Prácticas
Caso de estudio 4
Backtracking

A. Niño - R. Sobrinos Universidad de Castilla-La Mancha

Presentación del problema (I)

Montse Creto es una estudiante de Grado que planea usar sus conocimientos de algorítmica para seleccionar las becas de colaboración que va a solicitar durante el curso académico. Antes de comenzar el curso, su Universidad publica un listado con todas las becas de colaboración que se ofrecen. El listado incluye los meses durante los que cada beca estará activa y el salario mensual correspondiente. Montse quiere planificar su curso y seleccionar qué becas tiene que solicitar para maximizar el beneficio total. Montse es una alumna excepcional, con un expediente sobresaliente, así que puede confiar en que conseguirá todas las becas que solicite. La normativa de la Universidad establece que no se pueden simultanear becas, que una vez concedida no es posible rechazar una beca, y que es necesario cubrir el periodo completo de la misma. En estas condiciones, y dado el siguiente listado de becas de colaboración,

Presentación del problema (II)

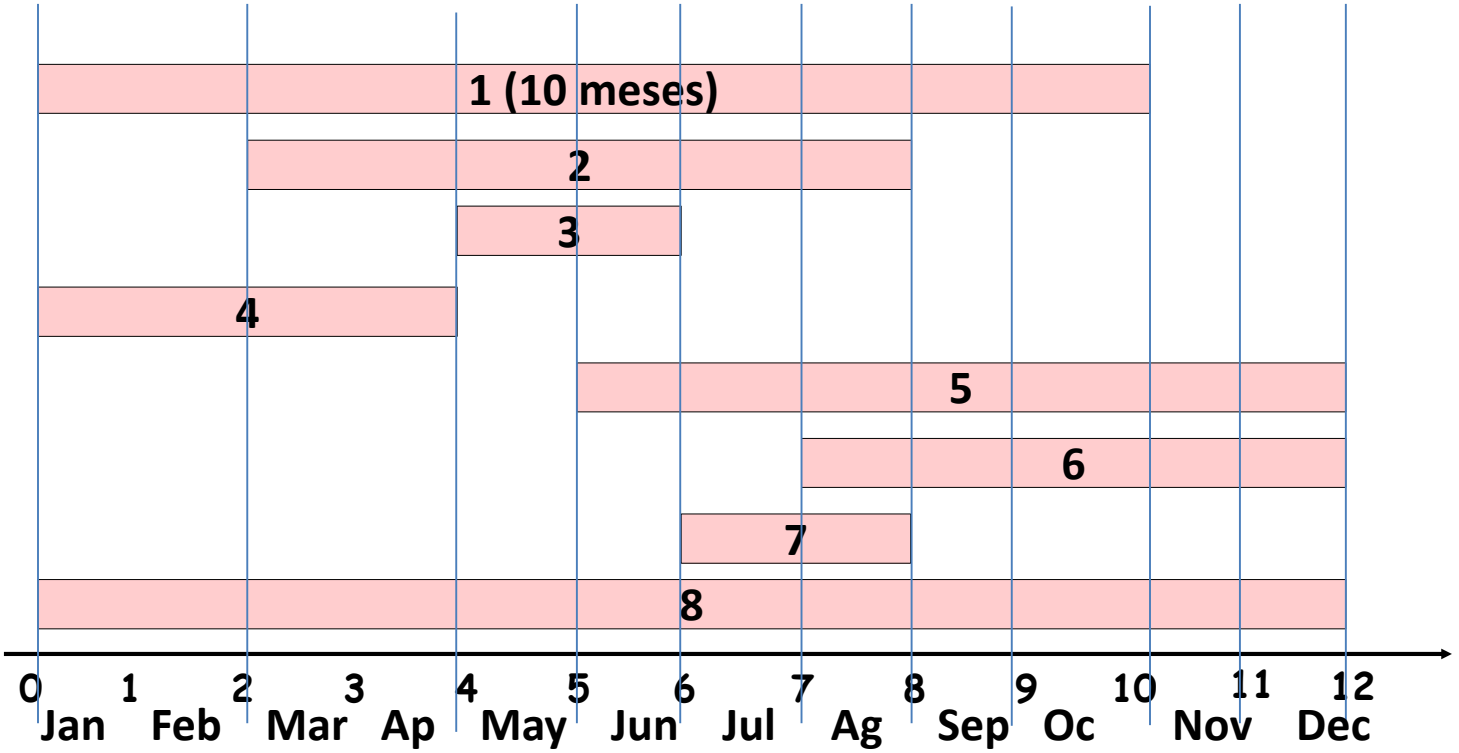
Beca	Mes de comienzo	Mes de finalización	Salario mensual (∈)
1	1	10	150
2	3	8	100
3	5	6	100
4	1	4	300
5	6	12	200
6	8	12	400
7	7	8	500
8	1	12	200

¿Cuál es la combinación óptima de becas que puede realizar Montse?

