

# **TEMA 8: ALGORITMOS SOBRE CONJUNTOS**

ALGORITMOS Y ESTRUCTURAS DE DATOS

M. Colebrook Santamaría

J. Riera Ledesma

# Objetivos

- Representación de conjuntos.
- Inserción de elementos en un conjunto.
- Eliminación de elementos en un conjunto.
- Pertenencia de elementos a un conjunto.
- Operaciones:
  - Complemento
  - Unión de conjuntos
  - Intersección de conjuntos
  - Diferencia de conjuntos
  - Diferencia simétrica de conjuntos

# Representación de conjuntos (1)

- En Matemáticas e Ingeniería Informática, el término **conjunto** (*set*) se refiere a una colección no ordenada de objetos denominados **elementos o miembros** del conjunto.
- Un conjunto se denota normalmente mostrando la lista de elementos encerrados entre llaves, { y }.
- Por ejemplo, el conjunto de todos los números primos entre 1 y 20 es { 2, 3, 5, 7, 11, 13, 17, 19 }.
- En un problema que involucra conjuntos, los elementos se seleccionan de algún conjunto dado denominado el **conjunto universal U**.
- Por ejemplo, si el conjunto universal U es el conjunto de nombres de los meses del año, entonces el conjunto de meses con 30 días o menos sería { febrero, abril, junio, septiembre, noviembre }.

# Representación de conjuntos (2)

- Los conjuntos cuyos elementos se escogen de un **conjunto universal finito  $U$**  pueden ser representados en memoria usando **bits**, cuyo número será igual a la cardinalidad de  $U$ .
- Cada **bit** corresponde exactamente con un **elemento** del conjunto universal  $U$ .
- Cada conjunto se representa como un *string* de bits, en el que cada bit corresponde con un elemento del conjunto  $U$ . Si el bit  $i$  está a 1, significa que ese elemento está en el conjunto.

# Representación de conjuntos (3)

- Como ilustración, supongamos que U es el conjunto de **letras mayúsculas**. Por tanto, cualquier conjunto se puede representar usando 26 bits.

- Por ejemplo, el conjunto de las vocales mayúsculas

{ A, E, I, O, U }

se pueden representar como:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0

# Representación de conjuntos (4)

- En nuestro caso, vamos a simplificar la representación para poder implementar **conjuntos de números enteros sin signo** dentro del dominio  $0..2^n-1$ , con  $n>3$ .
- Por ejemplo, si  $U$  es el conjunto de números entre 0 y 15, con  $|U| = 16$ , podemos representar cualquier conjunto con **16 bits** (2 bytes).
- Es decir, el conjunto  $A = \{ 2, 3, 5, 7, 11 \}$  podemos representarlo de la siguiente forma:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A:	0	0	0	0	1	0	0	0	1	0	1	0	1	1	0	0

# Inserción de elementos en un conjunto

- Para cualquier elemento  $x \in U$ , su inserción en un conjunto se realiza usando los **operadores binarios**:

| (OR) y << (SHIFT: desplazamiento).

- Como  $|U| = 16$ , cualquier conjunto se representa como:

```
typedef unsigned short conjunto_t;  
conjunto_t A = 0;
```

- Por tanto, si  $i = 5$  es el valor a añadir al conjunto A:

```
conjunto_t uno = 1;  
A |= (uno << i); // uno << 5 = 00000000000100000
```

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A:	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

# Eliminación de elementos en un conjunto

- Para eliminar un elemento de un conjunto usaremos los **operadores binarios** & (AND), ~ (NOT) y << (SHIFT).
- Por ejemplo, dado el conjunto  $A = \{ 5, 7, 11 \}$

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A:	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0

podemos eliminar el valor  $i = 7$  de la siguiente forma:

conjunto\_t                      uno                      =                      1;

uno <<= i;    //    uno << 7 = 0000000010000000

A &= ~uno;    //    ~uno                      = 1111111101111111

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A:	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0



# Pertenencia de elementos a un conjunto $\in$

- La pertenencia ( $\in$ ) de un elemento  $x$  a un conjunto  $A$ , se desarrolla usando los operadores binarios  $\&$  (AND) y  $\ll$  (SHIFT).
- Por ejemplo, dado el conjunto  $A = \{ 5, 7, 11 \}$ , podemos comprobar eficientemente si el elemento  $i = 7$  pertenece al conjunto.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A:	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0

