Matemática Discreta

(Un curso de Matemática Discreta con Maxima)

Curso 2022-23

E.T.S. de Ingeniería Informática

UNIVERSIDAD
DE MÁLAGA

22 de enero de 2023

Matemática Discreta

Índice general

TEMA 1

Preliminares y teoría de números

Contenidos

- Lección 1.1: Los números enteros. Conjuntos numéricos y propiedades. Introducción al programa Maxima. Recursividad. Operador Sumatorio. Numéricos combinatorios y binomio de Newton.
- Lección 1.2: Aritmética entera. División euclídea. Sistemas de numeración posicionales. Divisibilidad y números primos. Teorema fundamental de la aritmética.
- Lección 1.3: Ecuaciones diofánticas. Algoritmo de Euclides. Ecuaciones diofánticas.
- Lección 1.4: Aritmética modular. Relación de congruencia. Aritmética modular. Teorema de Euler-Fermat. Aplicación a los criterios de divisibilidad. Congruencias lineales. Sistemas de congruencias lineales. Método de encriptación RSA.

Prerrequisitos: El punto de partida de los contenidos de este tema está en las matemáticas básicas de primaria. Se necesitará la realización con destreza de operaciones con números naturales, operaciones básicas con vectores y la resolución de ecuaciones e inecuaciones simples.

Objetivos: Los objetivos fundamentales de este tema son conocer los fundamentos de la aritmética entera y modular. Básicamente, aprender a plantear y resolver problemas cuyas soluciones deben ser números enteros.... De forma transversal, se aprenderán técnicas básicas de demostración para aprender a razonar y dar respuestas justificadas.

1.1. Los números enteros

1.1.1. Conjuntos numéricos y propiedades

Recordemos las propiedades básicas que verifican las operaciones de suma y producto entre números.

- *Asociatividad:* Si a, b y c son números, entonces

$$(a+b)+c = a+(b+c), \qquad (a\cdot b)\cdot c = a\cdot(b\cdot c).$$

- *Existencia de elemento neutro:* el número 0 es el elemento neutro para la suma, es decir, para todo número a

$$a+0 = 0+a = a$$

- *Existencia de elemento unidad:* el número 1 es la unidad para el producto, es decir, para todo número a

$$a\cdot 1 = 1\cdot a = a$$

- *Existencia de elemento opuesto:* el número $-a$ es el opuesto de a respecto de la suma, es decir, $a+(-a) = (-a)+a = 0$ para todo número a.

- *Existencia de elemento inverso:* el número $a^{-1} = \frac{1}{a}$ es el inverso de a respecto del producto, es decir, $a\cdot\frac{1}{a} = \frac{1}{a}\cdot a = 1$, para todo número $a \neq 0$.

- *Conmutatividad:* Todos los números a y b verifican

$$a+b = b+a, \qquad a\cdot b = b\cdot a.$$

- *Distributividad:* Si a, b y c son números, entonces

$$a\cdot(b+c) = a\cdot b + a\cdot c, \quad (b+c)\cdot a = b\cdot a + c\cdot a.$$

  Si aplicamos estas igualdades de derecha a izquierda, decimos que *sacamos un factor común.*

- *Ley de tricotomía*: Cada par de números a y b verifica una y solo una de las siguientes relaciones:

$$a = b \qquad a < b \qquad b < a$$

  Esta propiedad también se enuncia diciendo que el orden es *total.*

- *La suma es cerrada para el orden:* Si $a > b$, entonces $a + c > b + c$.

- *El producto es cerrado para el orden:* Si $a > b$, $c > 0$, entonces $a \cdot c > b \cdot c$.

Estas propiedades deben ser conocidas y se habrán usado para resolver ecuaciones e inecuaciones y para simplificar expresiones algebraicas en la resolución de múltiples ejercicios. Es conveniente acostumbrarse a sus denominaciones y entender su significado. Los números reales verifican todas las propiedades, pero si nos restringimos a subconjuntos numéricos como los enteros o naturales, algunas propiedades no se verifican en general.

TEOREMA 1.1.1 *La suma en el conjunto de los números naturales $\mathbb{N}$, verifica las propiedades: asociatividad, existencia de elemento neutro y conmutatividad. El producto en el conjunto de los números naturales verifica las propiedades: asociatividad, conmutatividad y distributividad respecto de la suma. Además, la suma y el producto son cerrado para el orden en $\mathbb{N}$.*

TEOREMA 1.1.2 *La suma en el conjunto de los números enteros $\mathbb{Z}$, verifica las propiedades: asociatividad, existencia de elemento neutro, existencia de elemento opuesto y conmutatividad. El producto en el conjunto de los números enteros verifica las propiedades: asociatividad, conmutatividad y distributividad respecto de la suma. Además, la suma y el producto son cerrado para el orden en $\mathbb{Z}$.*

TEOREMA 1.1.3 *La suma en el conjunto de los números racionales $\mathbb{Q}$, verifica las propiedades: asociatividad, existencia de elemento neutro, existencia de elemento opuesto y conmutatividad. El producto en el conjunto de los números racionales verifica las propiedades: asociatividad, existencia de elemento inverso, conmutatividad y distributividad respecto de la suma. Además, la suma y el producto son cerrados para el orden en $\mathbb{Q}$.*

Aunque ni en $\mathbb{N}$ ni en $\mathbb{Z}$ se verifica la propiedad de existencia de elementos inverso, sí se verifica la propiedad de cancelación.

TEOREMA 1.1.4 (CANCELACIÓN) *Si $n \neq 0$ y $n \cdot m = n \cdot k$, entonces $m = k$.*

También es conveniente recordar las propiedades de la relación de orden entre números y del valor absoluto.

