



Unidad Didáctica II.
Algoritmos de recorridos de estructuras lineales y
no lineales
Tema 4. Algoritmos de búsqueda

Algoritmia

Profesor: Andrés Muñoz

Escuela Politécnica

Andrés Muñoz
Universidad Católica San Antonio de Murcia - Tlf: (+34) 968 27 88 00 info@ucam.edu - www.ucam.edu

UCAM | UNIVERSIDAD CATÓLICA
SAN ANTONIO

Índice

- ✓ *Introducción*
- ✓ *Búsqueda lineal*
- ✓ *Búsqueda binaria*
- ✓ *Árboles de búsqueda*
- ✓ *Patrones de búsqueda*

Índice

- ✓ *Introducción*
- ✓ *Búsqueda lineal*
- ✓ *Búsqueda binaria*
- ✓ *Árboles de búsqueda*
- ✓ *Patrones de búsqueda*

Introducción

- ✓ Definición de búsqueda:
 - Elementos utilizados:
 - Estructura de datos ED que contiene elementos de tipo T.
 - Un valor constante 'c' (clave) incluido en el tipo T.
 - Definición: encontrar un elemento de la estructura cuyo valor clave sea igual al valor constante.

Introducción

- ✓ A veces es necesario localizar un conjunto de elementos que cumplan una determinada condición.
 - Búsqueda por rango
 - Búsqueda de patrones
- ✓ La estrategia y la eficiencia de la búsqueda dependerán en general de la forma en que se estructuren los datos.

5

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Índice

- ✓ *Introducción*
- ✓ *Búsqueda lineal*
- ✓ *Búsqueda binaria*
- ✓ *Árboles de búsqueda*
- ✓ *Patrones de búsqueda*

6

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda lineal

- ✓ Definición: recorrer y examinar cada uno de los elementos hasta alcanzar el final de la lista de datos.
- ✓ Si se encontrara el elemento buscado se informa de la/las posición/es donde ha sido localizado.
- ✓ ¿Cuándo aplicarla?
 - Cuando los datos se encuentran organizados de forma lineal (listas, colas).
 - No hay información extra que permita una estrategia mejor.

Búsqueda lineal

- ✓ Optimizaciones:
 - Únicamente se desea obtener la posición donde se localiza por primera vez al elemento deseado.
 - La lista se encuentra ordenada (es posible parar sin tener que llegar al final de la lista)

Búsqueda lineal

✓ Ejemplo: Búsqueda lineal en una lista de enteros

```
typedef struct n_lista{
    int contenido;
    struct n_lista *siguiente;
}NodoLista, *PuntNodoLista;

PuntNodoLista encontrar (PuntNodoLista l, int valor){
    PuntNodoLista p;
    p = l;
    while(p!=NULL)
        if (p->contenido == valor)
            return p;
        else
            p = p -> siguiente;
    return p;
}
```

Búsqueda lineal

✓ Ejemplo: Búsqueda lineal en una lista de enteros

- Análisis de complejidad, siendo *n* el número de nodos de la lista

	el nodo existe			el nodo no existe
caso	mejor	peor	medio	
nº comparaciones	1	n	≈ n/2	n

Búsqueda lineal

✓ Ejemplo: Búsqueda lineal en una lista **ordenada** de enteros

```
typedef struct n_lista{
    int contenido;
    struct n_lista *siguiente;
}NodoLista, *PuntNodoLista;

PuntNodoLista encontrar (PuntNodoLista l, int valor){
    PuntNodoLista p;
    p = l;
    while(p!=NULL && p->contenido < valor)
        p = p -> siguiente;
    return ((p!=NULL && p->contenido == valor)?p:NULL;
}
```

Búsqueda lineal

- ✓ Ejemplo: Búsqueda lineal en una lista de **ordenada** de enteros
- Análisis de complejidad, siendo *n* el número de nodos de la lista

caso	el nodo existe			el nodo no existe		
	mejor	peor	medio	mejor	peor	medio
nº comparaciones	1	n	≈ n/2	1	n	≈ n/2

Índice

- ✓ *Introducción*
- ✓ *Búsqueda lineal*
- ✓ *Búsqueda binaria*
- ✓ *Árboles de búsqueda*
- ✓ *Patrones de búsqueda*

13

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda binaria

- ✓ Se parte de la suposición de que el vector está ordenado (ascendente o descendente)
 - Si no lo está, hay que ordenarlo previamente
- ✓ **Idea.** Hasta encontrar el elemento buscado o el subvector de búsqueda no tenga elementos, repetir los siguientes pasos:
 - Se divide el vector en dos partes iguales
 - Si el elemento del centro del vector es **mayor** que el elemento buscado, sólo se tendrá que seguir buscando por la **primera mitad**
 - Si el elemento del centro del vector es **menor** que el elemento buscado, sólo se tendrá que seguir buscando por la **segunda mitad**
 - Si el elemento central es el buscado, se acaba la búsqueda
 - Si el subvector de búsqueda no tiene elementos, devolver que no se ha encontrado el elemento buscado

14

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda binaria

- ✓ Implementación para vector ordenado de enteros (primera versión)

```
int busqueda_binaria_1 (int v[], int nelem, int
valor){
    int superior, inferior, medio;

    superior = nelem - 1;
    inferior = 0;
    while (inferior < superior){
        medio = (inferior + superior) / 2;
        if (valor > v[medio])
            inferior = medio + 1;
        else
            superior = medio;
    }
    return ((v[superior] == valor)?superior:-1);
}
```

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Tema, Asignatura

15

Búsqueda binaria

- ✓ Problema con la primera versión de la implementación:
- No se aprovecha el hecho de que puede suceder que el valor esté en *v[medio]*.
- ✓ Ejercicio: En la sentencia *return* de esta función, ¿podría sustituirse *v[superior] == valor* por *v[inferior] == valor*?
¿Por qué?
- ✓ Ejercicio: ¿Por qué para calcular *inferior* se le suma uno a *medio* tras la comparación y a *superior* no?

Búsqueda binaria

✓ Implementación para vector ordenado de enteros (segunda versión)

```
int busqueda_binaria_2 (int v[], int nelem, int
valor){
    int superior, inferior, medio;

    superior = nelem - 1;
    inferior = 0;
    while (inferior <= superior){
        medio = (inferior + superior) / 2;
        if (valor < v[medio])
            superior = medio - 1;
        else if (valor > v[medio])
            inferior = medio + 1;
        else
            return medio;
    }
    return -1;
}
```

17

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda binaria

✓ Problema con la segunda versión de la implementación:

- Esta segunda versión realiza dos comparaciones en cada iteración mientras que la anterior sólo realiza 1.

18

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda binaria

✓ Complejidad

- Análisis de complejidad, siendo n el número de nodos de la lista

	el nodo existe			el nodo no existe
caso	mejor	peor	medio	
nº comparaciones	1	$\log_2(n)$	$\approx \log_2(n)$	$\log_2(n)$

19

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Índice

- ✓ *Introducción*
- ✓ *Búsqueda lineal*
- ✓ *Búsqueda binaria*
- ✓ *Árboles de búsqueda*
- ✓ *Patrones de búsqueda*

20

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

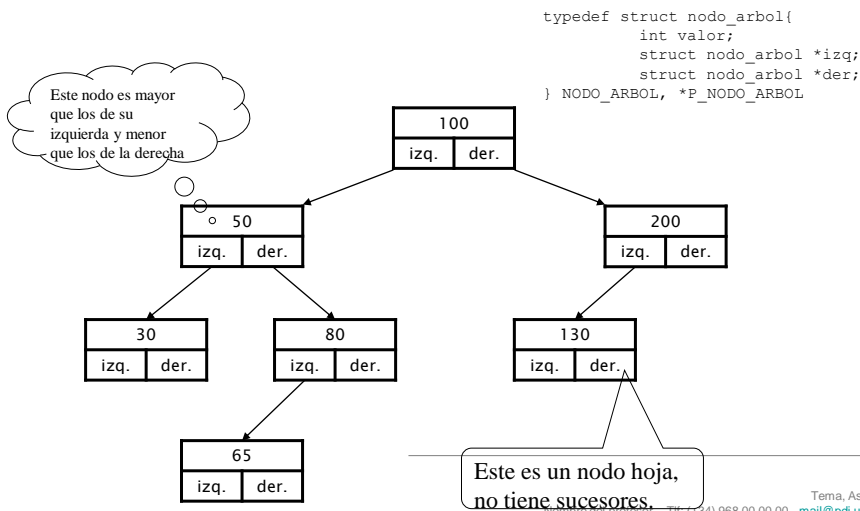
- ✓ Vamos a trabajar con árboles binarios
- ✓ Los árboles binarios de búsqueda (ABB) cumplen las siguientes condiciones para cada nodo N:
 - El valor de la clave de la raíz del subárbol izquierdo es menor que el valor de la clave del nodo N
 - El valor de la clave raíz del subárbol derecho es mayor que el valor de la clave del nodo N.