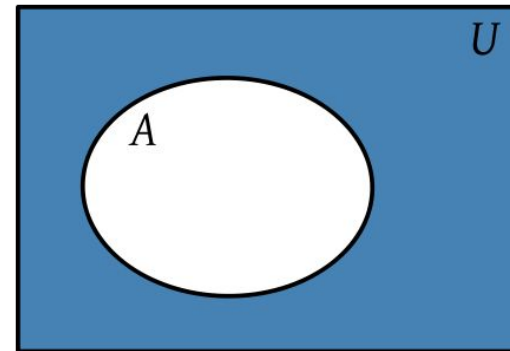
A=0000100010100000  
0000000010000000

```
conjunto_t uno = 1;
conjunto_t uno <<= i;
if ((A & uno) != 0) cout << i << " ∈ A";
else cout << i << " ∉ A";
```

# Operaciones: Complemento [

- El **complemento de un conjunto A** es el conjunto  $A^c$  que contiene todos los elementos que **no** pertenecen a A.

$$A^c = \{x : x \in U \wedge x \notin A\}$$



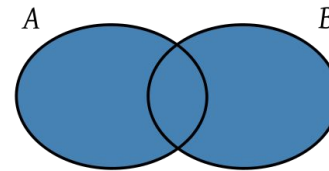
- Esta operación se implementa con el **operador binario**  $\sim$  (NOT).

conjunto\_t Ac =  $\sim$ A;

# Operaciones: Unión de conjuntos $\cup$

- La **unión de conjuntos**  $A \cup B$  es el conjunto de todos los elementos que pertenecen al menos a uno de los conjuntos A y B.

$$A \cup B = \{x : x \in A \vee x \in B\}$$



- Sea  $A = \{5, 7, 11\}$  y  $B = \{1, 3, 5\}$ ,  $A \cup B = \{1, 3, 5, 7, 11\}$

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A:	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0
B:	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0
C:	0	0	0	0	1	0	0	0	1	0	1	0	1	0	1	0

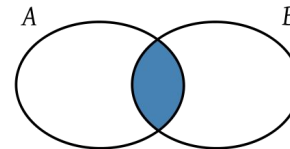
conjunto\_t A, B, C;

C = A | B;

# Operaciones: Intersección de conjuntos $\cap$

- La **intersección de conjuntos**  $A \cap B$  es el conjunto de todos los elementos comunes a A y B.

$$A \cap B = \{x : x \in A \wedge x \in B\}$$



- Sea  $A = \{ 5, 7, 11 \}$  y  $B = \{ 1, 3, 5 \}$ ,  $A \cap B = \{ 5 \}$

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A:	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0
B:	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0
C:	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

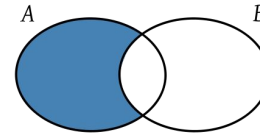
```
conjunto_t A, B, C;
```

```
C = A & B;
```

# Operaciones: Diferencia de conjuntos \

- La **diferencia de conjuntos**  $A \setminus B$  es el conjunto que resulta de eliminar de A cualquier elemento que esté en B.

$$A \setminus B = \{x : x \in A \wedge x \notin (A \cap B)\}$$



- Sea  $A = \{5, 7, 11\}$  y  $B = \{1, 3, 5\}$ ,  $A \setminus B = \{7, 11\}$

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A:	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0
B:	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0
C:	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0

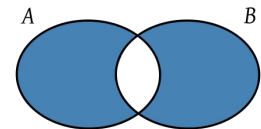
conjunto\_t A, B, C;

C = A & ~B;

# Operaciones: Diferencia simétrica de conjuntos $\Delta$

- La **diferencia simétrica**  $A \Delta B$  es el conjunto con todos los elementos que pertenecen, o bien a A, o bien a B, pero no a ambos a la vez.

$$A \Delta B = \{x : (x \in A \vee x \in B) \wedge x \notin (A \cap B)\}$$



- Sea  $A = \{5, 7, 11\}$  y  $B = \{1, 3, 5\}$ ,  $A \Delta B = \{1, 3, 7, 11\}$

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A:	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0
B:	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0
C:	0	0	0	0	1	0	0	0	1	0	0	0	1	0	1	0

```
conjunto_t A, B, C;
C = (A | B) & ~(A & B);
```

# Referencias

- ★ Olsson, M. (2018), “C++ 17 Quick Syntax Reference”, Apress. Disponible en PDF en la BBTK-ULL:  
<https://doi.org/10.1007/978-1-4842-3600-0>
- ★ Stroustrup, B. (2002), “El Lenguaje de Programación C++”, Addison Wesley.
- ★ C++ Syntax Highlighting (código en colores):  
[tohtml.com/cpp](http://tohtml.com/cpp)
- ★ Ecuaciones editadas con:  
[s1.daumcdn.net/editor/fp/service\\_nc/pencil/Pencil\\_chromestore.html](http://s1.daumcdn.net/editor/fp/service_nc/pencil/Pencil_chromestore.html)