Técnicas de Programación Avanzadas (TPA)

Universidad Europea
LAUREATE INTERNATIONAL UNIVERSITIES

Tema 6. Avance Rápido

Tema 6. Avance Rápido

1. Planteamiento general
2. Ejemplos
   1. Cambio mínimo
   2. Mochila
   3. Grafos

Técnicas de Programación Avanzadas (TPA)

Planteamiento general

- Se trata de un nuevo paradigma o estrategia de resolución de problemas.
- Se utiliza especialmente en problemas de optimización.
- Los algoritmos de avance rápido o voraces buscan seleccionar la opción más beneficiosa en cada paso, hasta alcanzar la solución.

Universidad Europea

Técnicas de Programación Avanzadas (TPA)

Planteamiento general

- Supongamos que tenemos:
  1. Un conjunto de datos de entrada
  2. Una función objetivo (a maximizar o minimizar).
- **Objetivo:** encontrar un subconjunto de datos que:

Técnicas de Programación Avanzadas (TPA)

Planteamiento general

- Debemos distinguir entre:
  - **Solución factible:** cualquier subconjunto que a partir del conjunto de datos de entrada satisfaga las restricciones del problema.
  - **Solución óptima:** aquella solución factible para la que la función objetivo devuelve un valor óptimo.

Universidad Europea

Técnicas de Programación Avanzadas (TPA)

Planteamiento general

- La estrategia es la siguiente:

1) Comenzar con una solución "vacía"
2) Repetir hasta alcanzar la solución:
   a) Seleccionar el elemento **X** más prometedor de los disponibles.
   b) Si la solución temporal continuase siendo factible en caso de incluir **X**, se añade

Planteamiento general

- Seudocódigo

```
Función AvanceRapido( C: conjunto; in-out Sol:
  Conjunto)

  Sol = ∅
  mientras (C ≠ ∅) y esSolución(Sol)==falso hacer
    x = seleccionarElemento (C)
    C = C – {x}
    si (esFactible ( Sol U {x} ) entonces
       Sol = Sol U {x}
  si esSolución(Sol) entonces devolver Sol
  si no devolver <error>
```


Planteamiento general

- La **complejidad** dependerá de varios factores:
  - La función de selección del elemento
  - El orden inicial de los elementos de entrada
- En general es más rápido que otros enfoques siempre que sea aplicable (excepto tal vez DyV)

Tema 6. Avance Rápido

1. *Planteamiento general*
2. **Ejemplos**
   1. **Cambio mínimo**
   2. **Mochila**
   3. **Grafos**

Ejemplos

Devolución del cambio mínimo

- Dados los siguientes datos de entrada:
  - M un sistema monetario con monedas de valores $\{V_1, V_2, \ldots, V_n\}$,
  - C, la cantidad que se quiere devolver
- Se busca devolver dicha cantidad C utilizando el menor número posible de monedas

Técnicas de Programación Avanzadas (TPA)

Ejemplos

Devolución del cambio mínimo

- **Ejemplo:**
  - Monedas: {2, 1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01}
  - Cantidad C = 5,89 €
  - Solución factible:
    589 monedas de 0,01 Euros.
  - Solución factible:
    5 x 1€ 39 x 0,01€
    1 x 0,50€
  - <u>Solución óptima:</u>
    2 x 2€ 1 x 0,50€ 1 x 0,10€ 2 x 0,02€
    1 x 1€ 1 x 0,20€ 1 x 0,05€

Universidad Europea

Técnicas de Programación Avanzadas (TPA)

Ejemplos

Devolución del cambio mínimo

- **Ejemplo:**
  - Monedas: {2, 1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01}
  - Cantidad: 5,89 €

Representación de la solución:

Un vector con el número de monedas seleccionadas de cada tipo, identificadas por posición.

Ejemplos

Devolución del cambio mínimo

- **Ejemplo:**
  - Monedas: {2, 1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01}
  - Cantidad: 5,89 €

Objetivo

Minimizar $\sum_{i=1}^{n} c_i$

teniendo en cuenta que $\sum_{i=1}^{n} c_i \cdot v_i = Cantidad$, con n>=0

Ejemplos

Devolución del cambio mínimo

```
Función DevolverCambio (M: vector [1..n] real; C
real): vector [1..n] int

    Sol = ∅   /*conjunto vacío*/
    cant = 0  /*cero*/
    mientras (cant != C) hacer
        x = seleccionar (C, cant, M)
        si x = 0 //no hay valor adecuado de moneda
        entonces devolver ∅ /*tenemos un error*/
```