21

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda



22

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Conceptos

- **Camino** de un nodo n_1 a otro n_k es la secuencia de nodos n_1, n_2, \dots, n_k tal que n_i es el padre de n_{i+1}
- La **profundidad** de un nodo n es la longitud del camino entre la raíz y n .
- Para un ABB, el valor medio de la profundidad es $O(\log n)$, donde n es el número de elementos a insertar
 - Conviene mantener el equilibrio del árbol y no tener árboles desbalanceados
- La **altura** de n es el camino más largo de n a una hoja.
 - La altura de un árbol es la altura de la raíz

23

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Operaciones básicas (repaso)

- **Buscar:** devuelve la posición del nodo con la clave x .
- **Insertar:** coloca la clave x . Si ya estuviese, no se hace nada
 - Se pueden realizar operaciones de contador/actualización.
- **Eliminar:** borra el nodo con clave x .
 - Si x está en una hoja, se elimina de inmediato.
 - Si el nodo tiene un hijo, se ajusta un apuntador del padre del nodo a eliminar al hijo antes de eliminarlo.
 - Si el nodo tiene dos hijos,
 - se sustituye x por la clave mas pequeña del subárbol derecho.
 - A continuación se elimina en el subárbol derecho el nodo con dicha clave más pequeña

24

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Ventajas

- Búsqueda en $O(\log n)$ en caso promedio

✓ Desventajas

- Inserción debe ser ordenada ($O(n)$)
- Se debe cuidar que no se desbalancee el árbol

25

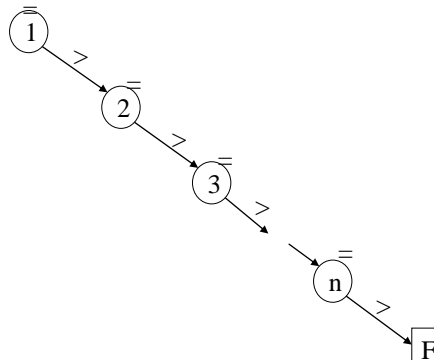
Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Problema de balanceo

- ¿Qué ocurre si tenemos un árbol como el de la figura?
- ¿Cuánto costará buscar el nodo con valor "n"?



26

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

- ✓ Para resolver el problema anterior podemos utilizar *árboles balanceados*
- ✓ Definición: Un árbol **perfectamente balanceado** es aquel en que la cantidad de nodos de su subárbol izquierdo difiere como máximo en 1 de la cantidad de nodos del subárbol derecho
 - En el peor caso, la búsqueda necesita $O(\log n)$.
 - La inserción puede necesitar reorganizar todo el árbol, $O(n)$. → Demasiado costoso!!

27

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

- ✓ **Árbol balanceado ó AVL (Adelson-Velskii y Landis).**
 - Es un árbol binario de búsqueda, con una condición de balanceo más débil que hace que no sea tan costoso el proceso de balancear un árbol
 - Para todo nodo, **la altura de sus subárboles difiere como máximo en 1.**
 - Supondremos que la altura del árbol vacío es -1
 - En cada nodo tendremos dos valores: h_i y h_d , que indican la altura de los subárboles izquierdo y derecho, respectivamente.

28

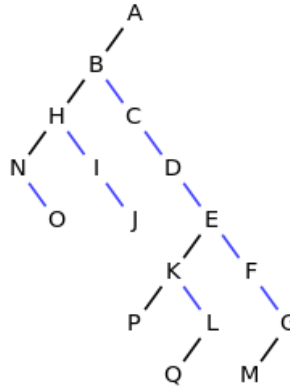
Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Ejemplo

▪ ¿ Altura de...

- Nodo A?
- Nodo B?
- Nodo E?
- Nodo F?
- Nodo M?



29

Tema, Asignatura

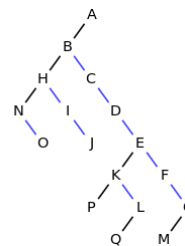
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Ejemplo

▪ ¿ Altura de...

- **Nodo A?**
 - hi: 7; hd: 0
- **Nodo B?**
 - hi: 3; hd: 6
- **Nodo E?**
 - hi: 3; hd: 3
- **Nodo F?**
 - hi: 0; hd: 2
- **Nodo M?**
 - hi: 0; hd: 0



30

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Árbol balanceado ó AVL (Adelson-Velskii y Landis).

- Cada nodo guarda además de la información habitual un valor llamado Factor de Equilibrio (FE)

$$FE = \text{altura subárbol derecho} - \text{altura subárbol izquierdo}$$
- El valor de FE en AVL debe ser 0, -1 ó 1 para que esté balanceado
 - 0 → los dos subárboles tienen la misma altura
 - -1 → El subárbol izquierdo es un nivel más alto
 - 1 → El subárbol derecho es un nivel más alto
- Si $|FE| \geq 2$ es necesario reequilibrar

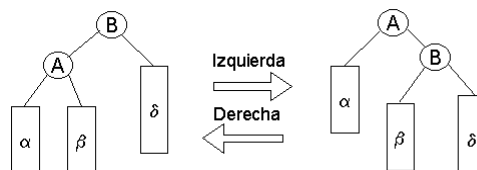
31

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Árbol balanceado ó AVL (Adelson-Velskii y Landis).

- Las operaciones básicas en AVL son las mismas que en un árbol binario, pero seguidas de una operación de **rotación**
- Existen dos casos (para cada subárbol izquierda o derecha):
 - Rotación simple → Se modifica el orden de varios nodos para rebalancear
 - Rotación doble → Se producen dos rotaciones, primero en un subárbol y luego en el otro



32

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Inserción en AVL

- Insertar como en un árbol binario
- Poner el FE al nuevo nodo
- Recorrer el camino hacia atrás hasta el padre para ir actualizando el FE de cada nodo, rebalanceando en caso necesario según los siguientes casos
- Una vez hecho el primer balanceo, no son necesarios más

33

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Rotación simple derecha (**FE > 1 en un nodo**)

- Cuándo: Desbalanceo en la derecha del subárbol derecha
- Dado un árbol raíz R con subárbol izquierdo I y derecho D
 1. Formar un árbol cuya raíz sea la raíz del subárbol derecho D
 2. Hijo derecho será el hijo derecho (y sus descendientes) de la raíz del subárbol derecho D
 3. Hijo izquierdo será nuevo árbol con raíz R, con hijo izquierdo el subárbol izquierdo I y con hijo derecho el hijo izquierdo (y sus descendientes) de la raíz del subárbol derecho D

34

Tema, Asignatura

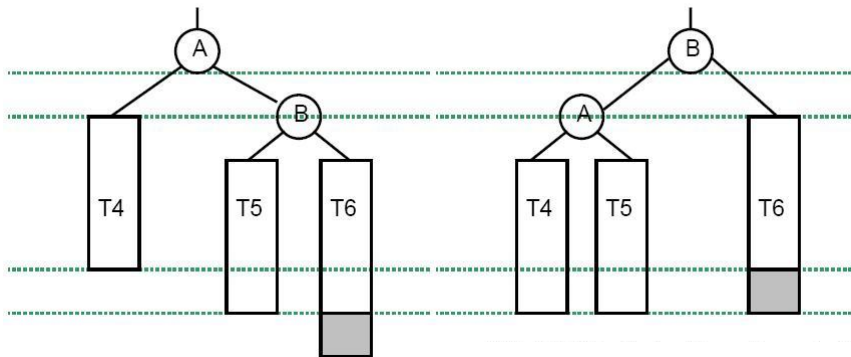
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Rotación simple derecha (pasos)

Inicio (A desbalanceado)

Fin



35

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Rotación simple izquierda (**FE < -1 en un nodo**)

