Ejercicio: Vuelta atrás

Elegir el esquema algorítmico de vuelta atrás para la resolución del problema de encontrar el subconjunto del conjunto de números

$$\{20, 10, 2, 8, 6, 18, 4\}$$

que sumen 32, con el menor número de sumandos.

a) Explicar qué deben verificar las funciones `Criterio()` y `Solución()`.

b) Describir los pasos que ejecutaría el algoritmo una vez encontrada la solución `(0,0,1,1,0,1,1)`, mostrando la salida proporcionada por las funciones evaluadas, hasta llegar de nuevo a evaluar la función `Solución()`.

Ejercicio: Vuelta atrás

Elegir el esquema algorítmico de vuelta atrás para la resolución del problema de encontrar el subconjunto del conjunto de números

{20, 10, 2, 8, 6, 18, 4}

que sumen 32, con el menor número de sumandos.

Caso c) Problema de optimización (maximización).

Quiero minimizar el nº de sumandos

voa: valor óptimo actual
soa: solución óptima actual

voa=10 (mayor que cualquier valor posible)

Sólo almaceno la mejor solución encontrada hasta el momento

Valor(s)<voa

```
Backtracking (var s: TuplaSolución)
nivel:= 1
s:= sINICIAL
voa:= -∞; soa:= ∅
repetir
  Generar (nivel, s)
     si Solución (nivel, s) AND Valor(s)>voa entonces
     voa:=Valor(s); soa:=s
     si Criterio (nivel, s) AND (nivel<n) entonces
        nivel:= nivel + 1
     mientras NOT MasHermanos (nivel, s) AND (nivel>0)
          hacer Retroceder (nivel, s)
hasta nivel==0
```

Ejercicio: Vuelta atrás

Encontrar el subconjunto del conjunto de números {20, 10, 2, 8, 6, 18, 4} que sumen 32, con el menor número de sumandos.

a) Explicar qué deben verificar las funciones `Criterio()` y `Solución()`.

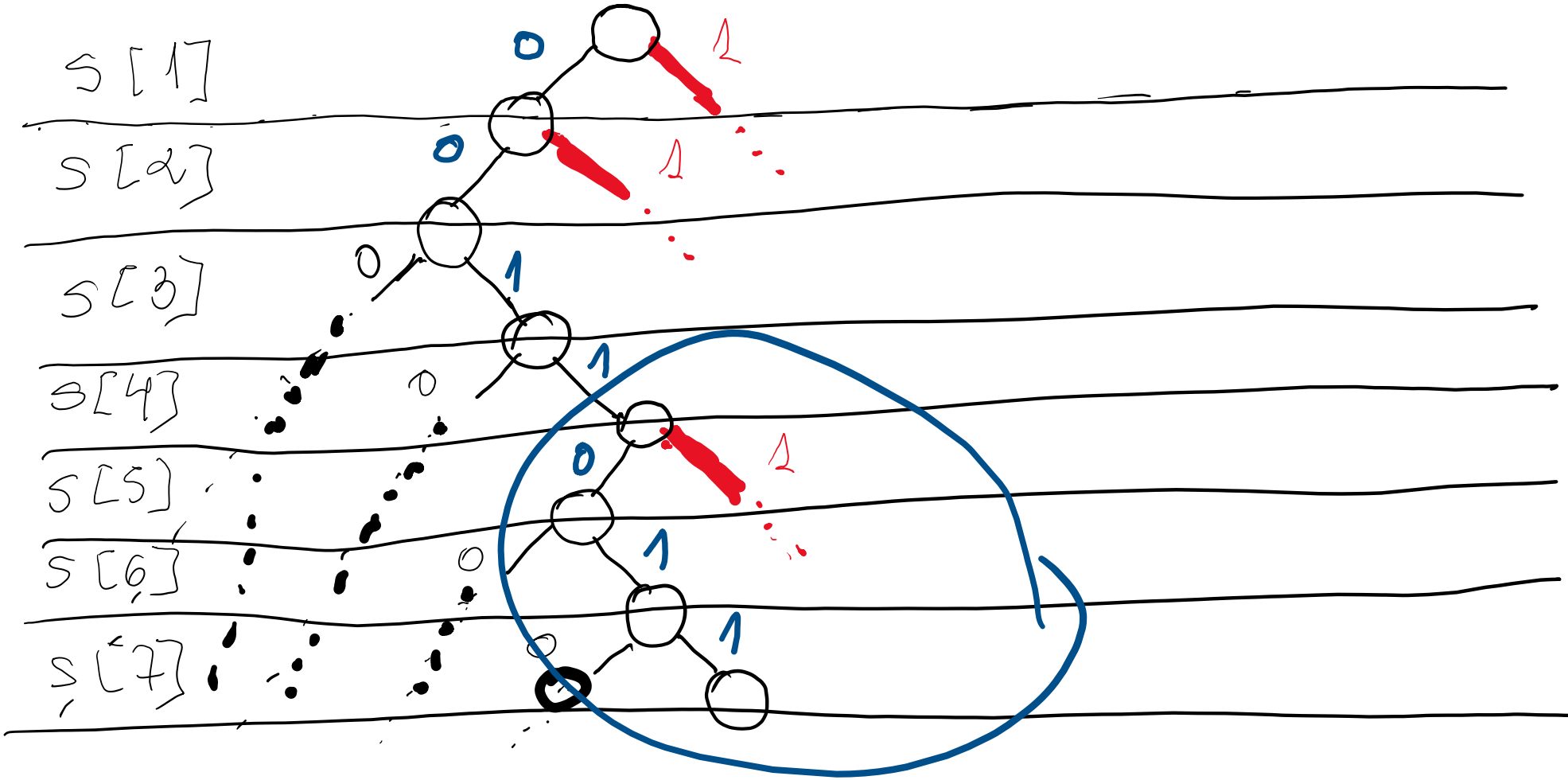
La función `Criterio()` debe comprobar que los términos elegidos no sumen más de 32 (pues en ese caso ya no podrán formar parte de una solución al problema).

