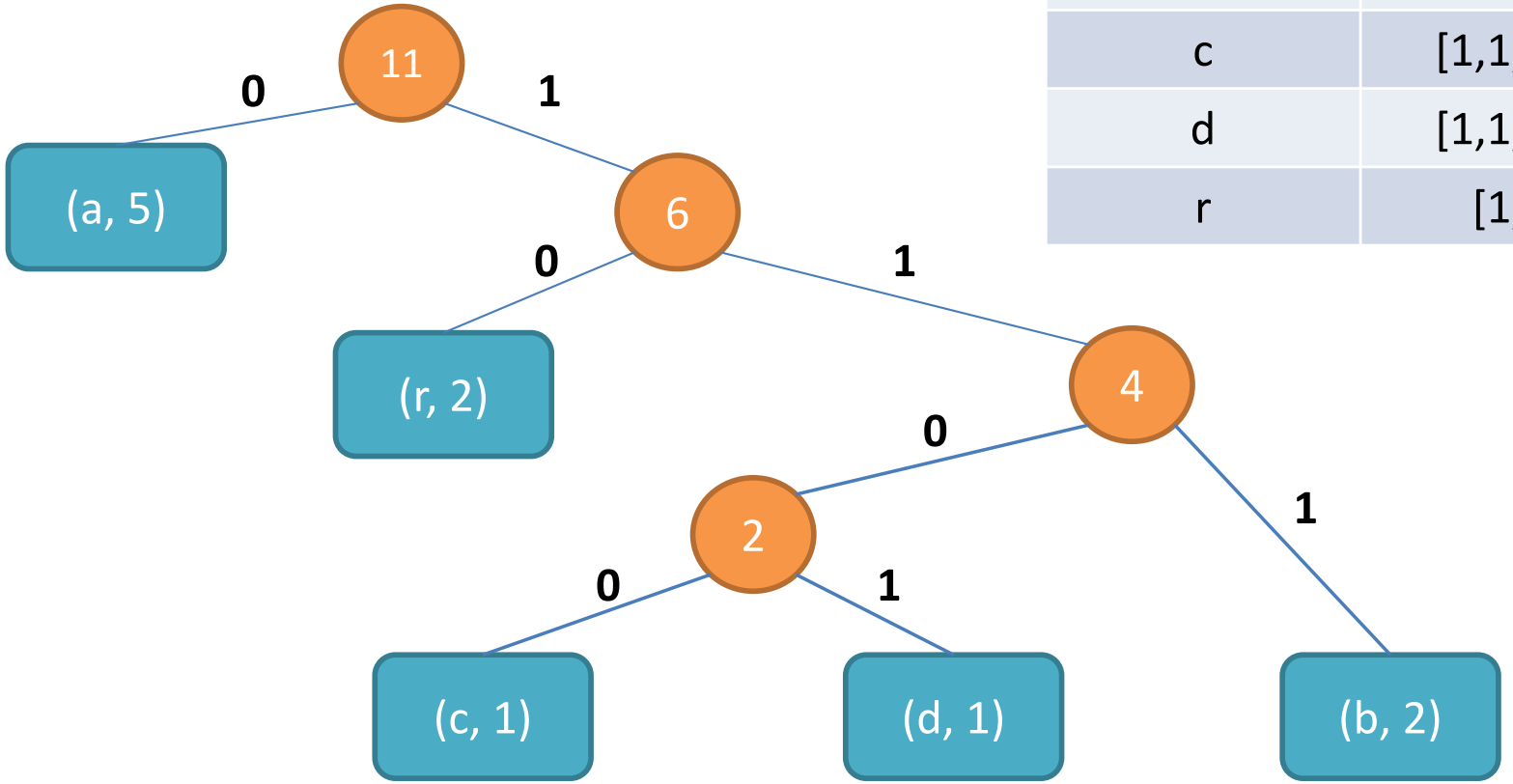
Carácter	Código
a	[1]
b	[0,0,1]
c	[0,0,0,0]
d	[0,0,0,1]
r	[0,1]

Lo representaremos con un diccionario



Otro código para “abracadabra”

Carácter	Código
a	[0]
b	[1,1,1]
c	[1,1,0,0]
d	[1,1,0,1]
r	[1,0]



Codificación Huffman

- “abracadabra” en ASCII de 8 bits ocupa
  11 caracteres * 8 bits = **88 bits**
- “abracadabra” en Huffman ocupa **23 bits**

```
[0,1,1,1,1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]
 a     b   r  a     c     a     d     a   b     r  a
```