La cazadora de becas (I)

Visualización del problema

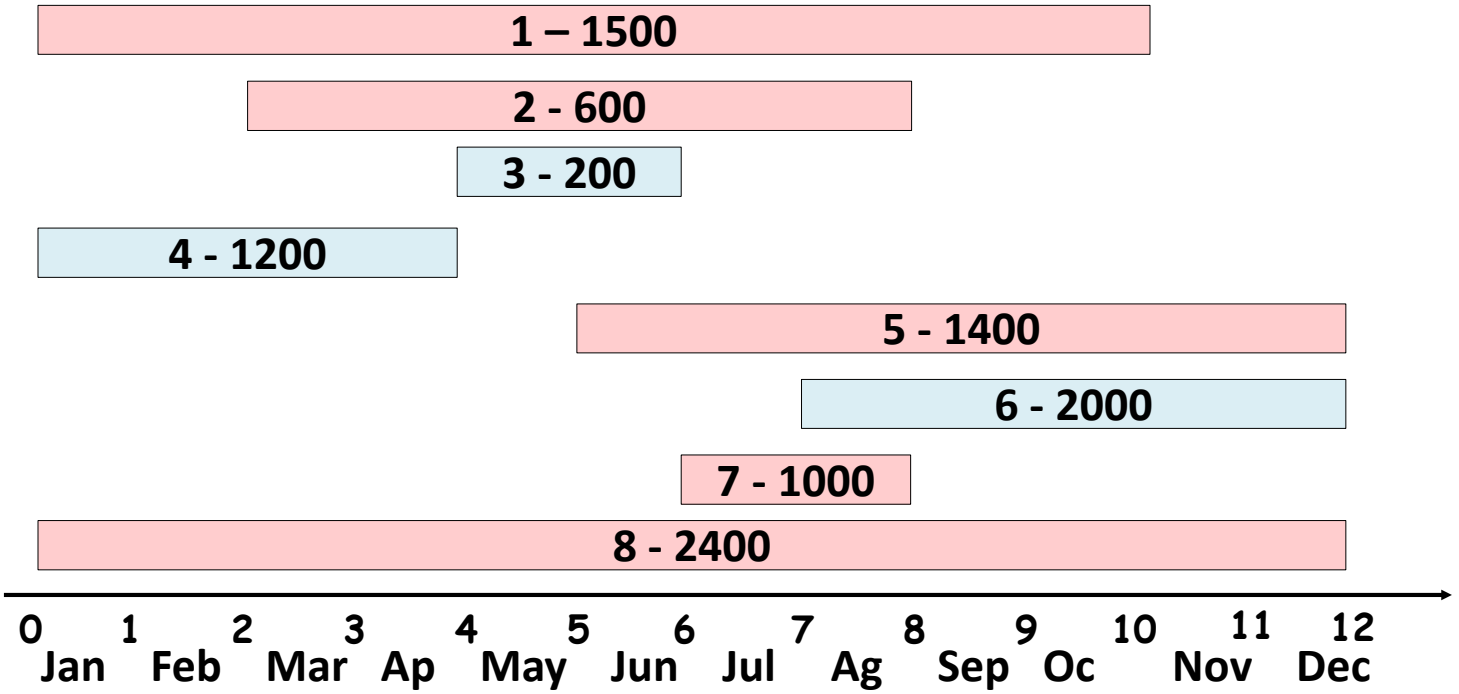
- Distribución de becas (observe que los números del eje indican el fin de un mes)



La cazadora de becas (II)

Visualización del problema

- **Selección óptima (3400 euros)**



La cazadora de becas (III)

- **Punto clave: Probar todas las posibles soluciones, guardando la mejor, descartando tan pronto como sea posible líneas de búsqueda infructuosas.**