1. *Reflexiva:* $n \leq n$
2. *Antisimétrica:* Si $n \leq m$, y $m \leq n$, entonces $n = m$
3. *Transitiva:* Si $n \leq m$, y $m \leq k$,entonces $n \leq k$
4. $|n| \geq 0$
5. $|n| = 0$, si y solo si $n = 0$
6. $|n \cdot m| = |n| \cdot |m|$
7. $|n + m| \leq |n| + |m|$

1.1.2. Software matemático

Este curso se va a desarrollar integrando una herramienta informática de cálculo matemático: `Maxima`. Vamos a aprender a realizar algunos cálculos con ella y también a definir o describir algunas de las funciones y algoritmos que aprendemos en el curso.

El programa `Maxima` es un proyecto de software libre escrito en el lenguaje `Lisp`. Aunque su interface original se basa en línea de comandos, existen varios entornos gráficos que se comunican con él para ofrecer sistemas más fáciles de usar y con resultados visuales más amigables. Entre ellos, recomendamos `wxMaxima`, que posiblemente recoge las características más evolucionadas. También es posible utilizar este sistema via web, sin la necesidad de instalarlo en un ordenador.

`http://maxima.sourceforge.net/`
`https://wxmaxima-developers.github.io/wxmaxima/`
`http://maxima.cesga.es`

Utilizaremos los menús y los paneles de `wxMaxima` para hacer la mayoría de las operaciones y cálculos, aunque en algunos casos tendremos que escribir los operadores. Dado que estamos ante un lenguaje funcional, la forma de escribir los operadores y funciones será muy similar al lenguaje matemático. Por ejemplo, en la siguiente tabla vemos algunas expresiones y funciones matemáticas habituales:

`3*x*y^3-2*(x-y)^2`	$3xy^3 - 2(x-y)^2$
`3-x/(x^2-1)`	$3 - \dfrac{x}{x^2-1}$
`sqrt(x)`	$\sqrt{x}$
`exp(x)`	$\exp(x) = e^x$
`%e`	Número e
`%pi`	Número π
`%i`	Unidad imaginaria
`log(x)`	$\log(x) = \ln(x) = \mathrm{L}(x)$ (logaritmo neperiano)
`sin(x)`	$\mathrm{sen}(x)$
`cos(x)`	$\cos(x)$
`tan(x)`	$\mathrm{tg}(x)$
`abs(x)`	$\lvert x \rvert$ (valor absoluto)
`floor(x)`	$\lfloor x \rfloor$ (parte entera, función suelo)
`ceiling(x)`	$\lceil x \rceil$ (función techo)

Una de las grandes ventajas de este tipo de programas es la posibilidad de definir etiquetas o parámetros para representar expresiones matemáticas a lo largo de un desarrollo y simplificar el trabajo, esto lo hacemos utilizando ":". En el siguiente

ejemplo, utilizamos la etiqueta `pol` para representar un polinomio de grado 2 y utilizamos el operador `solve` para resolver la ecuación polinómica determinada por eso polinomio:

```
(%i1) pol: x^2+3*x+2$
(%i2) solve(pol=0,x);
```

$$[x = -2, x = -1]$$

También podemos asignar nombres a una función tal y como hacemos habitualmente en matemáticas escribiendo $f(x) = x^2+3x+2$ y de forma que posteriormente podemos usar $f(2)$ para representar el valor de la función cuando la evaluamos en 2. Para hacer esto en `Maxima`, tenemos que usar ":=" o el operador `define`

```
(%i1) f(x):=x^2+3*x+2$
(%i2) f(10);
```

132

Como hemos visto, las líneas de entrada pueden terminar en ";" o con "$". En `wxMaxima` no es necesario escribir nada al final de la línea, ya que el sistema lo añade de forma automática. Si terminamos la línea con el símbolo de dolar, no se mostrará la salida; esto es útil cuando estamos haciendo definiciones. Por otra parte, en este texto vamos a escribir en rojo los operadores y funciones predefinidos en `Maxima`.

1.1.3. Recursividad

La recursividad es un método de resolución de problemas o de implementación de soluciones y es un principio fundamental en las ciencias de la computación. Resolver un problema mediante recursión significa que la solución depende de otras soluciones que son instancias del mismo problema. La mayoría de los lenguajes de programación soportan la recursividad, permitiendo que una función se llame a sí misma. En los lenguajes imperativos, se usan estructuras como `while` y `for` para realizar tareas repetitivas. Los lenguajes de programación funcionales no suelen definir bucles, sino que posibilitan la recursión llamando código de forma repetitiva. Un ejemplo típico de función recursiva es el operador factorial.

Definición 1.1.5 (Factorial) *Definimos el* factorial *de un número natural n, denotado por $n!$, como sigue:*

$$0! = 1$$
$$n! = (n-1)! \cdot n \quad \textit{para todo } n \geq 1$$

Como vemos, la definición llama al operador que estamos definiendo, pero aplicado a un número menor, hasta llegar a un *caso base*, en este caso 0!. También lo

podemos escribir como sigue:

$$n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n, \quad \text{para todo } n \geq 1$$

Ejemplo 1.1.6

$$0! = 1, \qquad 1! = 1, \qquad 2! = 1 \cdot 2 = 2, \qquad 3! = 1 \cdot 2 \cdot 3 = 6$$

$$10! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot 10 = 3\,628\,800$$ □

Ejemplo con Maxima 1.1.7 En `Maxima` podemos utilizar `if-then-else` para definir funciones por casos o ramas. Esta misma estructura se puede utilizar para definir funciones recursivas si la función que estamos definiendo se llama a sí misma. Como vemos en este ejemplo, podemos trasladar literalmente la definición recursiva del factorial al lenguaje de `Maxima`.

```
(%i1) fact(n):= if n=0 then 1 else n*fact(n-1)$
(%i2) fact(5);
```

120

No obstante, el operador factorial está predefinido en `Maxima` con el mismo símbolo que utilizamos en matemáticas.

```
(%i3) 5!;
```

120 □

La teoría de la computabilidad ha demostrado que las dos formas de abordar la recursividad en lenguajes imperativos y funcionales, son matemáticamente equivalentes, es decir, en ambos se pueden resolver los mismos tipos de problemas. Usando la recursión regular, cada llamada recursiva inserta una entrada en la pila de llamadas. De esta forma, cuando se completa la recursión, la aplicación tiene que quitar cada entrada por completo hacia abajo. Esta ineficiencia se puede corregir con la recursión de cola, es decir, haciendo llamadas de subrutinas como acción final de un procedimiento. En los lenguajes funcionales, esto se consigue añadiendo un argumento adicional a la función en donde se "guardan" las evaluaciones anteriores; es lo que se denomina *recursión de cola* o *tail recursion* en inglés.