- Cuándo: Desbalanceo a la izquierda del subárbol izquierdo
- Dado un árbol raíz R con subárbol izquierdo I y derecho D:
 1. Formar un árbol cuya raíz sea la raíz del subárbol izquierdo I
 2. Hijo izquierdo será el hijo izquierdo (y sus descendientes) de la raíz del subárbol izquierdo I
 3. Hijo derecho será nuevo árbol con raíz R, con hijo izquierdo el hijo derecho (y sus descendientes) de la raíz subárbol izquierdo I y con hijo derecho el subárbol derecho D

36

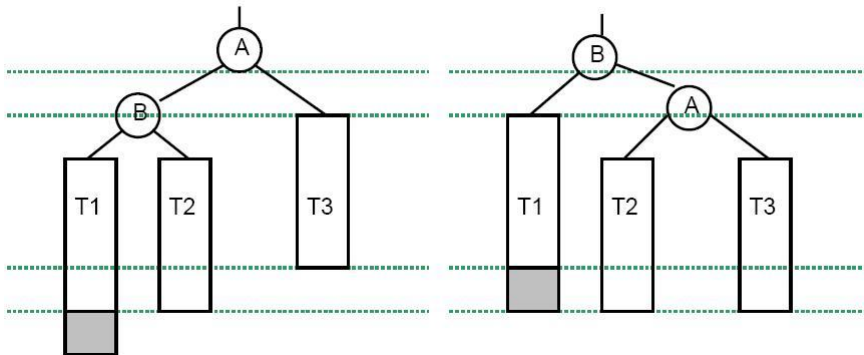
Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

- ✓ Rotación simple izquierda (pasos)

Inicio (A desbalanceado)

Fin



37

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

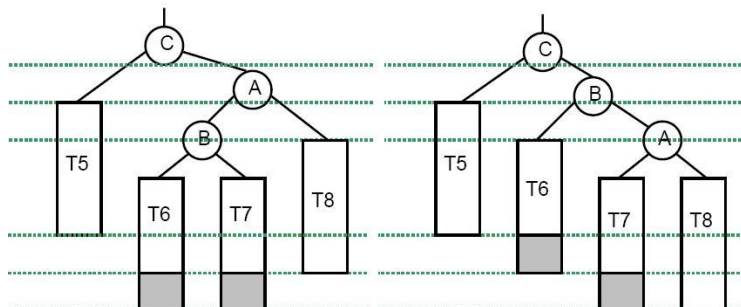
Árboles de búsqueda

- ✓ Rotación doble derecha. (**FE > 1**). Paso 1

- Cuándo: Desbalanceo a la izquierda del subárbol derecha
- Se aplica primero rotación simple izquierda a la raíz del subárbol derecha D y luego simple derecha sobre la raíz del árbol

Inicio (C desbalanceado)

Primera rotación (izquierda en A)



38

Tema, Asignatura

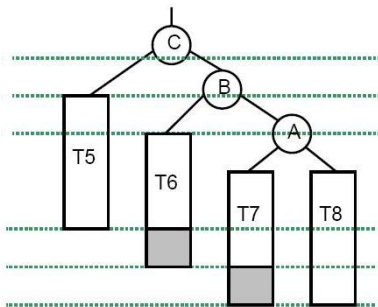
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

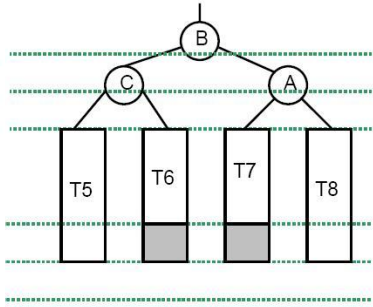
✓ Rotación doble derecha. ($FE > 1$). Paso 2

- Cuándo: Desbalanceo a la izquierda del subárbol derecha
- Se aplica primero rotación simple izquierda a la raíz del subárbol derecha D y luego simple derecha sobre la raíz del árbol

Primera rotación



Fin



39

Tema, Asignatura

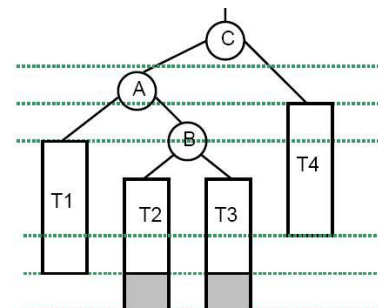
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

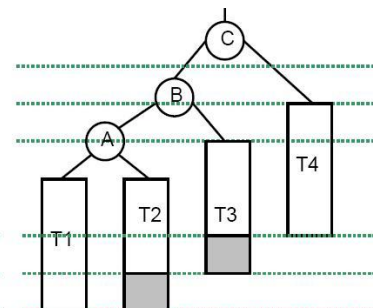
✓ Rotación doble izquierda ($FE < -1$). Paso 1

- Cuándo: Desbalanceo a la derecha del subárbol izquierda
- Se aplica primero rotación simple derecha a la raíz del subárbol izquierda I y luego simple izquierda a la raíz del árbol

Inicio (C desbalanceado)



Primera rotación (derecha en A)



40

Tema, Asignatura

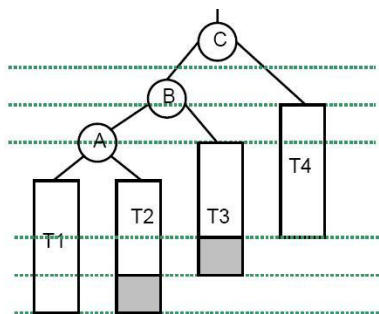
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

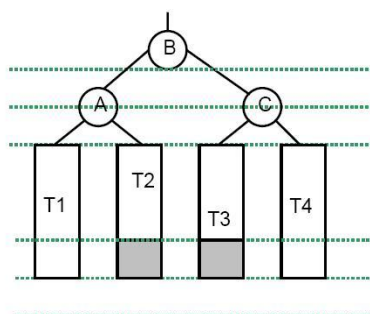
✓ Rotación doble izquierda (FE < -1). Paso 2

- Cuándo: Desbalanceo a la derecha del subárbol izquierda
- Se aplica primero rotación simple derecha a la raíz del subárbol izquierda I y luego simple izquierda a la raíz del árbol

Primera rotación



Fin



41

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Eliminación en AVL

- Eliminar como en un árbol binario
- Recorrer el camino hacia atrás hasta el padre para ir actualizando el FE de cada nodo, rebalanceando en caso necesario como en los casos de inserción vistos anteriormente
- En la eliminación es posible que tengamos que utilizar las rotaciones dobles
- También puede ocurrir que se deban realizar varias operaciones de balanceo en nodos superiores al balanceado, se debe llegar obligatoriamente a la raíz
 - Ejercicio: Proponer un ejemplo en el que ocurra esto

42

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Complejidad AVL

- Las operaciones de rotación, tanto simple como doble, tienen una complejidad de $O(\log(n))$, donde n es la altura del árbol
- Mejora la complejidad de un árbol perfectamente balanceado, y prácticamente se mantiene la eficiencia de búsqueda a $O(\log(n))$ en todos los casos

✓ Otros árboles balanceados

- Árbol rojo-negro (binario)
- Árboles B y B+
 - No tienen por qué ser binarios
 - Se utilizan para índices (bases de datos)

43

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Ejemplo 1

- Inserte las claves en el orden indicado a fin de incorporarlas a un árbol AVL.
- 5,10,20,30,40,50,60

44

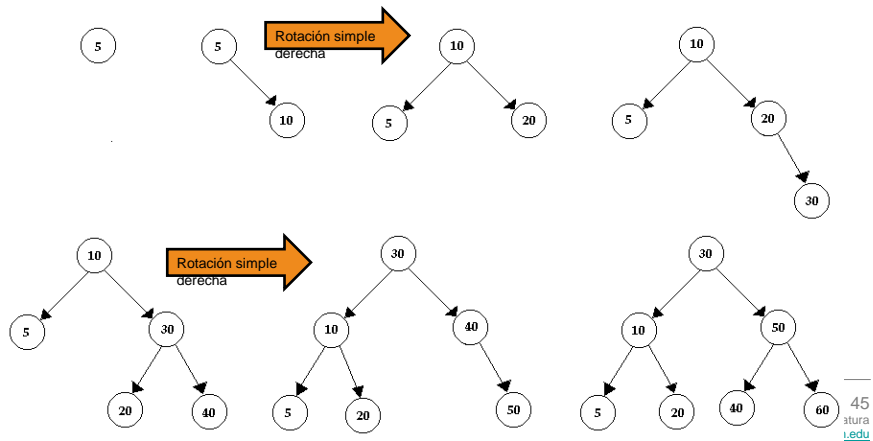
Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Árboles de búsqueda