Reemplazar cada carácter por su código

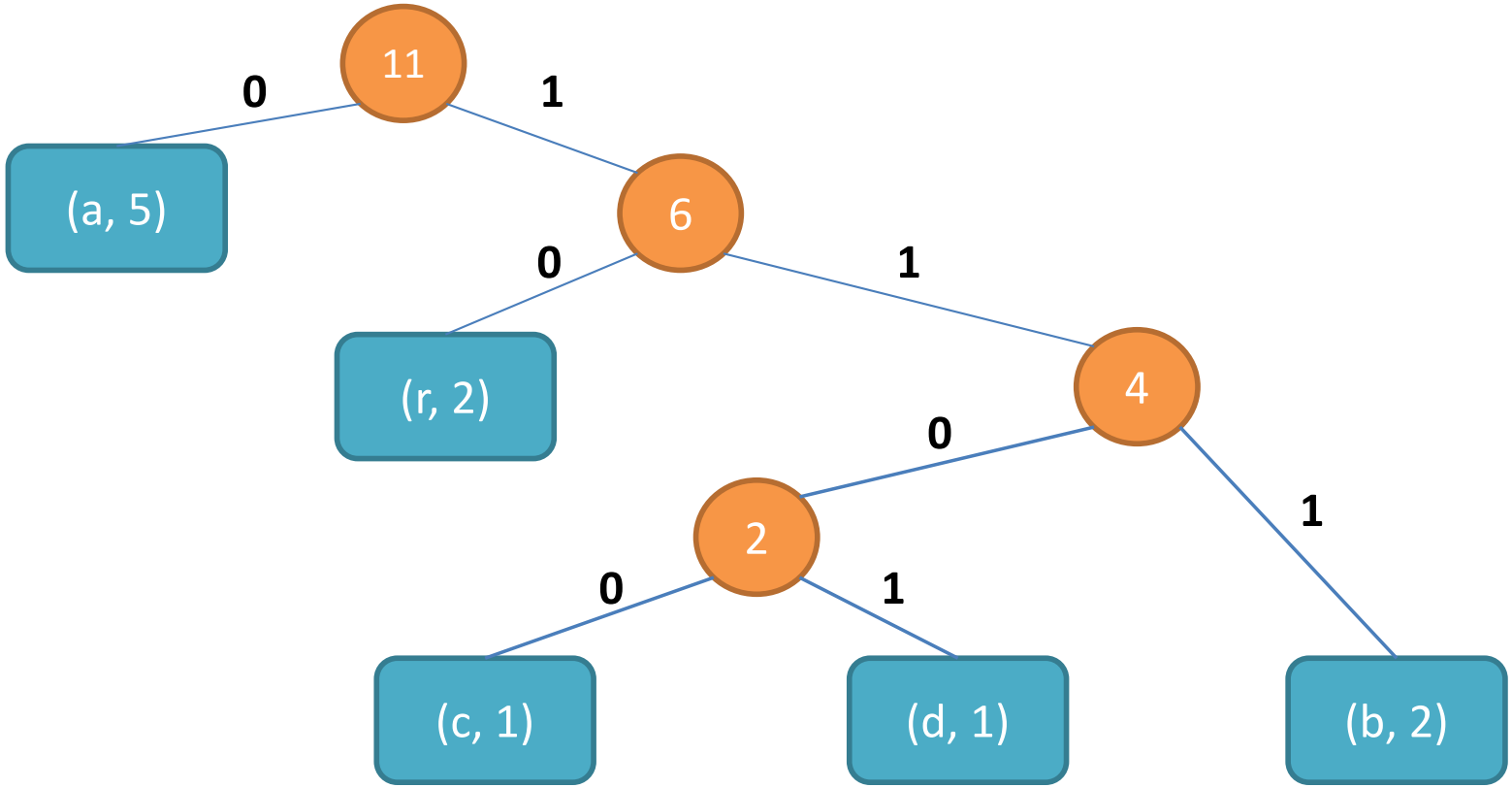
Carácter	Código
a	[0]
b	[1,1,1]
c	[1,1,0,0]
d	[1,1,0,1]
r	[1,0]

Decodificación Huffman

- Se puede usar un diccionario inverso; pero es más simple y eficiente utilizar el árbol de Huffman
- Para decodificar un carácter se recorre el mensaje codificado bit a bit, descendiendo desde la raíz del árbol por izquierda o derecha
- Al llegar a una hoja se ha decodificado el carácter; se repite el proceso para el resto del mensaje