Ejemplos

Devolución del cambio mínimo

- El algoritmo se basa en la función **`seleccionar`**, que devuelve el valor V_i más alto de moneda, contando con que no supere la cantidad que falte por devolver:

$$V_i + cant <= C$$

- Devuelve error cuando no hay solución posible.
- **Ejemplo:**

    M = {2, 4, 6}

    C = 5

Técnicas de Programación Avanzadas (TPA)

Ejemplos

Devolución del cambio mínimo

- ¿Garantiza la estrategia de Avance Rápido (AR) siempre la solución óptima?
- **Ejemplo:**

    M = { 100, 90, 1}

    C = 180

    Solución A.R. → Sol = [1, 0, 80] → 81 monedas

Tema 6. Avance Rápido

1. *Planteamiento general*
2. *Ejemplos*
   1. *Cambio mínimo*
   2. **Mochila**
   3. **Grafos**

Ejemplos

Problema de la Mochila

- **Problema:**
  - Tenemos n objetos: x_1, x_2, …, x_n
  - Cada objeto x_i tiene un peso p_i $(1 \leq i \leq n)$
  - Cada objeto x_i tiene asociado un beneficio b_i
- **Objetivo:**
  - Llenar una mochila maximizando el beneficio de los objetos seleccionados, y respetando siempre su

Técnicas de Programación Avanzadas (TPA)

Ejemplos
Problema de la Mochila

- **Solución:**
  - Un vector en el que en cada posición se almacena un número entre 0 y 1.
    - 0 indica que no se selecciona el elemento
    - 1 indica que se ha seleccionado el 100% de ese elemento
    - 0.4, por ejemplo, indica que se ha seleccionado el 40%

Universidad Europea

Técnicas de Programación Avanzadas (TPA)

Ejemplos
Problema de la Mochila

- **Formalmente:**

maximizar $f(X) = \Sigma\,(b_i \cdot x_i)$

sujeto a $\Sigma\,(p_i \cdot x_i) \leq M$

con $0 \leq x_i \leq 1$

Ejemplos

Problema de la Mochila

- **Condiciones:**
  - Sólo interesan las situaciones en las que P·1 > M.
    - Es decir, cuando el peso de todos los objetos supera la capacidad de la mochila.
  - Toda solución óptima cumplirá que P·X = M

Técnicas de Programación Avanzadas (TPA)

Ejemplos

Problema de la Mochila

- **Ejemplo:**

  M = 20

  B = [25, 24, 15]; P = [18, 15, 10]

  Algunas soluciones factibles:

  S1 = [1, 2/15, 0]; f(X) = 28,42

  S2 = [0, 2/3, 1]; f(X) = 31

Técnicas de Programación Avanzadas (TPA)

Ejemplos
Problema de la Mochila

- **Enfoques:**
  A. Seleccionar el elemento con mayor beneficio
  B. Tomar el elemento con menor peso
  C. Tomar los elementos por orden de beneficio/peso

- ¿Cuál será la mejor opción?
- ¿Se garantiza la solución óptima?

Universidad Europea

Técnicas de Programación Avanzadas (TPA)

Ejemplos
Problema de la Mochila

- **Enfoques:**

  Ejemplo 1:

  N = 4; M = 10; P = [10, 3, 3, 4]; B = [10, 9, 9, 9]

  Opción A: Sol = [1, 0, 0, 0] → 10 (Beneficio máx.)

  Opción B: Sol = [0, 1, 1, 1] → 27 (Peso máx.)

Técnicas de Programación Avanzadas (TPA)

Ejemplos

Problema de la Mochila

- **Enfoques:**

  Ejemplo 2:

  N = 4; M = 10; P = [10, 3, 3, 4]; B = [10, 1, 1, 1]

  Opción A: Sol = [1, 0, 0, 0] → 10 (Beneficio máx.)
  Opción B: Sol = [0, 1, 1, 1] → 3 (Peso máx.)
  Opción C: Sol = [1, 0, 0, 0] → 10 (Beneficio/Peso máx.)

  La opción C es la que garantiza la solución óptima.