Ejemplo con Maxima 1.1.8 En este ejemplo, vamos a definir la función factorial con recursión de cola. El operador tiene dos argumentos, de forma que en el segundo se acumula la salida en cada paso recursivo. De esta forma, `fact_tail`$(n, 1) = n!$.

```
(%i1) fact_tail(n,s):= if n=0 then s else fact_tail(n-1,n*s)$
(%i2) fact_tail(5,1);
```

120

Una opción muy interesante de `Maxima` es la que da la posibilidad de ver la "traza" de los operadores definidos por el usuario: `trace(<op>)` provocará que a partir de ese momento la salida del operador `<op>` muestre los cálculos intermedios.

```
(%i3) trace(fact_tail)$
(%i4) fact_tail(5,1);
1" Enter "fact_tail" "[5,1]
 .2" Enter "fact_tail" "[4,5]
 ..3" Enter "fact_tail" "[3,20]
 ...4" Enter "fact_tail" "[2,60]
 ....5" Enter "fact_tail" "[1,120]
 .....6" Enter "fact_tail" "[0,120]
 .....6" Exit  "fact_tail" "120
 ....5" Exit  "fact_tail" "120
 ...4" Exit  "fact_tail" "120
 ..3" Exit  "fact_tail" "120
 .2" Exit  "fact_tail" "120
1" Exit  "fact_tail" 120
```

120

Como podemos ver el factorial de 5 se calcula durante el proceso de evaluación y en la fase de salida no se realiza ningún cálculo. □

1.1.4. Números combinatorios

Definición 1.1.9 (Números combinatorios) *Sean n y k dos números naturales tales que $0 \leq k \leq n$. Se define el* número combinatorio $\binom{n}{k}$, *que se lee "n sobre k", como*

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!}$$

Proposición 1.1.10 *Para todo $n \in \mathbb{N}$ y todo $k \in \mathbb{N}$:*

$$\binom{n}{k} = \binom{n}{n-k}$$

Ejemplo 1.1.11

- $\binom{10}{7} = \frac{10!}{7! \cdot 3!} = \frac{10 \cdot 9 \cdot 8 \cdot \cancel{7!}}{\cancel{7!} \cdot 3!} = \frac{10 \cdot 9 \cdot 8}{3!} = \frac{10 \cdot 9 \cdot 8}{3 \cdot 2} = 10 \cdot 3 \cdot 4 = 120$

- $\binom{0}{0} = \dfrac{0!}{0! \cdot 0!} = 1$
- $\binom{n}{0} = \dfrac{n!}{0! \cdot n!} = \dfrac{n!}{n!} = 1$
- $\binom{n}{n} = \dfrac{n!}{n! \cdot 0!} = \dfrac{n!}{n!} = 1$
- $\binom{n}{k} = \dfrac{n!}{k! \cdot (n-k)!} = \binom{n}{n-k}$ □

La forma habitual de calcular los números combinatorios será la siguiente: expandimos parcialmente el factorial del numerador y simplificamos con uno de los factores del denominador. Obtenemos así una igualdad que puede tomarse como definición de los números combinatorios y que es válida incluso para valores reales de n.

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!} = \frac{n(n-1)\dots(n-k+1) \cdot \cancel{(n-k)!}}{k! \cdot \cancel{(n-k)!}} = \\ = \frac{n}{k} \cdot \frac{n-1}{k-1} \cdot \frac{n-2}{k-2} \cdots \frac{n-k+2}{2} \cdot \frac{n-k+1}{1}$$

Podemos escribir esta última igualdad como

$$\binom{n}{k} = \frac{n-k+1}{1} \cdot \frac{n-k}{2} \cdots \frac{n}{k}$$

Es decir empezando en $\dfrac{n-k+1}{1}$, vamos añadiendo factores que se obtienen incrementando numerador y denominador hasta llegar a $\dfrac{n}{k}$. Esto es lo que utilizamos para definir el operador binomial con recursión de cola .

Ejemplo con Maxima 1.1.12 La siguiente función define de forma recursiva de los números combinatorios:

```
(%i1) binom(n,k):= binom_tail(k,n-k+1,1,1)$
(%i2) binom_tail(k,d,c,s):= if k=1 then s*d/c else
                            binom_tail(k-1,d+1,c+1,s*d/c)$
(%i3) binom(10,5);
```

252

La variable k funciona como un contador que determina el número de factores d y c del numerador y denominador respectivamente que tenemos que añadir. Además, cada resultado parcial que se acumula en la variable s es un número combinatorio y, por lo tanto, es un número natural.

```
(%i4) trace(binom_tail)$
(%i5) binom(7,3);

1 Introducir binom_tail [3,5,1,1]
 .2 Introducir binom_tail [2,6,2,5]
 ..3 Introducir binom_tail [1,7,3,15]
 ..3 Salir binom_tail 35
 ...

                              35
```

Naturalmente, `Maxima` incluye un operador primitivo para el cálculo de los números combinatorios.

```
(%i4) binomial(10,5);

                              252
```

□

La siguiente propiedad es la más importante de los números combinatorios, siendo el fundamento del *triángulo de Tartaglia-Pascal*.