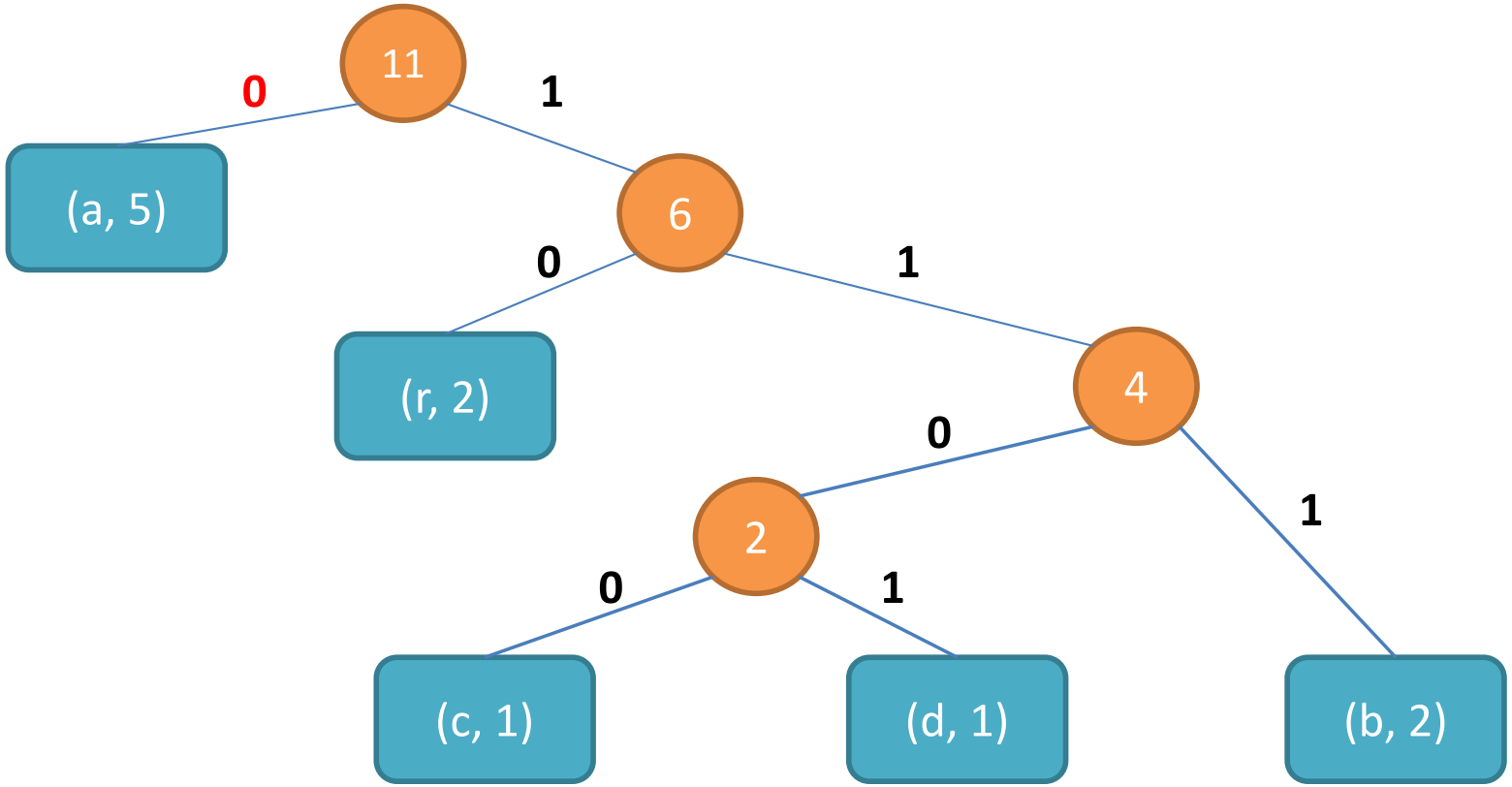
Decodificación (I)

[0,1,1,1,1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]