La función `Solución()` debe comprobar que el nivel en el que nos encontramos sea el último. Tenemos la posibilidad de decidir sobre 7 candidatos, luego el tamaño del vector solución es 7, y éste será el último nivel. Además, debe comprobar que la suma de los términos elegidos es exactamente 32.

Ejercicio: Vuelta atrás

Subconjunto de {20, 10, 2, 8, 6, 18, 4} que sumen 32, con el menor número de sumandos.

b) Describir los pasos que ejecutaría el algoritmo una vez encontrada la solución (0,0,1,1,0,1,1), mostrando la salida proporcionada por las funciones evaluadas, hasta llegar de nuevo a evaluar la función Solución.



Ejercicio: Vuelta atrás

Subconjunto de {20, 10, 2, 8, 6, 18, 4} que sumen 32, con el menor número de sumandos.

b) Describir los pasos que ejecutaría el algoritmo una vez encontrada la solución (0,0,1,1,0,1,1), mostrando la salida proporcionada por las funciones evaluadas, hasta llegar de nuevo a evaluar la función Solución.

Caso c) Problema de optimización (maximización).

Quiero minimizar el nº de sumandos

voa: valor óptimo actual
soa: solución óptima actual

voa=10 (mayor que cualquier valor posible)

Sólo almaceno la mejor solución encontrada hasta el momento

Valor(s)<voa

```
Backtracking (var s: TuplaSolución)
nivel:= 1
s:= sINICIAL
voa:= -∞; soa:= ∅
repetir
  Generar (nivel, s)
   si Solución (nivel, s) AND Valor(s)>voa entonces
   voa:=Valor(s); soa:=s
   si Criterio (nivel, s) AND (nivel<n) entonces
      nivel:= nivel + 1
   mientras NOT MasHermanos (nivel, s) AND (nivel>0)
        hacer Retroceder (nivel, s)
hasta nivel==0
```

Como es un problema binario, ya que los elementos del conjunto solución son 0 ó 1 (el elemento no está o sí está en la solución), asumo que

sINICIAL=(-1,-1,-1,-1,-1,-1,-1)

Solución(nivel,s): s=(0,0,1,1,0,1,1)→ es solución pues nivel es 7 (el último elemento tiene valor asignado distinto de -1). Además, con los elementos seleccionados, la suma es 2+8+18+4=32

Valor(s) es el número de sumandos, en este caso 4. Tengo que comprobar que 4<voa
voa inicialmente lo había puesto a un valor muy grande (10). Como esta es la primera solución encontrada (podéis comprobar que cualquier combinación binaria menor no es solución), se cumple que Valor(s)<voa
Por tanto, como se cumplen las dos condiciones: voa=4 y soa=(0,0,1,1,0,1,1).