TEOREMA 1.1.13 (DE PASCAL) *Para todo $n \in \mathbb{N}$ y todo $k \in \mathbb{N}$:*

$$\binom{n}{k} + \binom{n}{k+1} = \binom{n+1}{k+1}$$

EJEMPLO 1.1.14 Vamos a demostrar el Teorema de Pascal. Para ello, solo utilizamos la suma de fracciones igualando denominadores y la simplificación de un numerador sacando factor común.

$$\begin{aligned}
&\binom{n}{k} + \binom{n}{k+1} = \\
&\quad = \frac{n\cdot(n-1)\cdots(n-k+1)}{k!} + \frac{n\cdot(n-1)\cdots(n-k+1)\cdot(n-k)}{(k+1)!} = \\
&= \frac{(k+1)\cdot n\cdot(n-1)\cdots(n-k+1)}{(k+1)\cdot k!} + \frac{n\cdot(n-1)\cdots(n-k+1)(n-k)}{(k+1)!} = \\
&\quad = \frac{(k+1+n-k)\cdot n\cdot(n-1)\cdots(n-k+1)}{(k+1)!} = \\
&\quad = \frac{(n+1)\cdot n\cdot(n-1)\cdots(n-k+1)}{(k+1)!} = \binom{n+1}{k+1}
\end{aligned}$$

□

El Teorema de Pascal permite calcular los números combinatorios usando una representación geométrica que se donomina *triángulo de Tartaglia* o *triángulo de Tartaglia-Pascal*. Construimos este triángulo colocando en el vértice superior, el número $\binom{0}{0}$ y debajo de él colocamos los números $\binom{1}{0}$ y $\binom{1}{1}$; formamos así un primer

triángulo con solo tres números. A partir de aquí, vamos añadiendo nuevas filas usando la siguiente regla: debajo de cada par de números, colocamos su suma:

$$\begin{array}{ccc} \binom{n}{k} & & \binom{n}{k+1} \\ & \searrow \quad \swarrow & \\ & \binom{n}{k}+\binom{n}{k+1} & \end{array} \overset{\text{T.P.}}{=} \begin{array}{ccc} \binom{n}{k} & & \binom{n}{k+1} \\ & \searrow \quad \swarrow & \\ & \binom{n+1}{k+1} & \end{array}$$

Adicionalmente, cada fila se comienza con $\binom{n}{0} = 1$ y se termina con $\binom{n}{n} = 1$. Vemos a continuación el triángulo resultante hasta la quinta fila, a la izquierda con los números combinatorios indicados y a la derecha con los valores resultantes.

$$\begin{array}{ccccccccccc} & & & & \binom{0}{0} & & & & & & \\ & & & \binom{1}{0} & & \binom{1}{1} & & & & & \\ & & \binom{2}{0} & & \binom{2}{1} & & \binom{2}{2} & & & & \\ & \binom{3}{0} & & \binom{3}{1} & & \binom{3}{2} & & \binom{3}{3} & & & \\ \binom{4}{0} & & \binom{4}{1} & & \binom{4}{2} & & \binom{4}{3} & & \binom{4}{4} & & \\ \binom{5}{0} & & \binom{5}{1} & & \binom{5}{2} & & \binom{5}{3} & & \binom{5}{4} & & \binom{5}{5} \end{array} \qquad \begin{array}{ccccccccccc} & & & & 1 & & & & & & \\ & & & 1 & & 1 & & & & & \\ & & 1 & & 2 & & 1 & & & & \\ & 1 & & 3 & & 3 & & 1 & & & \\ 1 & & 4 & & 6 & & 4 & & 1 & & \\ 1 & & 5 & & 10 & & 10 & & 5 & & 1 \end{array}$$

1.1.5. Operador sumatorio

El operador $\sum$ o *sumatorio* se utiliza para expresar sumas:

$$\sum_{k=m}^{n} f(k) = f(m) + f(m+1) + \cdots + f(n)$$

Los sumandos están determinados en función de una variable o índice, que tomará todos los valores entre dos números naturales m y n tales que $m \leq n$. Este operador también es frecuente en los lenguajes de programación, en los que toma una sintaxis similar a la que usa `Maxima`:

Ejemplo con Maxima 1.1.15 Vemos en este ejemplo como calcular la siguiente suma en `Maxima`

$$\sum_{k=1}^{100} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{100}$$

```
(%i1) sum(1/k,k,1,100);
```

$$\frac{14466636279520351160221518043104131447711}{2788815009188499086581352357412492142272}$$

```
(%i2) float(%);
```

5.187377517639621

En este ejemplo, hemos usado la variable %, que representa al resultado del último cálculo efectuado. Además, observamos que por defecto, el programa realiza los cálculos de forma exacta o simbólica y si queremos obtener los resultados aproximados debemos pedirlo explícitamente; esto podemos hacerlo con el operador `float` o con la opción `numer`.

```
(%i3) sum(1/k,k,1,100),numer;
```

5.187377517639621 □

El operador sumatorio se usará en distintas asignaturas, por lo que es muy conveniente aprender a manejarlo correctamente. Vemos a continuación algunos ejemplos sencillos, pero que ayudarán a entender algunas propiedades sus propiedades.