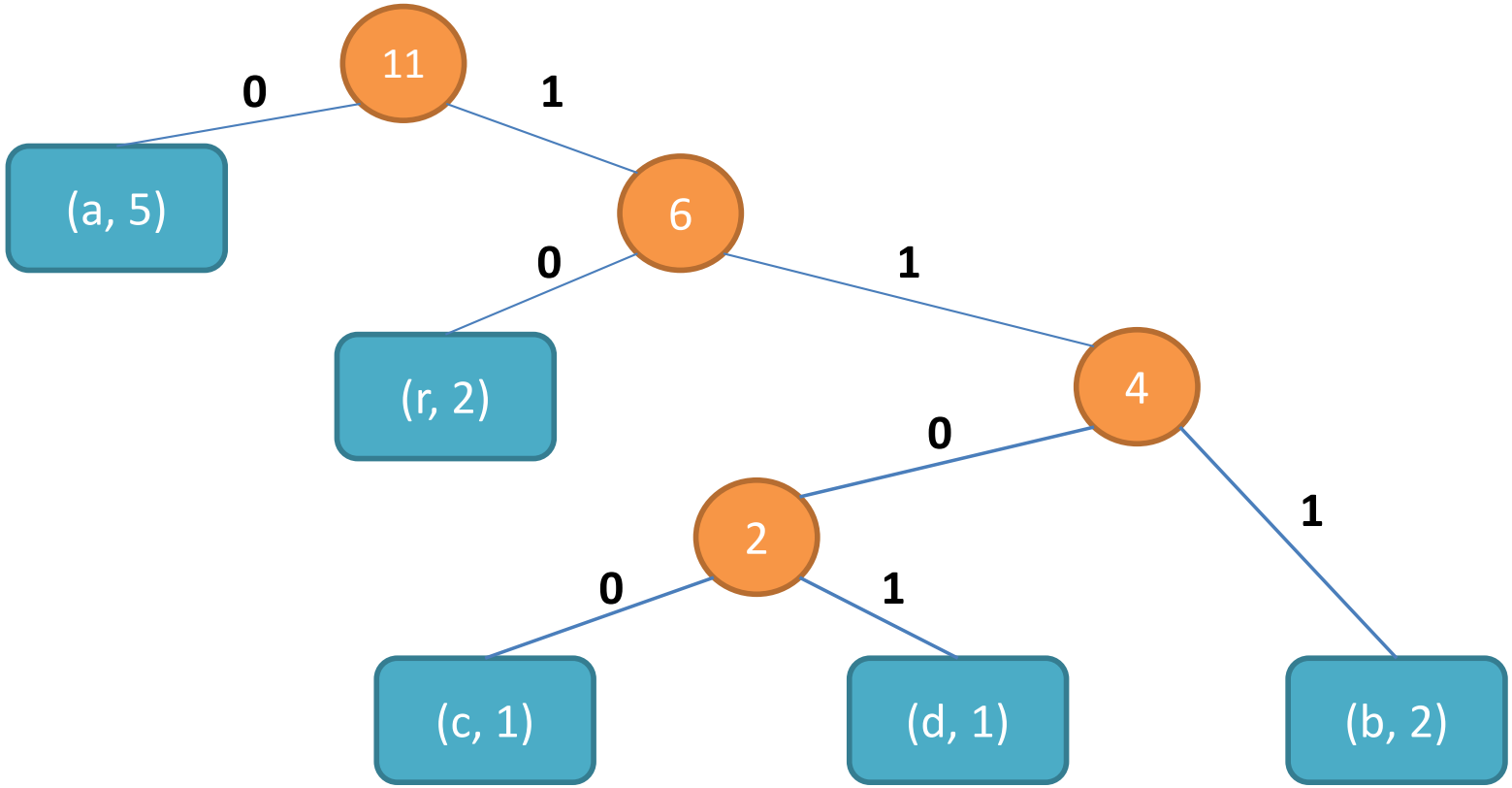
Decodificación (II)