✓ Ejemplo 1 (SOLUCIÓN)

- Inserte las claves en el orden indicado a fin de incorporarlas a un árbol AVL.
- 5, 10, 20, 30, 40, 50, 60



Árboles de búsqueda

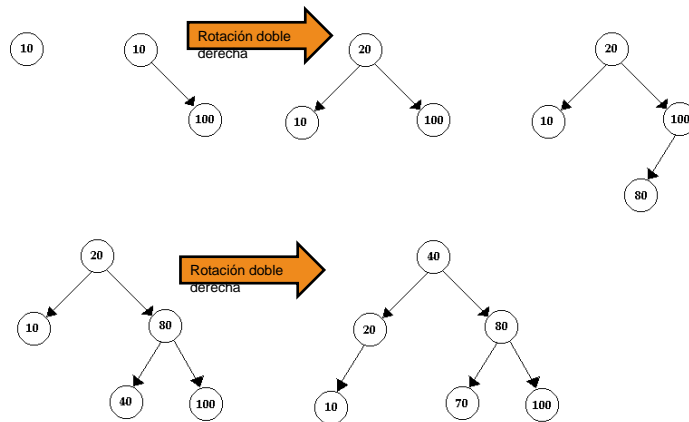
✓ Ejemplo 2

- Inserte las claves en el orden indicado a fin de incorporarlas a un árbol AVL.
- 10, 100, 20, 80, 40, 70

Árboles de búsqueda

✓ Ejemplo 2 (SOLUCIÓN)

- Inserte las claves en el orden indicado a fin de incorporarlas a un árbol AVL.
- 10,100,20,80,40,70



47

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Índice

- ✓ *Introducción*
- ✓ *Búsqueda lineal*
- ✓ *Búsqueda binaria*
- ✓ *Árboles de búsqueda*
- ✓ *Patrones de búsqueda*
 - ✓ *Búsqueda directa*
 - ✓ *Knuth, Morris & Pratt (KMP)*
 - ✓ *Boyer & Moore (BM)*

48

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda

- ✓ Problema: necesitamos encontrar una secuencia (patrón) dentro de otra.
- ✓ De manera formal:
 - Secuencia $\rightarrow S = s_0, s_1, s_2, s_3, \dots s_n$
 - Patrón $\rightarrow P = p_0, p_1, p_2, p_3, p_4, \dots p_m$
- y $n > m$, queremos saber si existe $i \geq 0$ tal que

$$p_j = s_{i+j} \quad \forall j \in \{0, 1, \dots, m\}$$
- ✓ Es decir, buscamos una posición i dentro de S a partir de la cual está el patrón P .

49

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda

- ✓ Veremos tres implementaciones:
 - Búsqueda directa: es el más sencillo, cuando se produce un fallo empieza a comparar desde la siguiente posición de la cadena donde comenzó.
 - KMP: un poco más complicado, intenta “aprender” las distintas subsecuencias que aparecen en el patrón de búsqueda.
 - BM: optimiza el “aprendizaje” sobre el patrón.

50

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. Búsqueda directa

- ✓ Es la forma más sencilla de realizar una búsqueda.
- ✓ Se comprueba la igualdad para las diferentes subsecuencias de S.
- ✓ Si se produce un fallo se vuelve a la posición siguiente desde la que se comenzó.
 - Desde $i = 0$ hasta $n-m$
 - Desde $i = 1$ hasta $(n-m) + 1$
 - Desde $i = 2$ hasta $(n-m) + 2$
 - ...

51

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. Búsqueda directa

- ✓ Ejemplo
 - S = 'xyxxxyxyxyxyxyxyxyxx'
 - P = 'xyxyxyxyxx'

S=	X	Y	X	X	Y	X	Y	X	Y	Y	X	Y	X	Y	X	Y
P=	X	Y	X	Y	Y	X	Y	X	Y	X	X					

↑
j

52

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. Búsqueda directa

✓ Ejemplo

- S = 'xyxxxyxyxyxyxyxyxyxyxx'
- P = 'xyxyxyxyxyxyxx'

S=	X	Y	X	X	Y	X	Y	X	Y	Y	X	Y	X	Y	X	Y
P=	X	Y	X	Y	Y	X	Y	X	Y	X	X					

↑
i

Patrones de búsqueda. Búsqueda directa

✓ Ejemplo

- S = 'xyxxxyxyxyxyxyxyxyxyxx'
- P = 'xyxyxyxyxyxyxx'

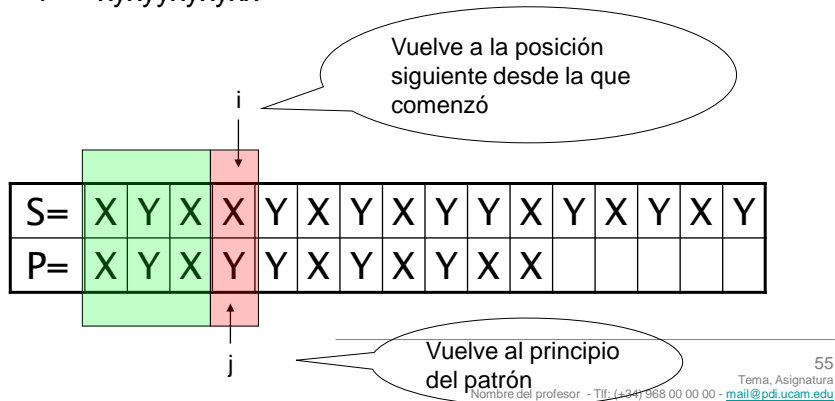
S=	X	Y	X	X	Y	X	Y	X	Y	Y	X	Y	X	Y	X	Y
P=	X	Y	X	Y	Y	X	Y	X	Y	X	X					

↑
i

Patrones de búsqueda. Búsqueda directa

✓ Ejemplo

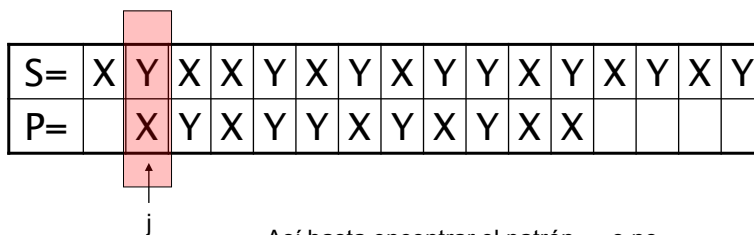
- S = 'xyxxxyxyxyxyxyxyxyxyxx'
- P = 'xyxyxyxyxyxx'



Patrones de búsqueda. Búsqueda directa

✓ Ejemplo

- S = 'xyxxxyxyxyxyxyxyxyxyxx'
- P = 'xyxyxyxyxyxx'



Patrones de búsqueda. Búsqueda directa

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	y	x	y	x	y	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y	x	y	x	y	x	x							
7:							x																
8:								x	y	x													
9:									x														
10:										x													
11:											x	y	x	y	y								
12:												x											
13:													x	y	x	y	y	x	y	x	y	x	x

Soluciones

Patrones de búsqueda. Búsqueda directa

✓ Complejidad

- En el peor de los casos ($n \gg m$) se hacen $n * m$ comparaciones $\rightarrow O(n*m)$.
- El problema es que no aprovechamos el conocimiento que adquirimos sobre el patrón.
- ¿Cómo mejorar?

Búsqueda Patrones: ¿Podemos mejorar?

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	y	x	y	x	y	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y	x	y	x	y	x	x							
7:							x																
8:								x	y	x													
9:									x														
10:										x													
11:											x	y	x	y	y								
12:												x											
13:													x	y	x	y	y	x	y	x	y	x	x

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda Patrones: ¿Podemos mejorar?

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	y	x	y	x	y	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y	x	y	x	y	x	x							
7:							x																
8:								x	y	x													
9:									x														
10:										x													
11:											x	y	x	y	y								
12:												x											
13:													x	y	x	y	y	x	y	x	y	x	x

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda Patrones: ¿Podemos mejorar?

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	y	x	y	x	y	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y	x	y	x	y	x	x							
7:							x																
8:								x	y	x													
9:									x														
10:										x													
11:											x	y	x	y	y								
12:												x											
13:													x	y	x	y	y	x	y	x	y	x	x

Sabemos que aquí no hay una 'x'

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda Patrones: ¿Podemos mejorar?