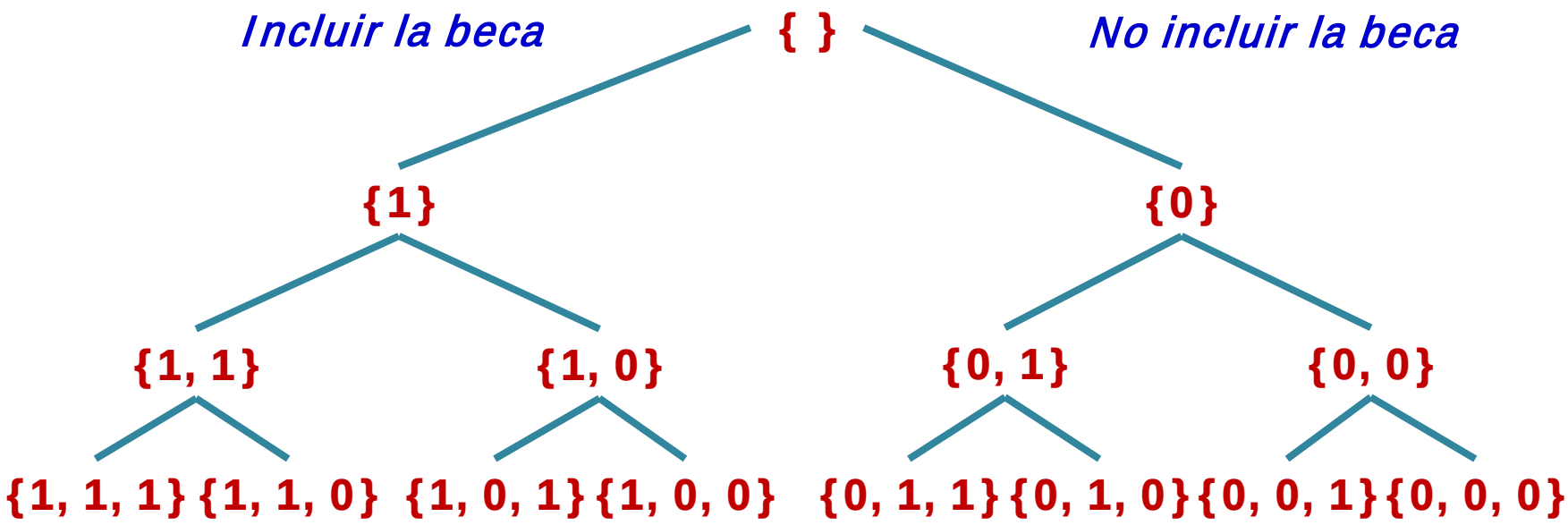
- **Para seleccionar la mejor solución hay que determinar los euros totales para cada conjunto de becas.**

- **Para incluir una beca debemos comprobar su compatibilidad con las anteriores.**

- **Vamos a representar la solución con la típica tupla (1 si la beca está incluida y 0 si no).**

La cazadora de becas (IV)