[0,1,1,1,1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]
a



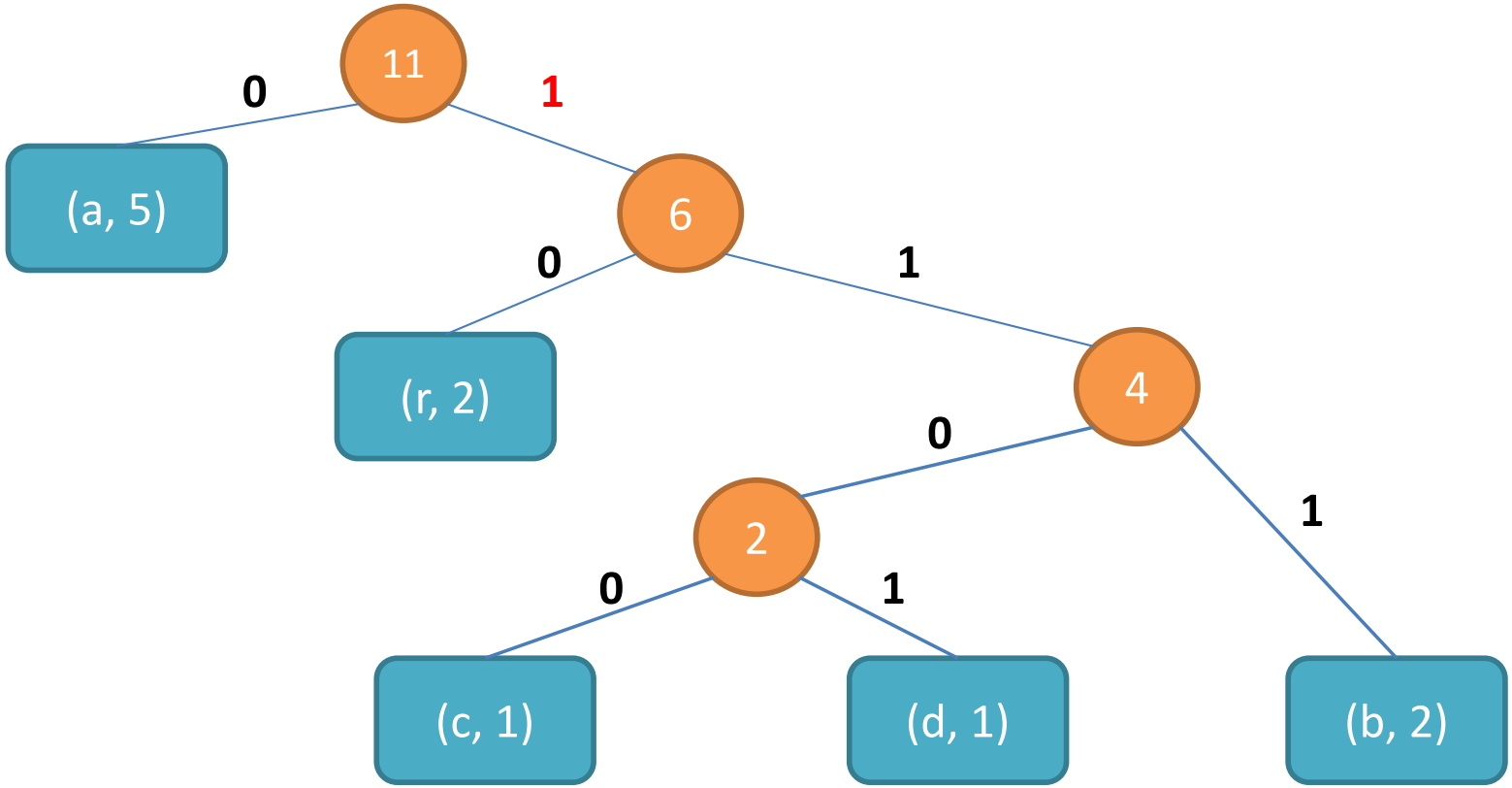
Decodificación (III)

[1,1,1,1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]



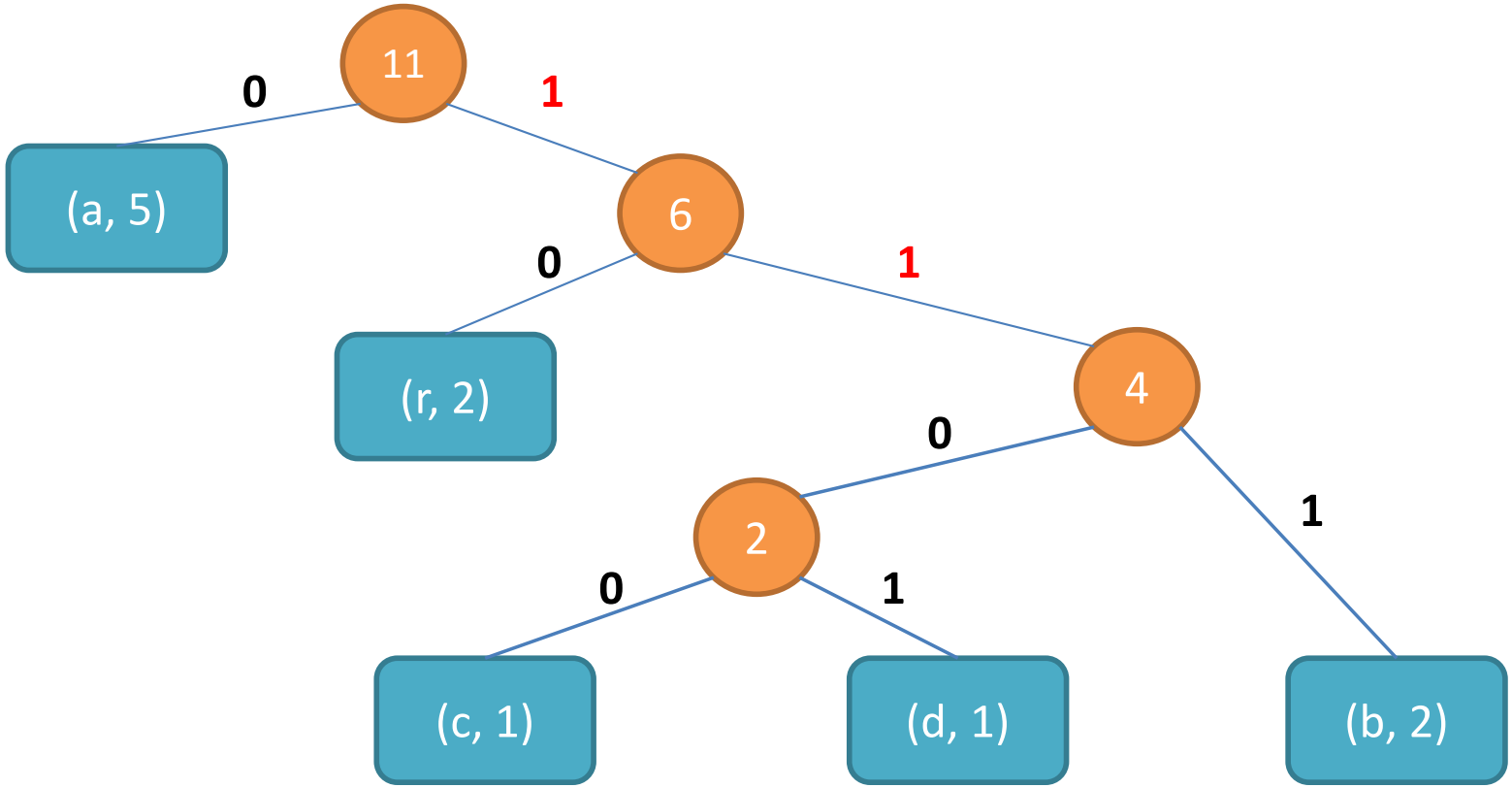
Decodificación (IV)