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	y	x	y	x	y	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y	x	y	x	y	x	x							
7:							x																
8:								x	y	x													
9:									x														
10:										x													
11:											x	y	x	y	y								
12:												x											
13:													x	y	x	y	y	x	y	x	y	x	x

Sabemos que aquí no hay una 'x'

Y seguro que esta subsecuencia está en S

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda Patrones: ¿Podemos mejorar?

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	x	y	x	y	x	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y	x	y	x	y	x	x							
7:							x																
8:								x	y	x													
9:									x														
10:										x													
11:											x	y	x	y	y								
12:												x											
13:													x	y	x	y	y	x	y	x	y	x	x

¿Se parece esto al comienzo del patrón?

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda Patrones: ¿Podemos mejorar?

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	y	x	y	x	y	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y	x	y	x	y	x	x							
7:							x																
8:								x	y	x													
9:																							
10:																							
11:																							
12:																							
13:																							

Sí. Entonces, ¿Porqué no empezamos a comparar desde ahí para la posición de la secuencia donde se ha producido el fallo?

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Búsqueda Patrones: ¿Podemos mejorar?

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	y	x	y	x	y	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y	x	y	x	y	x								
7:							x																
8:								x	y	x													
9:									x														
10:										x													
11:											x	y	x	y	y								
12:												x											
13:													x	y	x	y	y	x	y	x	y	x	x

Nos
ahorraríamos
todas estas
comparaciones

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. KMP

- ✓ Knuth, Morris y Pratt, 1977.
- ✓ Intenta utilizar el conocimiento que adquiere del patrón.
- ✓ Analiza el patrón y genera un array llamado *siguiente*, que incluye la posición a partir la cual seguir comparando.
- ✓ No estudia la secuencia S, analiza el patrón P.

UCAM

UNIVERSIDAD CATÓLICA
SAN ANTONIO

Patrones de Búsqueda :

KMP

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	y	x	y	x	y	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y	x	y	x	y	x	x							
7:							x																
8:								x	y	x													
9:									x														
10:										x													
11:											x	y	x	y	y								
12:												x											
13:													x	y	x	y	y	x	y	x	y	x	x

UCAM

UNIVERSIDAD CATÓLICA
SAN ANTONIO

Patrones de Búsqueda:

KMP

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
	x	y	x	x	y	x	y	x	y	y	x	y	x	y	x	y	y	x	y	x	y	x	x
1:	x	y	x	y																			
2:		x																					
3:			x	y																			
4:				x	y	x	y	y															
5:					x																		
6:						x	y	x	y	y													
7:							x																
8:								x	y	x													
9:									x														
10:										x													
11:																							
12:																							
13:																							

- Ocurre una discrepancia entre la posición 5 del patrón (P_5) con la posición 8 de la secuencia (S_8)
- Los dos caracteres precedentes de S han tenido que ser 'xy' (P_3P_4).
- Resulta que los dos primeros caracteres de P (P_1P_2) son 'xy'. No sería necesario comparar P_1P_2 (seguro que están en la secuencia).
- Si desplazamos P en dos posiciones y seguimos comparando en S_8 nos ahorramos comparaciones.

Patrones de Búsqueda: KMP

✓ El algoritmo de búsqueda será el siguiente:

```
int KMP(char *S, int n, char *P, int m, int * sgte){  
    int i,j;  
    i = j = 0;  
  
    while(j < m && i < n){  
        while (j >= 0 && S[i] != P[j])  
            j = sgte[j];  
        i++;  
        j++;  
    }  
    return ((j == m)? (i - m) : -1);  
}
```

Array de enteros que
nos indica la posición
del patrón por donde
continuar la búsqueda

69

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de Búsqueda: KMP

```
int KMP(char *S, int n, char *P, int m, int * sgte){  
    int i,j;  
    i = j = 0;  
  
    while(j < m && i < n){  
        while (j >= 0 && S[i] != P[j])  
            j = sgte[j];  
        i++;  
        j++;  
    }  
    return ((j == m)? (i - m) : -1);  
}
```

Fijémonos en
esta parte del
código

70

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

[illegible]

Patrones de Búsqueda : KMP

Ejemplo

S = 'xyxxxyxyxyxyxyxyxyxyxx'

P = 'xyxyxyxyxyxx'

✓ Cálculo del array 'sgte' para el patrón P = 'xyxyxyxyxyxx'.

P [1] → Si falla sabemos que no será una 'y', pero puede ser una 'x'.
Por lo que sabemos del patrón, hay una 'x' en P[0]. Habrá que empezar desde esa posición pero comparando con el carácter actual de S.

Pos	0	1	2	3	4	5	6	7	8	9	10
P	X	Y	X	Y	Y	X	Y	X	Y	X	X
sgte	-1	0									

Patrones de Búsqueda : KMP

Ejemplo

S = 'xyxxxyxyxyxyxyxyxyxyxx'

P = 'xyxyxyxyxyxx'

✓ Cálculo del array 'sgte' para el patrón P = 'xyxyxyxyxyxx'.

P [2] → Si falla, seguro que en esa posición no hay una 'x'.
Por lo que sabemos del patrón, hay una 'x' y una 'y' en P[0] y P[1] respectivamente.
No podemos empezar a comparar desde P[0] para la misma posición en S ya que tiene que ser distinto de 'x'. Debemos de aumentar en 1 la posición en S.

Pos	0	1	2	3	4	5	6	7	8	9	10
P	X	Y	X	Y	Y	X	Y	X	Y	X	X
sgte	-1	0	-1								

Patrones de Búsqueda : KMP

Ejemplo

S = 'xyxxxyxyxyxyxyxyxyxyxx'

P = 'xyxyxyxyxyxx'

✓ Cálculo del array 'sgte' para el patrón P = 'xyxyxyxyxyxx'.

P [3] → Si falla, seguro que en esa posición no hay una 'y'.

Por lo que sabemos del patrón, en P[0], P[1] y P[3] está la subsecuencia 'xyx' respectivamente.

Como P[3] != 'y' y P[0] = 'x' podemos comenzar desde P[0] sin aumentar la posición en S.

Pos	0	1	2	3	4	5	6	7	8	9	10
P	X	Y	X	Y	Y	X	Y	X	Y	X	X
sgte	-1	0	-1	0							

Patrones de Búsqueda : KMP

Ejemplo

S = 'xyxxxyxyxyxyxyxyxyxyxx'

P = 'xyxyxyxyxyxx'

✓ Cálculo del array 'sgte' para el patrón P = 'xyxyxyxyxyxx'.

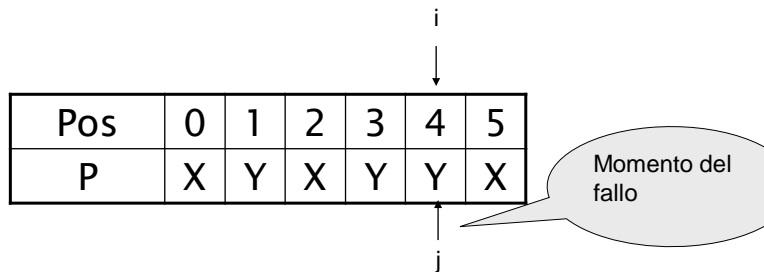
P [4] → Si falla, seguro que en esa posición no hay una 'y'.

Por lo que sabemos del patrón, en P[0], P[1], P[2] y P[3] está la subsecuencia 'xyxy'

Como P[2]P[3] = 'xy' y P[0]P[1]='xy', podemos comenzar a comparar desde P[2] para la misma posición de la secuencia. Ya que P[2] (una 'x') puede ser igual a lo que hay en P[4] (no es una 'y').