Ejemplo 1.1.16

1. La variable utilizada como índice de cada sumando no influye en el resultado y podremos cambiarla por la letra que deseemos siempre que no interfiera en el resto del problema. Por ejemplo, en los sumatorios siguientes utilizamos índices distintos para expresar la misma suma:

$$\sum_{k=1}^{10} k = 1+2+3+4+5+6+7+8+9+10$$
$$\sum_{i=1}^{10} i = 1+2+3+4+5+6+7+8+9+10$$

2. La expresión $\sum_{k=1}^{10} 2$ tiene 10 sumandos, pero ninguno depende de k, todos son iguales a 2, y por lo tanto:

$$\sum_{k=1}^{10} 2 = 2+2+2+2+2+2+2+2+2+2 = 10\cdot 2$$

3. Un sumatorio no es más que una suma, y por lo tanto le podemos aplicar las propiedades que hemos recordado antes para esta operación. Por ejemplo, la siguiente igualdad no es más que la aplicación de la propiedad asociativa:

$$\begin{array}{rcccc} \displaystyle\sum_{k=1}^{8} k = & \left(\displaystyle\sum_{k=1}^{4} k\right) & + & \left(\displaystyle\sum_{k=5}^{8} k\right) \\ 1+2+3+4+5+6+7+8 = & (1+2+3+4) & + & (5+6+7+8) \end{array}$$

4. De la misma forma, si la expresión que hay dentro del sumatorio es también una suma, las propiedades de asociatividad y conmutatividad nos permitirán manipulaciones como la mostrada en el siguiente ejemplo:

$$\begin{array}{rcccc} \displaystyle\sum_{k=1}^{4}(k+1) = & \left(\displaystyle\sum_{k=1}^{4}k\right) & + & \left(\displaystyle\sum_{k=1}^{4}1\right) \\ (1+1)+(2+1)+(3+1)+(4+1) = & (1+2+3+4) & + & (1+1+1+1) \end{array}$$

5. La distributividad es otra propiedad de la suma y también admite una formulación muy conveniente con los sumatorios.

$$\sum_{k=1}^{5}2k = 2\sum_{k=1}^{5}k$$
$$2\cdot 1+2\cdot 2+2\cdot 3+2\cdot 4+2\cdot 5 = 2(1+2+3+4+5)$$ □

Debemos asegurarnos de que todas las transformaciones que realicemos estén respaldadas por las propiedades de la suma y el producto, tal y como hemos hecho en los apartados del ejemplo anterior. En el ejemplo siguiente recogemos algunos errores bastante frecuentes en la manipulación de sumatorios.

Ejemplo 1.1.17

1. $\displaystyle\sum_{k=1}^{5}k^2 \neq \left(\sum_{k=1}^{5}k\right)^2$. Estas dos expresiones son distintas, ya que, en general, el cuadrado de una suma no es igual a la suma de los cuadrados

$$1^2+2^2+3^3+4^2+5^2 \neq (1+2+3+4+5)^2$$

2. Hemos visto anteriormente que, gracias a la propiedad distributiva, podemos sacar un factor común a todos los sumandos del sumatorio. Sin embargo:

$$\sum_{k=1}^{5}k(k+1) \neq k\left(\sum_{k=1}^{5}(k+1)\right)$$

La variable k toma un valor distinto en cada sumando y por lo tanto no se puede considerar factor común a todos ellos. Debemos pensar siempre que la variable que funciona como índice solo tiene sentido dentro del sumatorio. □

1.1.6. Binomio de Newton

La siguiente fórmula hace uso de los operadores que hemos definido en las secciones anteriores y expresa la forma expandida de cualquier potencia de un binomio.

TEOREMA 1.1.18 (FÓRMULA DEL BINOMIO DE NEWTON) *Para todo par de números a y b y todo número natural $n \geq 2$, se verifica que*

$$(a+b)^n = \sum_{j=0}^{n} \binom{n}{j} a^{n-j} b^j$$

EJEMPLO 1.1.19

- $(x-y)^2 = \binom{2}{0} x^2 (-y)^0 + \binom{2}{1} x(-y) + \binom{2}{2} x^0 (-y)^2 = x^2 - 2xy + y^2$
- $(s+t)^3 = \binom{3}{0} s^3 t^0 + \binom{3}{1} s^2 t + \binom{3}{2} s t^2 + \binom{3}{3} s^0 t^3 = s^3 + 3s^2 t + 3st^2 + t^3$
- $(z-2)^6 = z^6 - 12z^5 + 60z^4 - 160z^3 + 240z^2 - 192z + 64$
- $2^n = (1+1)^n = \binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \ldots + \binom{n}{n-1} + \binom{n}{n} = \sum_{j=0}^{n} \binom{n}{j}$ □

EJEMPLO CON MAXIMA 1.1.20 El operador `expand` elimina los paréntesis en cualquier expresión para obtener su forma expandida y en cierto sentido simplificada. En particular, podremos ver el resultado de aplicar la fórmula de Newton si lo aplicamos a la potencia de un binomio.

```
(%i1) expand((x-2)^7);
```

$$x^7 - 14x^6 + 84x^5 - 280x^4 + 560x^3 - 672x^2 + 448x - 128$$ □

1.2. Aritmética Entera

1.2.1. División euclídea

TEOREMA 1.2.1 (DIVISIÓN EUCLÍDEA) *Sean $n, m \in \mathbb{N}$, con $m > 0$. Entonces existen $q, r \in \mathbb{N}$, únicos, tales que*

$$n = m \cdot q + r \qquad y \qquad 0 \leq r < m$$

El número q se denomina *cociente* de la división de n entre m. El número r se denomina *resto* de la división de n entre m. La demostración de este teorema se basa en el algoritmo habitual de la división: se busca el mayor natural q tal que $n - m \cdot q$ es positivo; necesariamente, este número es menor que m. Básicamente, este es el proceso que seguimos al realizar el cálculo de la división a mano con el algoritmo que conocemos desde primaria.

