UNIVERSIDAD CATÓLICA
SAN ANTONIO
UCAM

ESCUELA UNIVERSITARIA POLITÉCNICA
Departamento de Ciencias Politécnicas
Grado en Ingeniería Informática

Prácticas Algoritmia
Boletín 2. Backtracking

Curso 2017/2018

Profesor: Andrés Muñoz

Boletín 2. Backtracking

1. Objetivo

El objetivo de este boletín es que los alumnos analicen, diseñen, codifiquen y evalúen algoritmos de backtracking.

2. Ejercicios

1. **(5 puntos)** El problema de la suma de subconjuntos consiste en que dado un conjunto de valores enteros positivos $V=\{v_1, v_2, v_3, \ldots, v_n\} \mid v_i > 0$, se deben encontrar todos los subconjuntos S_c de V cuyos valores sumen una cantidad C fijada de antemano (como el problema de encontrar todos los subconjuntos de {10,14,6} que sumen 20 como el visto en las diapositivas de clase).

   Por otro lado, la diferencia absoluta máxima de un subconjunto se define como el valor absoluto más alto que resulta de las restas de los elementos del subconjunto tomados de dos en dos. Por ejemplo, para el subconjunto {14, 25, 27} tenemos las restas (en valor absoluto):
   - |14-25| = 11
   - |14-27| = 13
   - |25-27| = 2

   siendo 13 la diferencia absoluta máxima.

   Una variante del problema de la suma de subconjuntos consiste en encontrar, de entre los subconjuntos S_c que suman el valor C, aquel que tenga la menor diferencia absoluta máxima. A continuación, se muestran dos ejemplos:

   - **Ejemplo 1**. V={1, 9, 10, 12, 30}, C = 31
     - Subconjuntos candidatos: S_1= {1, 30} y S_2= {9, 10, 12}
     - Diferencia absoluta máxima(S1) = 29
     - Diferencia absoluta máxima(S2) = 3
     - **Solución: S_2**

   - **Ejemplo 2**. V={21, 15, 20, 9, 10, 25}, C = 50
     - Subconjuntos candidatos: S_1= {21, 20, 9} y S_2= {15, 10, 25}
     - Diferencia absoluta máxima(S1) = 12
     - Diferencia absoluta máxima(S2) = 15
     - **Solución: S_1**

   Se pide diseñar e implementar un algoritmo que mediante la técnica de Backtracking permita, de forma eficiente en la medida posible, encontrar el subconjunto S_c con la menor diferencia absoluta máxima. Para ello el usuario debe poder introducir por teclado, y en este orden:
   - El número de elementos del conjunto V
   - Los valores $v_1, v_2, v_3, \ldots$ del conjunto V, $v_i > 0$
   - La cantidad C que deben sumar los subconjuntos, C > 0

2. **(5 puntos)** El alumno Aníbal Goritmo ha recibido el siguiente mensaje codificado de su archienemiga Efi Ciencia a través de este código numérico:

   6531484901984

   Nuestro alumno sabe que cada dígito del código se corresponde con una letra, pero no sabe cuál. Para poder descodificarlo necesita la tabla de conversión entre letras y dígitos. Para obtener dicha tabla de conversión ha recibido una pista de Malcom Plejidad en la que hay resolver la siguiente suma:

```
   H A R R Y
+  P O T T E R
   T R O L L S
```

   sabiendo que:

   - Cada letra corresponde a una cifra entre 0 y 9.
   - Cada letra es una única cifra para todo el problema. Por ejemplo, si la letra "R" fuese "3" este dato valdría para todas las "R" de las palabras HARRY, POTTER y TROLLS, y además el "3" del mensaje cifrado se sustituiría por la letra "R".
   - Cuando las letras de la suma se sustituyen por su valor, la operación aritmética debe ser correcta. Es decir, no es posible que "Y" valga 1, "R" valga 2 y "S" valga 8.
   - Los números que aparecen en la suma no pueden empezar por 0.
   - Hay que tener en cuenta el acarreo de la suma (son sumas "llevando"). Por ejemplo, dada la siguiente suma:

```
     B
+  V R
   O A
```

     donde B=6, R=5 y V = 2, entonces

     - A=1 (¡ojo!, A no vale 11)
     - O=3 (se suma el acarreo a V=2)

   - Sólo existe una solución al problema.

   Se pide diseñar e implementar un algoritmo que mediante la técnica de Backtracking permita que Aníbal Goritmo, de forma eficiente en la medida posible, encuentre la tabla de conversión entre letras y dígitos necesaria para poder descodificar el mensaje código.

3. Entregables y puntuación

- Memoria que contenga la información pedida en los ejercicios del boletín, junto a las decisiones más importantes tomadas para resolver cada ejercicio (NO INCLUIR EL CÓDIGO DE LOS PROGRAMAS. SIN MEMORIA NO SE CORREGIRÁN LAS PRÁCTICAS)

  Nota importante. Para ambos ejercicios se pide:

  i. Indicar y justificar la representación de la solución elegida.
  ii. Mostrar de manera esquemática el árbol de búsqueda que se va a generar.
  iii. Codificar el esquema del algoritmo, indicando qué esquema se ha utilizado y por qué, documentando los pasos más relevantes y explicando las funciones auxiliares utilizadas.
  iv. Calcular el número de nodos teóricos que se deben generar para cada problema y comparar con el número de nodos reales devueltos en la ejecución del código. ¿Hay diferencias? ¿Por qué?

- **Un proyecto de Dev-C++ que incluya los ficheros .c con el código de los programas y que esté listo para ser compilado y ejecutado** (NO ENTREGAD LOS FICHEROS DE CADA EJERCICIO POR SEPARADO ***O NO SE CORREGIRÁN LAS PRÁCTICAS***).

- **Puntuación:** La nota máxima del boletín son 10 puntos (se deben obtener mínimo 4 puntos para aprobar el boletín).
  - El 30% de la puntuación de cada ejercicio corresponderá a su documentación en la memoria que explique las decisiones más relevantes y al estilo de programación, teniendo en cuenta:
    - Comentarios adecuados en cantidad y calidad.
    - Tabulación correcta del código.
    - Uso de estructuras dinámicas en vez de estáticas, evitar el uso de variables globales, correcto paso de parámetros, etc.

- **Fecha de entrega: 17 de Diciembre 2017**