…

Ejercicio: Vuelta atrás

Subconjunto de {20, 10, 2, 8, 6, 18, 4} que sumen 32, con el menor número de sumandos.

b) Describir los pasos que ejecutaría el algoritmo una vez encontrada la solución (0,0,1,1,0,1,1), mostrando la salida proporcionada por las funciones evaluadas, hasta llegar de nuevo a evaluar la función Solución.

Caso c) Problema de optimización (maximización).

Quiero minimizar el nº de sumandos

```
Backtracking (var s: TuplaSolución)
nivel:= 1
s:= sINICIAL
voa:= -∞; soa:= ∅
repetir
  Generar (nivel, s)
     si Solución (nivel, s) AND Valor(s)>voa entonces
     voa:=Valor(s); soa:=s
     si Criterio (nivel, s) AND (nivel<n) entonces
        nivel:= nivel + 1
     mientras NOT MasHermanos (nivel, s) AND (nivel>0)
          hacer Retroceder (nivel, s)
hasta nivel==0
```

voa: valor óptimo actual
soa: solución óptima actual

voa=10 (mayor que cualquier valor posible)

Sólo almaceno la mejor solución encontrada hasta el momento

Valor(s)<voa

…

A continuación evalúo Criterio(7,s), que es cierto, pero nivel no es menor que n (es igual), por lo que no incremento el nivel.

MasHermanos(7,s) es falso (el valor en el nivel 7 es 1, luego ya he probado 0 y 1 pues se prueban en orden, y no quedan más opciones). nivel es mayor que 0, así que retrocedo.

Retroceder(7,s) implica deshacer la última asignación: s=(0,0,1,1,0,1,-1), lo que implica actualizar la suma parcial: suma=32-4=28. Y restar 1 al nivel→nivel=6

…

Ejercicio: Vuelta atrás

Subconjunto de {20, 10, 2, 8, 6, 18, 4} que sumen 32, con el menor número de sumandos.

b) Describir los pasos que ejecutaría el algoritmo una vez encontrada la solución (0,0,1,1,0,1,1), mostrando la salida proporcionada por las funciones evaluadas, hasta llegar de nuevo a evaluar la función Solución.

Caso c) Problema de optimización (maximización).

Quiero minimizar el nº de sumandos

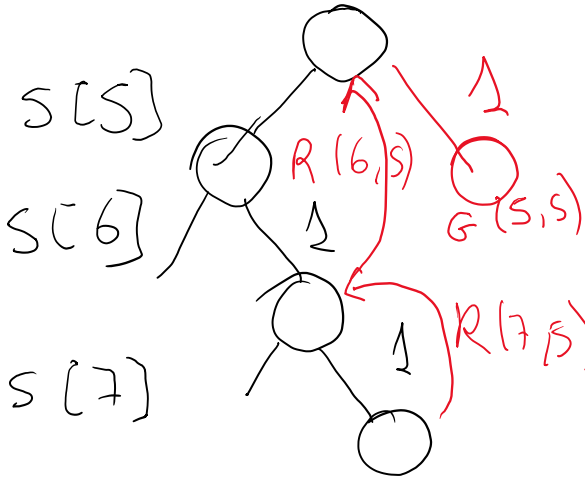
```
Backtracking (var s: TuplaSolución)
nivel:= 1
s:= sINICIAL
voa:= -∞; soa:= ∅
repetir
   Generar (nivel, s)
      si Solución (nivel, s) AND Valor(s)>voa entonces
      voa:=Valor(s); soa:=s
      si Criterio (nivel, s) AND (nivel<n) entonces
         nivel:= nivel + 1
      mientras NOT MasHermanos (nivel, s) AND (nivel>0)
           hacer Retroceder (nivel, s)
hasta nivel==0
```

voa: valor óptimo actual
soa: solución óptima actual

voa=10 (mayor que cualquier valor posible)

Sólo almaceno la mejor solución encontrada hasta el momento

Valor(s)<voa



...