EJEMPLO CON MAXIMA 1.2.2 Para determinar cociente y el resto con `Maxima`, usamos los operadores `quotient` y `remainder` respectivamente.

```
(%i1) quotient(231,17);
```

$$13$$

```
(%i2) remainder(231,17);
```

$$10$$ □

1.2.2. Sistemas de numeración posicionales

Un *sistema de numeración* es un conjunto de símbolos y de reglas mediante las cuales podemos representar los números enteros. El sistema de numeración que utilizamos habitualmente es un sistema *posicional* definido sobre la base de numeración 10. Esto significa que utilizamos diez símbolos que representan las cantidades desde cero hasta nueve, pero que al escribirlas de forma concatenada toman un valor que depende de la posición en la cadena. Por ejemplo, la cadena de dígitos 34735 representa al número

$$3 \cdot 10^4 + 4 \cdot 10^3 + 7 \cdot 10^2 + 3 \cdot 10^1 + 5 \cdot 10^0$$

Representaciones similares se pueden definir utilizando cualquier número como base. De hecho, diferentes civilizaciones han usado otras bases distintas de 10, como los babilonios, que usaron la base seis o los mayas que usaron la base veinte.

En la actualidad, el uso de distintas bases de numeración tiene importantes aplicaciones en computación. Por ejemplo, la base dos es la habitual para representar los datos informáticos a bajo nivel y las bases ocho y dieciséis se usan en criptografía y en comunicaciones.

TEOREMA 1.2.3 *Sea b un entero positivo, $b > 1$. Entonces cada entero positivo n se puede escribir de forma única como combinación lineal de potencias de b*

$$n = a_k \cdot b^k + a_{k-1} \cdot b^{k-1} + \cdots + a_1 \cdot b^1 + a_0 \cdot b^0$$

en donde cada a_j es un entero tal que $0 \leq a_j \leq b-1$ y $a_k \neq 0$.

Atendiendo a este teorema, el número n se escribe como

$$n = (a_k \ a_{k-1} \ \ldots \ a_1 \ a_0)_{(b}$$

Omitiremos los paréntesis si no lleva a confusión, y no añadiremos el indicador de la base cuando esta sea la base decimal. Habitualmente, se utiliza un único símbolo para representar cada número entre 0 y $b-1$. Si la base b es menor que 10, estos símbolos son los dígitos que habitualmente usamos y si la base es mayor de 10, utilizamos letras para representar los números mayores a 10. Por ejemplo, en la base 16, o hexadecimal, se utilizan los símbolos 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, en donde A = 10, B = 11, C = 12, D = 13, E = 14 y F = 15.

Ejemplo 1.2.4 Vamos a determinar el número representado por la secuencia 1B03A3 en la base hexadecimal pasándolo a la base decimal.

$$1B03A3_{(16} = 1 \cdot 16^5 + 11 \cdot 16^4 + 0 \cdot 16^3 + 3 \cdot 16^2 + 10 \cdot 16 + 3 = 1\,770\,403 \qquad \square$$

La demostración del teorema anterior consiste en establecer el procedimiento para encontrar los números a_j que formarán la representación. Estos números son los restos que se obtienen al aplicar sucesivamente el algoritmo de la división, tal y como vemos en el siguiente ejemplo

Ejemplo 1.2.5 Vamos a escribir el número $n = 3785$ en base $b = 7$; naturalmente, utilizamos los dígitos 0, 1, 2, 3, 4, 5, 6 para construir las secuencias en esta base de numeración tenemos que:

$$\begin{aligned} 3785 &= 7 \cdot 540 + 5 \implies a_0 = 5 \\ 540 &= 7 \cdot 77 + 1 \implies a_1 = 1 \\ 77 &= 7 \cdot 11 + 0 \implies a_2 = 0 \\ 11 &= 7 \cdot 1 + 4 \implies a_3 = 4 \\ 1 &= 7 \cdot 0 + 1 \implies a_4 = 1 \end{aligned}$$

El cociente de la última división es 0 y por lo tanto no podemos seguir dividiendo, así que el resto de esa división es el último dígito de la representación. En consecuencia

$$3785 = 14015_{(7}$$

No vamos a hacer la demostración formal del teorema, pero sí vamos a hacer la comprobación en este ejemplo de que el proceso de divisiones encadenadas genera la representación en la base dada. También podemos observar que la secuencia de dígitos obtenida es única, ya que el cociente y el resto en la división entera son únicos para cada pareja dividendo-divisor. Hacemos la comprobación sustituyendo los números de la izquierda, en la secuencia de divisiones, por la expresión de la derecha:

$$\begin{aligned} 3785 &= 7 \cdot 540 + 5 = \\ &= 7 \cdot (7 \cdot 77 + 1) + 5 = \\ &= 7 \cdot (7 \cdot (7 \cdot 11 + 0) + 1) + 5 = \\ &= 7 \cdot (7 \cdot (7 \cdot (7 \cdot 1 + 4) + 0) + 1) + 5 \end{aligned}$$

Si ahora quitamos los paréntesis sin operar las potencias de 7, obtenemos la expresión del número como combinación lineal de potencias de 7:

$$\begin{aligned} 3785 &= 7 \cdot (7 \cdot (7 \cdot (7 \cdot 1 + 4) + 0) + 1) + 5 = \\ &= 7 \cdot (7 \cdot (7^2 \cdot 1 + 7 \cdot 4 + 0) + 1) + 5 = \\ &= 7 \cdot (7^3 \cdot 1 + 7^2 \cdot 4 + 7 \cdot 0 + 1) + 5 = \\ &= 7^4 \cdot 1 + 7^3 \cdot 4 + 7^2 \cdot 0 + 7 \cdot 1 + 5 = \\ &= 1 \cdot 7^4 + 4 \cdot 7^3 + 0 \cdot 7^2 + 1 \cdot 7 + 5 \end{aligned} \qquad \square$$

`Maxima` no dispone de ningún operador predefinido para convertir números entre distintas bases, pero es fácil hacerlo usando los operadores disponibles sobre listas. Cuando tratamos con números escritos con dígitos en un sistema posicional, estos números deben ser entendidos como listas:

$$3785 \rightsquigarrow [3, 7, 8, 5]$$

Las *listas* son una estructura de datos básica en programación y todos los lenguajes incluyen un conjunto de operadores básicos para manipularlas. Como hemos visto arriba, la forma habitual de representar las listas es delimitando sus elementos, que estarán separados por comas, entre corchetes. A diferencia de los conjuntos, el orden en el que están colocados los elementos en la lista es deterninante de la misma y puede contener elementos repetidos:

$$[1, 2, 3] \neq [3, 2, 1]; \qquad [2, 2, 1, 3] \neq [2, 1, 3]$$

`Maxima` incluye distintos operadores para trabajar con listas; a continuación vemos los más básicos, los que definen la estructura, y a lo largo del curso aprenderemos algunos más:

`cons`$(x, [x_1, \ldots, x_n]) = [x, x_1, \ldots, x_n]$

`first`$([x_1, \ldots, x_n]) = x_1$

`rest`$([x_1, x_2, \ldots, x_n]) = [x_2, \ldots, x_n]$

Si x es una lista, $x[i]$ representa al i-ésimo elemento de x.