- **Árbol del espacio de estados (para tres becas)**

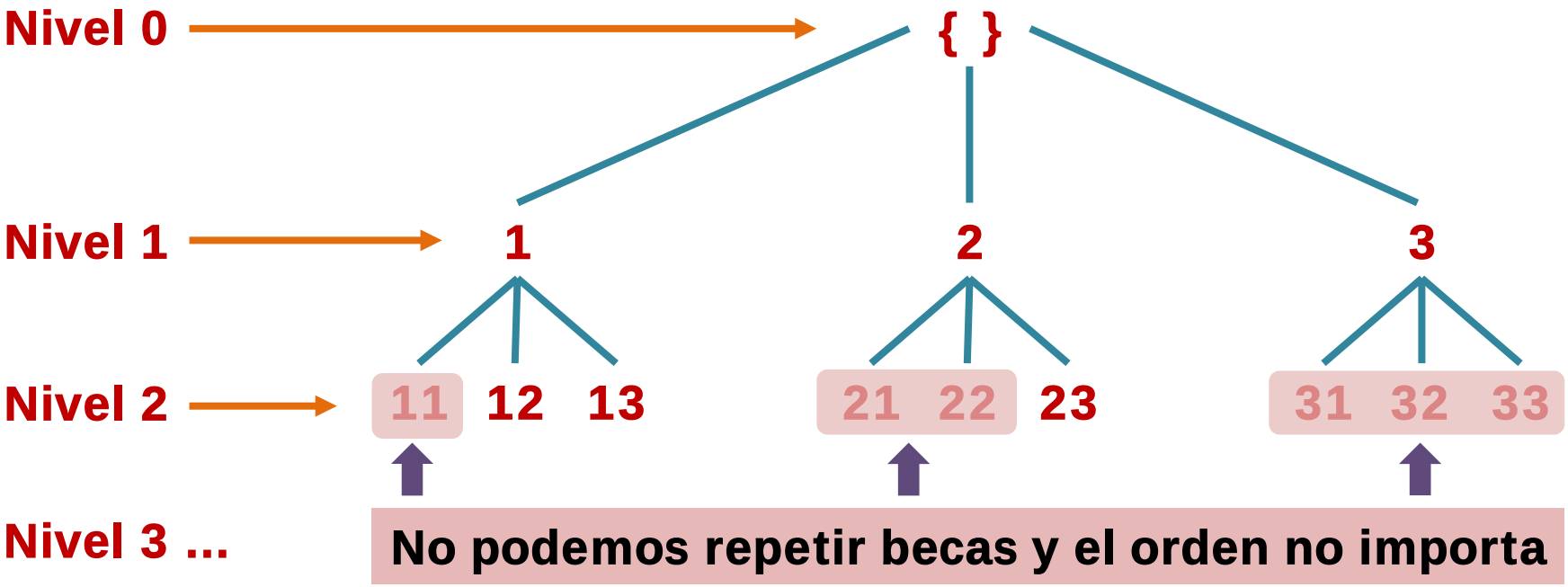


- **Cada nodo es una combinación (parcial) de becas**

- **Cada nivel representa una nueva beca añadida (o no) a la solución**

La cazadora de becas (V)

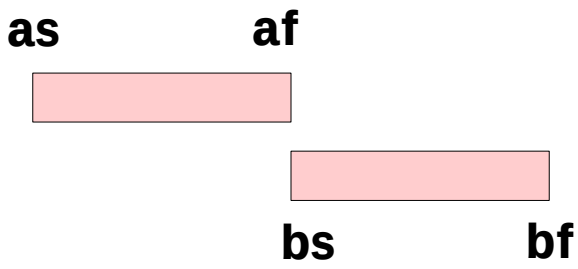
- **Segunda versión. Árbol del espacio de estados (para tres becas)**



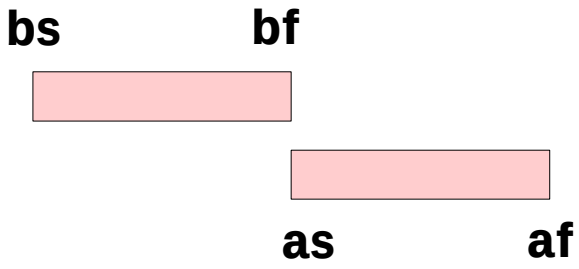
- **Consideramos todas las becas en cada nivel**
- **Cada nivel implica una o más becas en la solución**

La cazadora de becas (VI)

- **Como ver si dos becas son compatibles**



af ≤ bs



bf ≤ as

La cazadora de becas (VII)

Sugerencias de implementación:

- **Implementar una clase Beca que contenga toda la información relativa a ella (mes de inicio, mes de fin, importe, etc).**
- **Usar un array de objetos de tipo Beca para almacenar todas las becas.**
- **Se proporciona un fichero con los datos de las becas. El primer elemento (primera fila) indica el número de becas, y el resto indica, por este orden: id de la beca, mes de inicio, mes de fin, salario mensual. Los valores están separados por un tabulador.**

La cazadora de becas (VIII)

Sugerencias de implementación (cont.):

- Cargar inicialmente el array de becas con el fichero usando la clase Scanner de Java.

```
read = new Scanner(new File(doc));
int lon = read.nextInt();
…
String str=read.nextLine();
String[] datos=str.split("\t");
…
```

- Se puede optar por usar variables de instancia que sean accesibles al método de backtracking para almacenar tanto el mejor beneficio obtenido hasta ese momento como la combinación de becas que lo genera, así como cualquier otro elemento que consideréis.