Pos	0	1	2	3	4	5	6	7	8	9	10
P	X	Y	X	Y	Y	X	Y	X	Y	X	X
sgte	-1	0	-1	0	2						

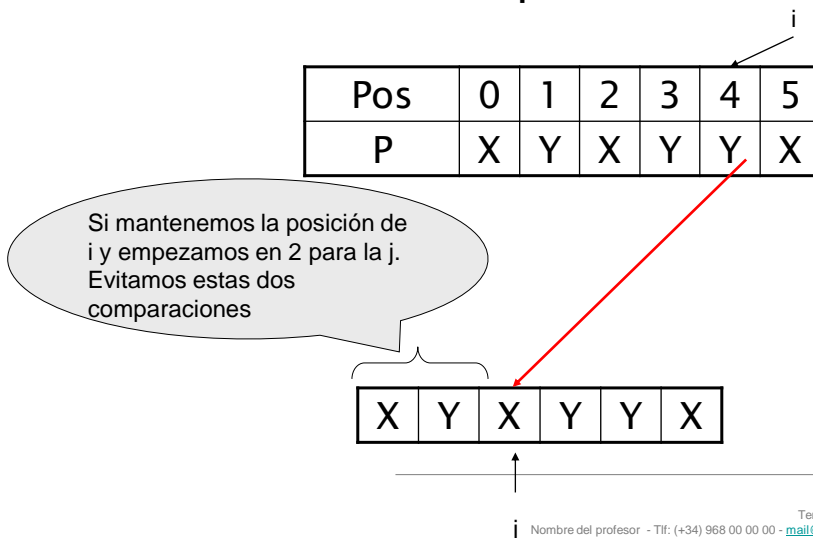
Patrones de Búsqueda: KMP



77

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de Búsqueda: KMP



78

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de Búsqueda : KMP

Ejemplo

S = 'xyxxxyxyxyxyxyxyxyxyxx'

P = 'xyxyxyxyxyxx'

✓ Cálculo del array 'sgte' para el patrón P = 'xyxyxyxyxyxx'.

P [7] → Si falla, seguro que en esa posición no hay una 'x'.

Como P[0]P[1] = 'xy' y P[5]P[6]='xy', podemos comenzar a comparar desde P[2] para la misma posición de la secuencia. **Pero como P[2] es una 'x' como P[7], no tiene sentido hacerlo!!!**

Se debe actuar como en P[2]. Entonces P[7] = P[2]

Pos	0	1	2	3	4	5	6	7	8	9	10
P	X	Y	X	Y	Y	X	Y	X	Y	X	X
sgte	-1	0	-1	0	2	-1	0	-1			

Patrones de Búsqueda : KMP

Ejemplo

S = 'xyxxxyxyxyxyxyxyxyxyxx'

P = 'xyxyxyxyxyxx'

✓ Cálculo del array 'sgte' para el patrón P = 'xyxyxyxyxyxx'.

P [8] → Si falla, seguro que en esa posición no hay una 'y'.

Como P[0]P[1]P[2] = 'xyx' y P[5]P[6][7]='xyx', podemos comenzar a comparar desde P[3] para la misma posición de la secuencia. **Pero como P[3] es una 'y' como P[8], no tiene sentido hacerlo!!!**

Se debe actuar como en P[3]. Entonces P[8] = P[3]

Pos	0	1	2	3	4	5	6	7	8	9	10
P	X	Y	X	Y	Y	X	Y	X	Y	X	X
sgte	-1	0	-1	0	2	-1	0	-1	0		

Patrones de Búsqueda : KMP

Ejemplo

S = 'xyxxxyxyxyxyxyxyxyxyxx'

P = 'xyxyxyxyxx'

✓ Cálculo del array 'sgte' para el patrón P = 'xyxyxyxyxx'.

P [9] → Si falla, seguro que en esa posición no hay una 'x'.

Por lo que sabemos del patrón, P[0..8] es 'xyxyxyxy'.

Si nos fijamos bien P[0..3] ('xyxy') es igual a P[5..8].

Podemos empezar en P[4] (es una 'y') para la misma posición en S (seguro que no es una 'x', por P[9]!='x', y **sí puede ser una 'y'**)

Pos	0	1	2	3	4	5	6	7	8	9	10
P	X	Y	X	Y	Y	X	Y	X	Y	X	X
sgte	-1	0	-1	0	2	-1	0	-1	0	4	

Patrones de Búsqueda : KMP

Ejemplo

S = 'xyxxxyxyxyxyxyxyxyxyxx'

P = 'xyxyxyxyxx'

✓ Cálculo del array 'sgte' para el patrón P = 'xyxyxyxyxx'.

Finalmente queda el array 'sgte' como sigue

Pos	0	1	2	3	4	5	6	7	8	9	10
P	X	Y	X	Y	Y	X	Y	X	Y	X	X
sgte	-1	0	-1	0	2	-1	0	-1	0	4	3

Patrones de búsqueda. KMP

- ✓ La cadena madre S siempre se recorre hacia la derecha (no hay retrocesos), aunque un mismo carácter de S puede compararse con varios del patrón P (**discrepancias**).
- ✓ Cuando haya una discrepancia se consultará una **tabla** para saber cuánto hay que retroceder en el patrón o, dicho de otra forma, cuántos desplazamientos del patrón hacia la derecha pueden hacerse.
 - La tabla que hemos visto en el ejemplo anterior

83

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. KMP

- ✓ ¿Cómo calcular la tabla?
 - En la tabla hay un entero por cada carácter de P , e indica cuántos desplazamientos hacia la derecha deben hacerse cuando ese carácter discrepe con uno de la cadena madre.
 - Para cada p_i de P , hay que calcular el sufijo más largo $p_{i-j}p_{i-j+1}\dots p_{i-1}$ que es igual al prefijo de P $p_0p_1\dots p_{j-1}$
 - Entoces, $\text{sgte}(p_i) = \max \{ j \mid 0 < j < i-1, p_{i-j}p_{i-j+1}\dots p_{i-1} = p_0p_1\dots p_{j-1} \}$
 - Pero si $p_i = p_j$ entonces $\text{sgte}(p_i) = \text{sgte}(p_j) !!$
 - Si no hay valor de j posible (no hay patrón), entonces
 - $\text{sgte}(p_i) = 0$ si $p_i \neq p_0$, o
 - $\text{sgte}(p_i) = -1$ si $p_i = p_0$

84

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. KMP

✓ ¿Cómo calcular la tabla?

▪ Algoritmo para calcular la tabla

- Si P_0 es fallo, entonces $P_0 = -1$
- Si P_1 es fallo
 - Si $P_0 == P_1$, entonces $P_1 = -1$. Pasamos al siguiente carácter de S y comenzamos la comparación desde el principio
 - Sino ($P_0 != P_1$), entonces $P_1 = 0$. El carácter actual de S podría coincidir con P_1 .

85

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. KMP

✓ ¿Cómo calcular la tabla?

▪ Si P_2 es fallo

- Si $P_0 == P_1$
 - Si $P_1 != P_2$ $P_2 = 1$ // empezar a partir de P_1
 - Sino (Son iguales, la comparación fallaría en este punto)
 - Si $P_0 != P_2$ entonces $P_2 = 0$
 - Sino $P_2 = -1$
- Sino
 - Si $P_0 != P_2$ entonces $P_0 = 0$
 - Sino $P_2 = -1$

86

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. KMP

✓ Algoritmo para calcular la tabla “siguiente” de P

```
void preKMP(char p[NUM_ELEM_P], int tablaNext[NUM_ELEM_P]) { // p es el patrón
    int i, j;
    i = 0;
    j = tablaNext[0] = -1;
    while (i < NUM_ELEM_P - 1) {
        while (j > -1 && p[i] != p[j]) { // Retrocede hasta encontrar
            j = tablaNext[j];           // el inicio de un patrón visto antes
        }
        i++;
        j++;
        if (p[i] == p[j]) // Patrón visto antes
            tablaNext[i] = tablaNext[j];

        else // Fin de patrón visto antes
            tablaNext[i] = j;
    }
}
```

87

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. KMP

✓ Algoritmo KMP

```
void KMPSearch(char s[NUM_ELEM], char p[NUM_ELEM_P], int *posiciones) {
    int i, j, k; // i índice de S, j índice de P, k de posiciones
    int tablaNext[NUM_ELEM_P]; // Crear la tabla de P y calcularla
    preKMP(p, tablaNext);
    i = j = k = 0;
    while (i < NUM_ELEM) {
        while (j > -1 && s[i] != p[j])
            j = tablaNext[j];
        i++;
        j++;
        if (j >= NUM_ELEM_P) { // Patrón encontrado, se almacena
            posiciones[k] = i - j;
            k++;
            j = 0; // Ponemos el patrón para seguir buscando
        }
    }
}
```

88

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. KMP

✓ Complejidad

- Cálculo de la tabla “siguiente” $\rightarrow O(m)$, siendo m el tamaño del patrón
- Ejecución del algoritmo KMP con “siguiente” sobre $S \rightarrow O(n+m)$, siendo n el tamaño de la secuencia a analizar
- La mejor tabla “siguiente” sería tener todos los valores a -1 para saltar al siguiente carácter de S

89

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

- ✓ R.S. Boyer y J.S. Moore, 1977
- ✓ Como el algoritmo KMP, el algoritmo BM puede encontrar todas las apariciones de un patrón P (de longitud m) en una cadena madre S (de longitud n) en un tiempo $O(n)$ en el caso peor.
- ✓ KMP examina cada carácter de S al menos una vez, por tanto realiza un mínimo de n comparaciones.
- ✓ BM es **sublineal**: no examina necesariamente todos los caracteres de S y el número de comparaciones es, a menudo, inferior a n .