Como decíamos arriba, la representación de los números usando una base de numeración es esencialmente una representación mediante listas. Pero habitualmente, representamos los dígitos por un único símbolo, de forma que podemos prescindir de los corchetes y las comas:

$$31AB_{(16} \rightsquigarrow [3, 1, 10, 11]$$

Para trabajar con `Maxima`, sin embargo, usaremos la notación completa de las listas y podremos utilizar números mayores que 10 como dígitos.

Para definir la conversión de un número en una base distinta de 10 a la base 10, vamos a utilizar vamos a utilizar el operador `lreduce`, *reducción por la izquierda*. Si $f(x, y)$ es una función de dos argumentos y ℓ es una lista de más de dos elementos, `lreduce`(f, ℓ) aplica la función f a los elementos de la lista agrupándolos desde la izquierda:

$$\texttt{lreduce}(f,[a,b,c]) = f(f(a,b),c)$$

Si observamos el ejemplo de más arriba, vemos que para convertir un número de base 7 a base 10, hemos usado la función $f(x,y) = 7x+y$ agrupando los dígitos de dos en dos desde la izquierda. Por lo tanto, podemos utilizar las siguientes instrucciones:

```
(%i1) b7(x,y):=7*x+y$
(%i2) lreduce(b7,[1,4,0,1,5]);
```

3785

También podemos hacerlo usando el operador sumatorio

```
(%i1) a: [1,4,0,1,5])$
(%i2) sum(a[k]*7^(5-k),k,1,5);
```

3785

Hemos visto que el proceso inverso consiste en realizar una serie de divisiones sucesivas, lo que podemos describir fácilmente con una definición recursiva:

```
(%i1) base(b,N,res):= if N=0 then res
          else base(b,quotient(N,b),cons(remainder(N,b),res))$
(%i2) base(7,3785,[]);
```

$$[1,4,0,1,5]$$

1.2.3. Divisibilidad y números primos

Definición 1.2.6 *Sean $n,m \in \mathbb{Z}$ con $m \neq 0$. Se dice que m* divide *a n si existe $q \in \mathbb{Z}$ tal que $n = m \cdot q$. En tal caso, escribimos $m \mid n$, o bien $n = \dot{m}$. Decimos igualmente que m es* divisor *de n o que n es un* múltiplo *de m.*

Ejemplo 1.2.7

- $6 \mid 192$, ya que $192 = 6 \cdot 32$
- $11 \mid 2310$, ya que $2310 = 11 \cdot 210$
- $8 \mid (-24)$, ya que $24 = 8 \cdot (-3)$ □

Teorema 1.2.8 *La relación de divisibilidad verifica las siguiente propiedades. Si $n,m,k \in \mathbb{Z}$:*

1. Reflexiva: $n \mid n$

2. Transitiva: *Si* $n \mid m$, *y* $m \mid k$, *entonces* $n \mid k$

3. Antisimétrica (solo en $\mathbb{N}$): *Si* $0 \leq n$, $0 \leq m$, $n \mid m$ *y* $m \mid n$, *entonces* $n = m$

La relación de divisibilidad no es antisimétrica en $\mathbb{Z}$, ya que $n \mid (-n)$, y $(-n) \mid n$. En general, para enteros arbitrarios se verifica que, si $n \mid m$ y $m \mid n$, entonces $n = m$, o bien $n = -m$.

El siguiente teorema recoge las propiedades algebraicas de la divisibilidad, es decir, su comportamiento con respecto a las operaciones de suma, resta y producto.

Teorema 1.2.9 *Sean* $n, m, k \in \mathbb{Z}$. *Entonces:*

1. $1 \mid n$ *y* $(-1) \mid n$

2. $n \mid n$ *y* $n \mid (-n)$

3. *Si* $k \mid n$ *y* $k \mid m$, *entonces* $k \mid (n + m)$, *y* $k \mid (n - m)$

4. *Si* $n \mid m$, *entonces* $n \mid (k \cdot m)$ *para todo* $k \in \mathbb{Z}$

5. *Si* $k \mid n$ *y* $k \mid m$, *entonces* $k \mid (s \cdot n + t \cdot m)$, *para todo* $s, t \in \mathbb{Z}$

Ejemplo 1.2.10 Dado que 27 y 39 son múltiplos de 3, podemos afirmar que

$$3 \mid (27 \cdot 13721 + 39 \cdot 7451)$$

sin necesidad de realizar las operaciones de la derecha.

Es importante tener en cuenta que no podemos aplicar esa propiedad en sentido contrario. En el ejemplo anterior, ninguno de los números 27, 13721, 39, 7451 es par, sin embargo el siguiente sí lo es

$$27 \cdot 13721 + 39 \cdot 7451 = 661056$$ □

Definición 1.2.11 *Un entero positivo* $p \neq 1$ *se dice que es* primo, *si sus únicos divisores positivos son 1 y p. Los números que no son primos se denominan* compuestos.

Naturalmente, si un número n es compuesto, siempre es posible encontrar un número primo que lo divide.