[1,1,1,1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]



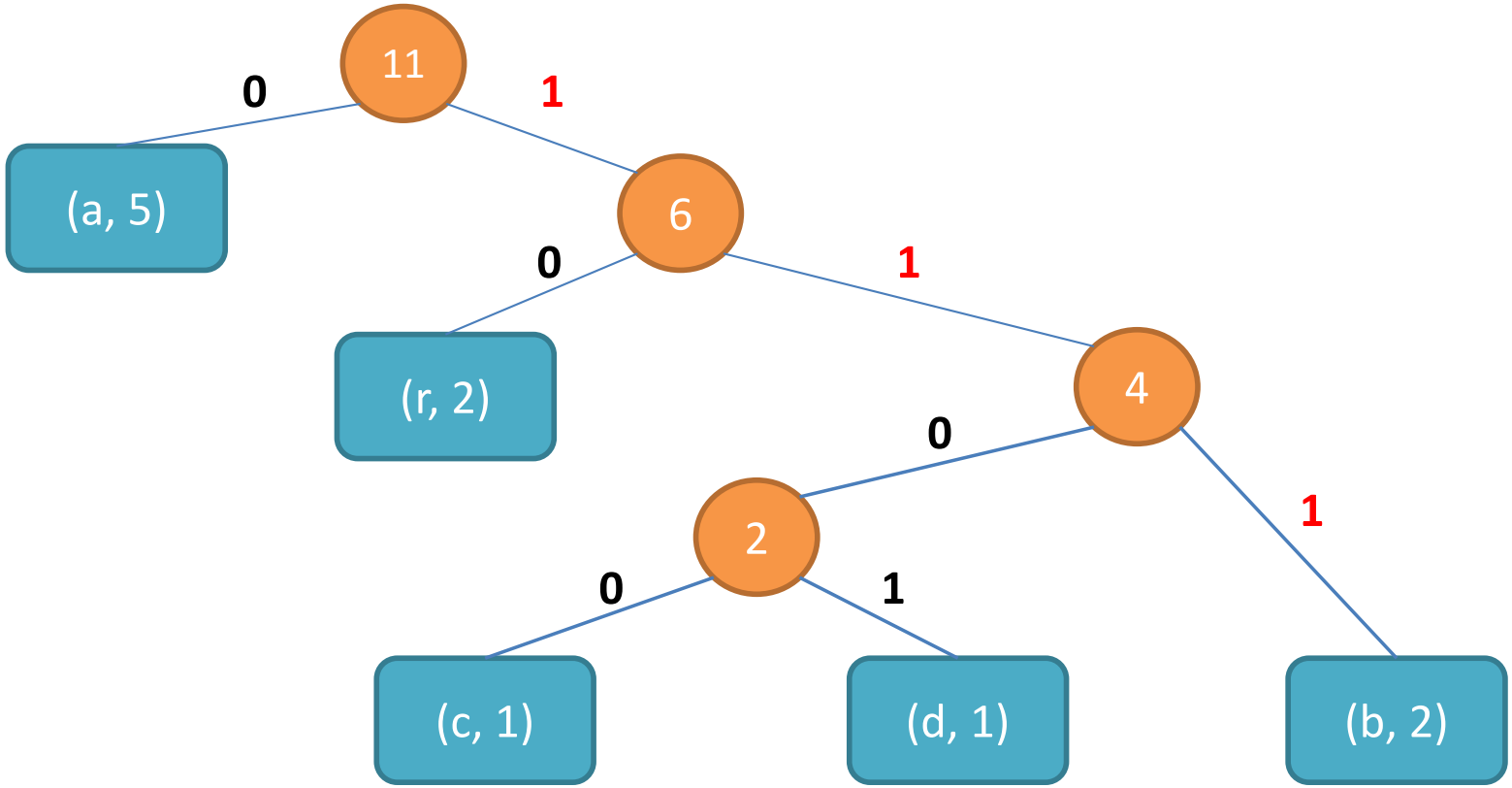
Decodificación (V)