90

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

- ✓ Además, BM tiende a ser más eficiente cuando m crece (longitud del patrón).
- ✓ En el mejor caso, BM encuentra todas las apariciones de P en S en un tiempo $O(m+n/m)$.
- ✓ Como en KMP, desplazamos P sobre S de izquierda a derecha examinando los caracteres enfrentados.
- ✓ Pero ahora la verificación de los caracteres de P se hace de derecha a izquierda después de cada desplazamiento del patrón.

91

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

- ✓ La búsqueda en BM se realiza mediante **reglas**
- ✓ **Regla A.** Después de un desplazamiento de S , se compara p_m con el carácter c_{i+m} de S y si son distintos:
 - **A.1** Si c_{i+m} aparece más a la izquierda en el patrón, se desplaza éste para alinear la última aparición (más a la derecha) de c_{i+m} en el patrón con el carácter c_{i+m} de S .
 - **A.2** Si c_{i+m} no está en P , se coloca éste (P) inmediatamente detrás de la aparición de c_{i+m} en S .

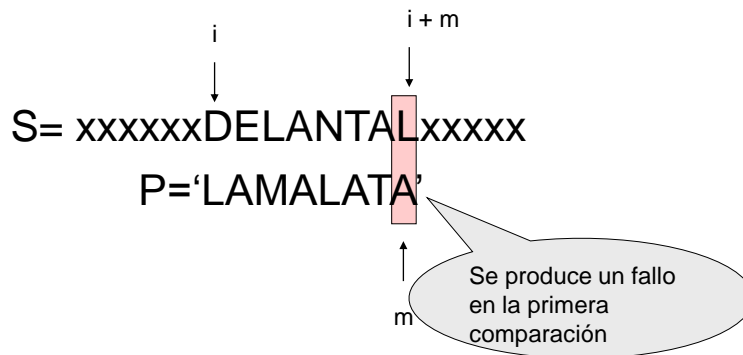
92

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Ejemplo de regla A.1

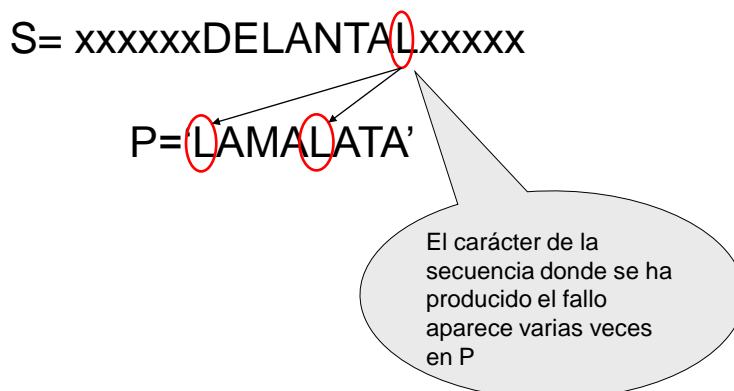


93

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Ejemplo de regla A.1



94

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Ejemplo de regla A.1

S= xxxxxxxxxANTALATAxxxxx

P='LAMALATA'

Podemos igualar la 'L' del patrón que aparece más cerca del final y empezar de nuevo el algoritmo comparando desde el final

95

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Ejemplo de regla A.2

S= xxxxxxDELANTALxxxxx

P='MATA'

Se produce un fallo en la primera comparación

96

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Ejemplo de regla A.2

El carácter en la secuencia
donde se ha producido el fallo
no está en el patrón

S= xxxxxxDELANTALxxxxx

P='MATA'

¿?

97

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

97

Tema, Asignatura
mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Ejemplo de regla A.2

Podemos igualar el comienzo del
patrón con el carácter siguiente
al que ha dado fallo ...

S= xxxxxxDELANTALAJACAxxxx

P='MATA'

... y comenzar de
nuevo el algoritmo

98

Tema, Asignatura
00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

- ✓ **Regla B.** Si un cierto número de caracteres al final de P se corresponde con caracteres de S , aprovechamos este conocimiento parcial de S (como en KMP) para desplazar P a una nueva posición compatible con la información que poseemos.

99

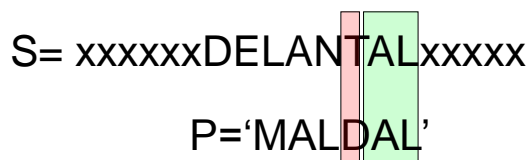
Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

- ✓ Ejemplo de regla B

S= xxxxxxDELANTALxxxxx

P='MALDAL'



Se produce un fallo
después de haber hecho
varias comprobaciones con
éxito

100

Tema, Asignatura
Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Ejemplo de regla B

S= xxxxxxDELANTALxxxxx

P='MALDAL'

La subsecuencia
analizada que ha tenido
éxito se encuentra en el
patrón

101

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Ejemplo de regla B

S= xxxDELANTALITALxx

P='MALDAL'

Igualamos la
subsecuencia
encontrada con la del
patrón ...

... y seguimos
con el algoritmo

102

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Veamos ahora un ejemplo completo

S = 'se espera cielo nublado para mañana'

P = 'lado'

se **e**spera cielo nublado para mañana
lado

¿Aparece 'e' en P? ...
No, entonces aplicamos la
regla A.2. Movemos P hasta el
carácter siguiente

103

Tema, Asignatura
mail@pdi.ucam.edu

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

se **r**espera cielo nublado para mañana
lado

¿Aparece 'r' en P? ...
No, entonces aplicamos la regla
A.2. Movemos P hasta el carácter
siguiente

se **i**espera cielo nublado para mañana
lado

¿Aparece 'i' en P? ...
No, entonces aplicamos la regla
A.2. Movemos P hasta el
carácter siguiente

104

Tema, Asignatura
mail@pdi.ucam.edu

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

se espera cielo nublado para mañana

lado

¿Aparece ' ' en P? ...
No, entonces aplicamos la regla
A.2. Movemos P hasta el
carácter siguiente

se espera cielo nublado para mañana

lado

¿Aparece 'l' en P? ...
Sí, entonces aplicamos la regla A.1.
Movemos P hasta igualar los caracteres

105

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

se espera cielo nublado para mañana

lado

Se repite este proceso hasta
encontrar la cadena buscada.
En este ejemplo sólo hemos
utilizado la regla A.

106

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Otro ejemplo

S = 'babcbabcaabcbabcacabc'

P = 'abcabcacab'

babcbab**cab**caabcbabcacabc
ab**cab**ca**cab**

La subsecuencia xcab (con x != a) se encuentra en el patrón. Aplicamos la regla B moviendo P para que coincidan

107

Tema, Asignatura
mail@pdi.ucam.edu

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

babcbab**cab**caabcbabcacabc
ab**cab**ca**cab**

Como hemos visto en el ejemplo anterior aplicamos la regla A.1.

108

Tema, Asignatura
mail@pdi.ucam.edu

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

- ✓ Para implementar este algoritmo se necesitan dos arrays de enteros:
 - D1[c]: Donde 'c' es cualquier elemento del juego de caracteres del patrón + una posición extra para el resto de caracteres.
 - D2[i]: Donde 'i' está en el rango desde 0 hasta m-2, siendo 'm' el número de elementos de P.
- ✓ Ambos representarán el número de posiciones que hay que desplazar el patrón sobre la secuencia.
 - D1 para las reglas A
 - D2 para la regla B (por eso llega hasta m-2)

109

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Cálculo de D1

- Para un carácter 'c', el valor D1[c] será:
 - Si 'c' no aparece en P, entonces $D1[c]=m$
 - Si no, $D1[c] = m - (\max \{i \mid P[i] = c\}) - 1$

La posición más a la derecha de P donde aparece 'c'

- Ejemplo. Vector D1 para el patrón "OSTENTE"

c	E	T	N	S	O	otros
D1[c]	0	1	2	5	6	7

110

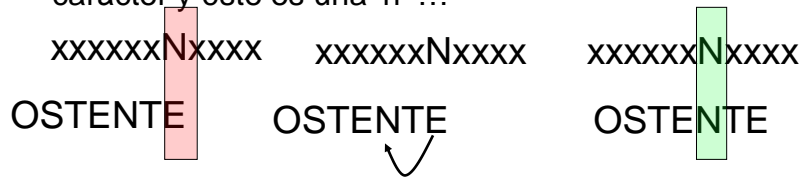
Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Cálculo de D1

- Ejemplo (cont). Si se produce un fallo en el primer carácter y éste es una 'n' ...