De nuevo evalúo MasHermanos(6,s). Como en la posición 6 el valor también es 1, no hay más hermanos y, al ser nivel positivo, puedo retroceder:

Retroceder(6,s)=>s=(0,0,1,1,0,-1,-1), suma=28-18=10 (pongo a -1 la posición 6 y resto el elemento del conjunto que corresponde) y nivel pasa a valer nivel=5

MasHermanos(5,s) ahora es verdadero, pues en nivel=5 la solución es 0. Aún puedo probar otro valor, el 1. Así que no retrocedo más, salgo del mientras y voy a Generar.

Generar(5,s) produce una solución parcial: s=(0,0,1,1,1,-1,-1). La suma ahora es suma=10+6=16. Y evalúo de nuevo la función solución:

Solución(5,s)=falso ya que no estoy en el último nivel y además suma tampoco es 32.

Ejercicio: Vuelta atrás

Subconjunto de {20, 10, 2, 8, 6, 18, 4} que sumen 32, con el menor número de sumandos.

b) Describir los pasos que ejecutaría el algoritmo una vez encontrada la solución (0,0,1,1,0,1,1), mostrando la salida proporcionada por las funciones evaluadas, hasta llegar de nuevo a evaluar la función Solución.

Caso c) Problema de optimización (maximización).

Quiero minimizar el nº de sumandos

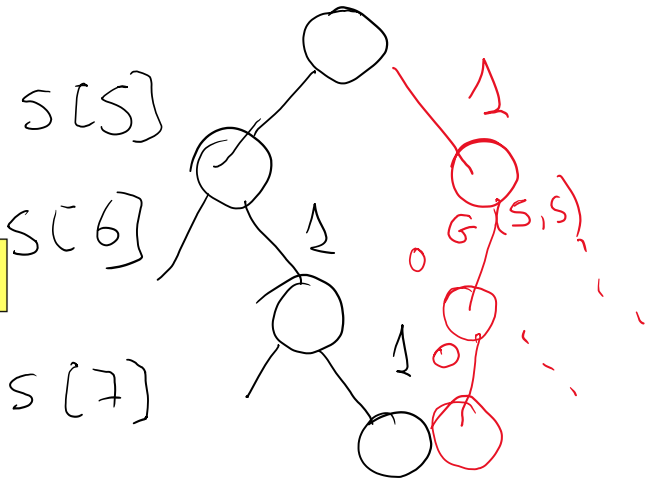
voa: valor óptimo actual
soa: solución óptima actual

voa=10 (mayor que cualquier valor posible)

Sólo almaceno la mejor solución encontrada hasta el momento

Valor(s)<voa

```
Backtracking (var s: TuplaSolución)
nivel:= 1
s:= sINICIAL
voa:= ∞; soa:= ∅
repetir
  Generar (nivel, s)
    si Solución (nivel, s) AND Valor(s)>voa entonces
    voa:=Valor(s); soa:=s
    si Criterio (nivel, s) AND (nivel<n) entonces
       nivel:= nivel + 1
    mientras NOT MasHermanos (nivel, s) AND (nivel>0)
         hacer Retroceder (nivel, s)
hasta nivel==0
```



...

Criterio(5,s) =TRUE y nivel<n→ nivel=6- De nuevo evalúo MasHermanos(6,s). Como en la posición 6 no se ha probado ningún valor, MasHermanos es TRUE.

Generar(6,s) produce una solución parcial: s=(0,0,1,1,1,0,-1). La suma ahora es suma=16+0. Y evalúo de nuevo la función solución:

Solución(6,s)=falso ya que no estoy en el último nivel y además suma tampoco es 32.

Criterio(6,s)=TRUE y nivel<n→nivel=7. De nuevo evalúo MasHermanos(7,s). Como en la posición 7 no se ha probado ningún valor, MasHermanos es TRUE.

Generar(7,s) produce una solución parcial: s=(0,0,1,1,1,0,0). La suma ahora es suma=16+0. Y evalúo de nuevo la función solución...