[1,1,1,1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]



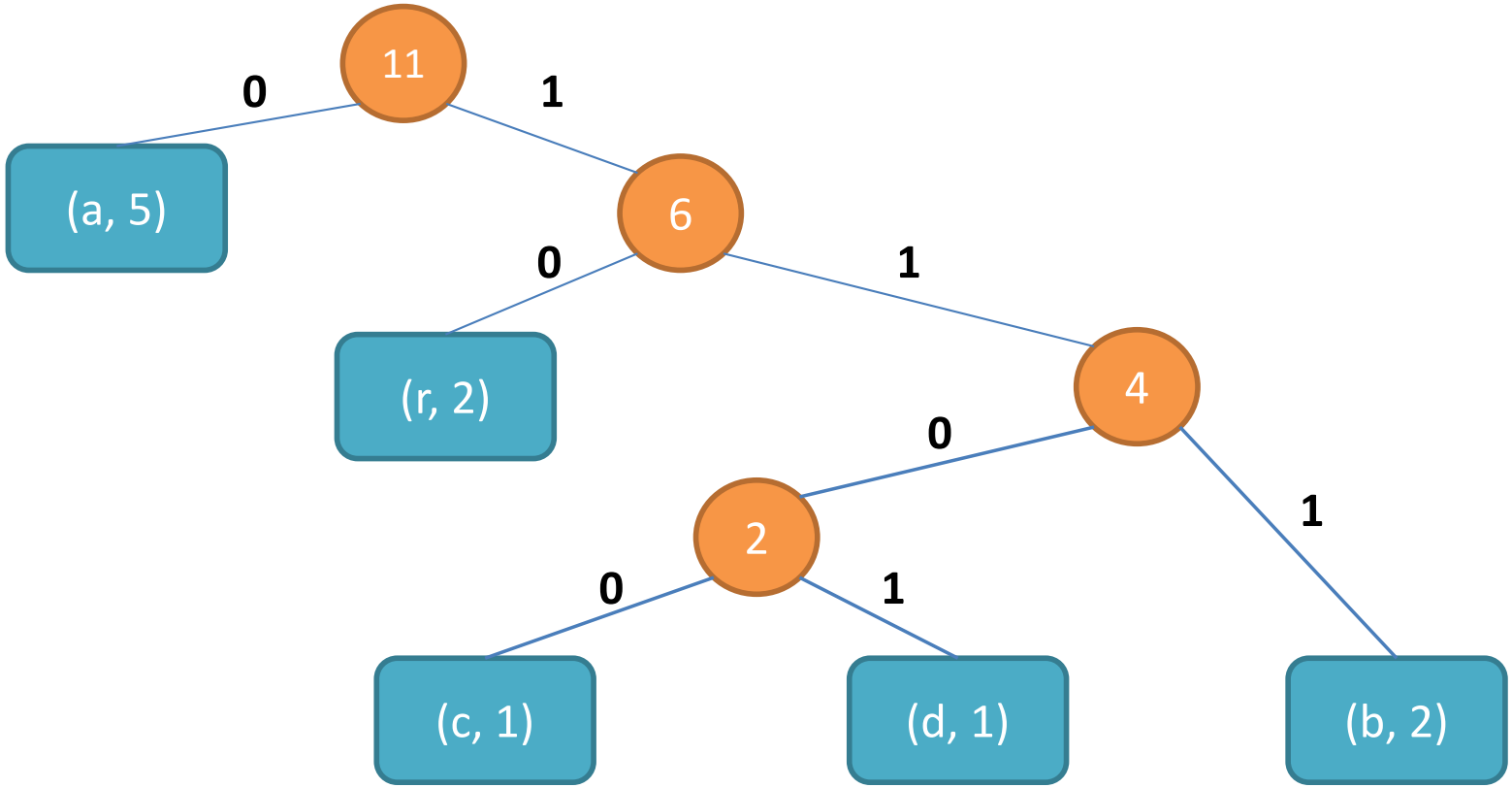
Decodificación (VI)