Habrá que mover el patrón 2 posiciones para casar los caracteres coincidentes

c	E	T	N	S	O	otros
D1[c]	0	1	2	5	6	7

111

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Cálculo de D2

- Similar a KMP, pero empezando desde la última posición a la primera
- No depende de S, sólo de P
- Para la última posición no existe valor de entrada, ya que si no se habría aplicado la regla A

112

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Cálculo de D2

- Ejemplo con el patrón “OSTENTE”

i	0	1	2	3	4	5	6
P[i]	O	S	T	E	N	T	E
D2[i]							-

Patrones de búsqueda. BM

✓ Cálculo de D2

- P[5]
 - Sabemos que P[6] es una ‘e’.
 - Y que en carácter de la secuencia S coincidente con P[5] no hay una ‘t’.
 - Aunque P[3] es una ‘e’, no podemos igualar P[3] con P[6] por que P[2] es una ‘t’ y sabemos que P[5] no tiene que serlo.
 - No hay coincidencia de patrón
 - Debemos indicar cuánto hay que sumar al inicio del patrón para ponerlo en la posición de la última letra del patrón + 1
 - Esto es igual a ... Tamaño del patrón!

i	0	1	2	3	4	5	6
P[i]	O	S	T	E	N	T	E
D2[i]						7	-

Patrones de búsqueda. BM

✓ Ejemplo de uso de D2

- $S[6] == P[6] \rightarrow \text{OK!}$
- $S[5] != P[5] \rightarrow \text{NO}$, aplicamos tabla D2
 - Sumamos el valor en $D2[5]$ a la posición i que marca el inicio actual del patrón
 - $D2[5] + 0 = 7 + 0 = 7 \rightarrow \text{En esta posición ponemos el inicio del patrón}$

i	0	1	2	3	4	5	6	7	8	9
S[i]	O	S	T	E	N	Z	E			
P[i]	O	S	T	E	N	T	E			
D2[i]						7	-			
Tras el fallo								O	S	T

115

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Cálculo de D2

- P[4]
- Sabemos que P[5,6] es 'te'.
- Y que en carácter de la secuencia S coincidente con P[4] no hay una 'n'.
- Ya que P[2,3] es 'te' y P[1] es 's', distinto de 'n', entonces podemos igualar P[2,3] con P[5,6]
- Desplazamos P el número de letras desde el inicio para que coincida con el patrón encontrado
 - Inicio patrón en la parte derecha – inicio patrón en la parte izquierda
 - $5 - 2 = 3$

i	0	1	2	3	4	5	6
P[i]	O	S	T	E	N	T	E
D2[i]					3	7	-

116

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Ejemplo de uso de D2

- $S[6] == P[6]$, $S[5] == P[5] \rightarrow \text{OK!}$
- $S[4] != P[4] \rightarrow \text{NO}$, aplicamos tabla D2
 - Sumamos el valor en $D2[4]$ a la posición i que marca el inicio actual del patrón
 - $D2[4] + 0 = 3 + 0 = 3 \rightarrow \text{En esta posición ponemos el inicio del patrón}$

i	0	1	2	3	4	5	6	7	8	9
S[i]	O	S	T	E	Z	T	E			
P[i]	O	S	T	E	N	T	E			
D2[i]					3	7	-			
Tras el fallo				O	S	T	E	N	T	E

117

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Cálculo de D2

- $P[3]$
 - Sabemos que $P[4..6]$ es 'nte'.
 - Y que en carácter de la secuencia S coincidente con $P[4]$ no hay una 'e'.
 - No existe ningún patrón "xnte" anterior a $P[4..6]$ en la P.
 - Probamos con el patrón "xnte", eliminando la "n". **Pero ahora el patrón de la izquierda debe empezar en la posición 0!!**
 - No hay patrón que cumpla esto, se pone el tamaño del patrón

i	0	1	2	3	4	5	6
P[i]	O	S	T	E	N	T	E
D2[i]				7	3	7	-

118

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Cálculo de D2

- Para el resto de P se calcula de forma similar a P[5]

i	0	1	2	3	4	5	6
P[i]	O	S	T	E	N	T	E
D2[i]	7	7	7	7	3	7	-

Patrones de búsqueda. BM

✓ Código en C. Código BM

```
#include <string.h>

void BM_search(char *s, char *p, int *posiciones) {
    int s_len, p_len;                /* * Calc string sizes */
    s_len = strlen(s);
    p_len = strlen(p);

    int D1[ALPHABET_SIZE];           /* * Tamaño del alfabeto */
    int D2[p_len+1];
    preparar_D1(p,p_len, D1);
    preparar_D2(p,p_len, D2);

    ... (siguiente diapositiva)
```

✓ Código en C. Código BM (II)

```
void BM_search(char *s, char *p, int *posiciones) {  
    /* Búsqueda Boyer-Moore */  
    int i= 0; int p= 0;  
    while(i <= (s_len - p_len)) {  
        int j = p_len;  
        while(j > 0 && p[j-1] == s[i+j-1])  
            j--;  
        if(j > 0) {  
            int k = D1[((size_t) s[i+j-1])-97]; // 97 compensa código ASCII  
            int m;  
            if(k < j && (m = j-k-1) > D2[j])  
                i+= m;  
            else  
                i+= D2[j];  
        }  
        else {  
            posiciones[p] = i;  
            p++; i++;  
        }  
    }  
}
```

121

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

✓ Código en C. Tabla D1

```
void preparar_D1(char *str, int size, int result[ALPHABET_SIZE]) {  
    int i;  
  
    for (i = 0; i < ALPHABET_SIZE; i++)  
        result[i] = -1;  
  
    for (i = 0; i < size; i++)  
        result[((size_t)str[i])-97] = i;  
}
```

122

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

✓ **Código en C. Tabla D2 (función auxiliar)**

```
void calcular_prefijo(const char* str, int size, int result[]) {
    int q;
    int k;
    result[0] = 0;

    k = 0;
    for (q = 1; q < size; q++) {
        while (k > 0 && str[k] != str[q])
            k = result[k-1];

        if (str[k] == str[q])
            k++;
        result[q] = k;
    }
}
```

123

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu✓ **Código en C. Tabla D2**

```
void preparar_D2(char *normal, int size, int result[]) {
    char *left = (char *) normal;
    char *right = left + size;
    char reversed[size+1];
    char *tmp = reversed + size;
    int i;

    // Invertir el patron
    *tmp = 0;
    while (left < right)
        * (--tmp) = *(left++);

    int prefix_normal[size];
    int prefix_reversed[size];

    calcular_prefijo(normal, size, prefix_normal);
    calcular_prefijo(reversed, size, prefix_reversed);
    ... (siguiente diapositiva)
```

124

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

✓ Código en C. Tabla D2 (II)

```
void preparar_D2(char *normal, int size, int result[]) {  
  
    for (i = 0; i <= size; i++) {  
        result[i] = size - prefix_normal[size-1];  
    }  
  
    for (i = 0; i < size; i++) {  
        const int j = size - prefix_reversed[i];  
        const int k = i - prefix_reversed[i]+1;  
  
        if (result[j] > k)  
            result[j] = k;  
    }  
}
```

125

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Patrones de búsqueda. BM

✓ Complejidad

- Pre-procesamiento del patrón (cálculo de las tablas) → $O(m)$
- Algoritmo BM → $O(n)$ en el peor caso, $O(n/m)$ en el mejor caso

126

Tema, Asignatura

Nombre del profesor - Tlf: (+34) 968 00 00 00 - mail@pdi.ucam.edu

Ejercicios

- ✓ KMP: calcular el vector siguiente para el patrón
P="alabalo"
- ✓ BM: calcular los vectores D1 y D2 para los patrones
P="alabalo" y "alosealosalo"

Bibliografía

- ✓ César Vaca Rodríguez. **Tablas de dispersión**. Departamento de Informática. Universidad de Valladolid
- ✓ Rosa Guerequeta y Antonio Vallecillo. **Técnicas de Diseño de Algoritmos**. Segunda Edición. Servicio de Publicaciones de la Universidad de Málaga, 2000.