Universidad Europea

Técnicas de Programación Avanzadas (TPA)

Ejemplos

Problema de la Mochila

```
Función mochila (B, P: vector[1..n] de real; M: real):
vector[1..n] de real
    /* supondremos B y P ordenados por b_i/p_i */
    para i desde 1 hasta n hacer
        Sol [i] = 0.0
    capacidad = M
    i = 1
    mientras (i <= n) y (P[i] <= capacidad) hacer
        capacidad = capacidad – P[i]
        Sol[i] = 1
```

Ejemplos

Problema de la Mochila

▪**Complejidad:**

– Pero caso: O(n) si tomamos el "mientras" como referencia

– Mejor caso: O(1)

– Considerando la ordenación previa: O(QuickSort) · n

Técnicas de Programación Avanzadas (TPA)

Tema 6. Avance Rápido

Técnicas de Programación Avanzadas (TPA)

Ejemplos

Grafos. Árbol de recubrimiento mínimo: Kruskal

- **Problema:**
  - Dado un Grafo G (V, A) conexo ponderado y no dirigido, se busca el subgrafo G’ (V, A’) tal que A’ ⊆ A y la suma de las aristas sea mínima.
  - G’ será el árbol de recubrimiento mínimo de G.

Universidad Europea

Técnicas de Programación Avanzadas (TPA)

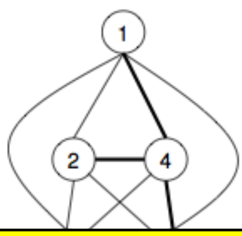
Ejemplos

Grafos. Árbol de recubrimiento mínimo: Kruskal

- **Ejemplo:**

V = {1, 2, 3, 4, 5}

A = [0 7,5 10 7 9,5
7,5 0 3,5 1,5 4
10 3,5 0 4 2,5
7 1,5 4 0 3



Técnicas de Programación Avanzadas (TPA)

Ejemplos

Grafos. Árbol de recubrimiento mínimo: Kruskal

▪**Características de la solución:**

- Si el grafo es conexo, existe solución.
- La solución tendrá n-1 aristas, siendo n el número de vértices del grafo.
  - Es el número mínimo para garantizar la conectividad
  - Más aristas provocarían ciclos

Universidad Europea

Técnicas de Programación Avanzadas (TPA)

Ejemplos

Grafos. Árbol de recubrimiento mínimo: Kruskal

▪**Enfoques de la solución:**

- Algoritmo de Kruskal:
  - Seleccionar en cada paso la arista de menor peso disponible, siempre que no forme un ciclo.
- Algoritmo de Prim:

Técnicas de Programación Avanzadas (TPA)

Ejemplos

Grafos. Árbol de recubrimiento mínimo: Kruskal

▪**Algoritmo de Kruskal:**

- Partir de una solución vacía
- Mientras el número de aristas seleccionadas < n-1, y queden aristas por marcar, hacer:
  - Seleccionar la arista A de menor coste no marcada ya
  - Marcarla
  - Si A no genera ciclo, entonces:
    - añadirla a la solución
  - Si no:
    - descartarla

Universidad Europea

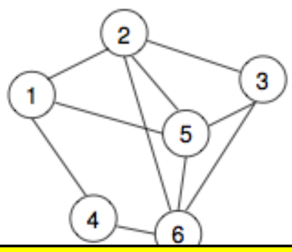
Técnicas de Programación Avanzadas (TPA)

Ejemplos

Grafos. Árbol de recubrimiento mínimo: Kruskal

▪**Ejemplo:**

$$A = \begin{pmatrix} 0 & 10 & \infty & 30 & 45 & \infty \\ 10 & 0 & 50 & \infty & 40 & 25 \\ \infty & 50 & 0 & \infty & 35 & 15 \\ 30 & \infty & \infty & 0 & \infty & 20 \\ 45 & 40 & 35 & \infty & 0 & 55 \\ \infty & 25 & 15 & 20 & 55 & 0 \end{pmatrix}$$



Ejemplos

Grafos. Árbol de recubrimiento mínimo: Kruskal

- **Ejemplo:**

ARISTA	COSTE	COMPONENTES CONEXAS
		(1), (2), (3), (4), (5), (6)
(1, 2)	10	(1, 2), (3), (4), (5), (6)
(3, 6)	15	(1, 2), (3, 6), (4), (5)
(4, 6)	20	(1, 2), (3, 4, 6), (5)
(2, 6)	25	(1, 2, 3, 4, 6), (5)
(1, 4)	30	rechazada
(3, 5)	35	(1, 2, 3, 4, 5, 6)

Algorítmica - José María Gómez Hidalgo

Ejemplos

Grafos. Árbol de recubrimiento mínimo: Kruskal

```
función kruskal (g: Grafo): lista de aristas
    Sol = ∅   /*conjunto vacío*/
    resto = g.listaAristas()
    mientras (Sol.longitud()<n-1) y (resto≠∅) hacer
        a = seleccionarAristaMinima (resto)
        resto = resto – {a}
        si (hayCiclos(Sol, a) = falso) /*no hay ciclos*/
```

Universidad Europea

LAUREATE INTERNATIONAL UNIVERSITIES

Madrid Valencia Canarias