[1,1,1,1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]
b



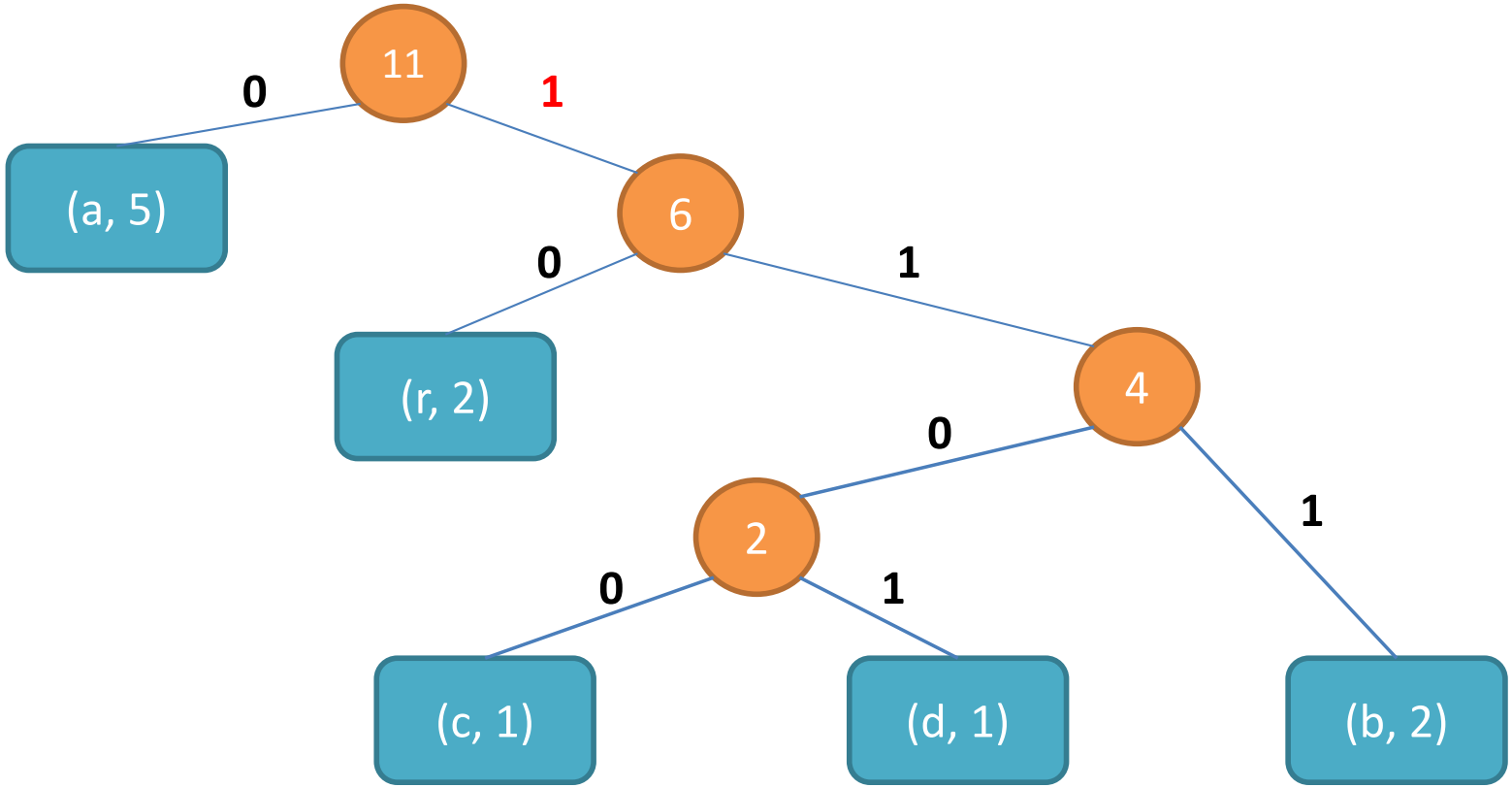
Decodificación (VII)

[1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]



Decodificación (VIII)

[1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]



Decodificación (y IX)

[1,0,0,1,1,0,0,0,1,1,0,1,0,1,1,1,1,0,0]
r