Durante mucho tiempo se mantuvo la duda de si el conjunto de los números primos era finito o no, hasta que Euclides dio una sencilla demostración de que hay infinitos números primos.

Teorema 1.2.12 (Euclides) *El conjunto de números primos es infinito.*

La demostración de Euclides de este resultado solo hace uso de las propiedades algebraicas. Si $p_1, \ldots, p_n$ son todos los primos menores que p_n, entonces $q = 1 + p_1 \cdot p_2 \cdot \cdots \cdot p_n$ es otro número primo estrictamente mayor que p_n, o bien es divisible por un número primo necesariamente mayor que p_n. Esto es cierto porque el número $q = 1 + p_1 \cdot p_2 \cdot \cdots \cdot p_n$ no es divisible por ninguno de los múmeros $p_1, \ldots, p_n$. Por lo tanto, siempre podemos construir un número primo que sea mayor que cualquier otro, y en consecuencia el conjunto de todos ellos tiene que ser infinito.

Determinar si un número dado es o no primo, es un problema complejo desde el punto de vista computacional. Actualmente, gran parte de la seguridad informática está sustentada en esta complejidad. Concretamente, son importantes los métodos y resultados para calcular primos suficientemente grandes.

La *criba de Eratóstenes* es un método simple para determinar los primos por debajo de un natural dado, aunque solo es eficiente para números no muy grandes.

También son importantes las familias o sucesiones de números primos, como los *primos de Fermat* o los *primos de Mersenne*.

Ejemplo 1.2.13 Los números primos de la forma $a^n - 1$ se denominan *Primos de Mersene*. Naturalmente, no todos los números de esta forma son números primos:

$$3^2 - 1 = 8 \quad \text{no es primo.}$$

De hecho, se verifica que, si $n > 1$ y $a^n - 1$ es un número primo, entonces $a = 2$ y n es primo, tal y como demostramos a continuación.

Concretamente, vamos a utilizar la divisibilidad de polinomios para probarlo, lo cual constituye una técnica bastante habitual.

- Necesariamente, $a \geq 2$ y para probar que $a = 2$, consideramos el polinomio $Q(x) = x^n - 1$. Este polinomio es divisible por $x - 1$, ya que $x_0 = 1$ es una raíz de Q; además, no es difícil deducir que

  $$x^n - 1 = (x - 1) \sum_{i=0}^{n-1} x^i$$

  Para $x = a$, tenemos entonces que

  $$a^n - 1 = (a - 1) \sum_{i=0}^{n-1} a^i$$

  Dado que $\sum_{i=0}^{n-1} a^i = 1 + a + \cdots + a^{n-1} \geq 2$, si $a^n - 1$ es primo, necesariamente $a - 1 = 1$ y $a = 2$.

- Para probar que si $2^n - 1$ es un número primo entonces n también es primo, vamos a demostrar el *contrarrecíproco* de esta implicación, es decir: vamos a

demostrar que si $n = m \cdot k$, con $m \geq 2$, $k \geq 2$, entonces $2^n - 1$ es un número compuesto.

Utilizamos de nuevo la siguiente igualdad de polinomios

$$x^k - 1 = (x-1)\sum_{i=0}^{k-1} x^i$$

Para $x = 2^m$, tenemos entonces que

$$2^n - 1 = (2^m)^k - 1 = (2^m - 1)\sum_{i=0}^{k-1} 2^{mi},$$

lo que nos daría una factorización de $2^n - 1$; debemos tener en cuenta que, dado que $m \geq 2$, se verifica que $2^m - 1 \geq 3$ y dado que $k \geq 2$ se verifica que $\sum_{i=0}^{k-1} 2^{mi} \geq 2$.

Es decir, los números primos de Mersenne son de la forma $2^p - 1$, en donde p es primo. Sin embargo, no es cierto que si p es primo, entonces $2^p - 1$ sea necesariamente primo. Por ejemplo:

$$2^{11} - 1 = 2047 = 23 \cdot 89 \qquad \square$$

1.2.4. Teorema fundamental de la aritmética

TEOREMA 1.2.14 (FUNDAMENTAL DE LA ARITMÉTICA) *Sea $n \in \mathbb{N}$, $n > 1$. Entonces, n se puede expresar de forma única como producto de potencias de primos. Es decir,*

$$n = p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k},$$

en donde $p_1 < p_2 < \ldots < p_k$ son primos y $e_j \in \mathbb{N}$ para cada j.

Para obtener la descomposición, empezamos dividiendo n por primos sucesivos y crecientes empezando por 2. Naturalmente, no es necesario probar con todos los primos menores que n y disponemos de varios resultados que determinan el mayor número primo que debe ser considerado. El criterio más simple es el siguiente: probaremos con los primos menores que la raíz cuadrada del número.

EJEMPLO 1.2.15

- Para deducir que 101 es primo, basta comprobar que no es divisible por $2, 3, 5$ y 7.
- El número 90 es compuesto y $90 = 2 \cdot 3^2 \cdot 5$.

El operador de `Maxima` con el que determinamos la factorización de números naturales es `factor`:

```
(%i1) factor(11016);
```

$$2^3\,3^4\,17$$

Y también podemos preguntarle sobre la primalidad de un número:

```
(%i2) primep(17);
```

true

```
(%i3) primep(22);
```

false

DEFINICIÓN 1.2.16 *Sean $n, m \in \mathbb{Z}^*$. El* máximo común divisor *de n y m es el mayor entero que es divisor de n y de m y lo denotamos por* $\mathrm{mcd}(n, m)$. *Es decir, $d = \mathrm{mcd}(n, m)$ es el único entero positivo tal que*

- *$d|a$, $d|b$ y*
- *si $c|a$, $c|b$, entonces $c|d$*

El operador de `divisors` de `Maxima` determina el conjunto de divisores de un número. Lo vamos a utilizar en el siguiente ejemplo.

EJEMPLO CON MAXIMA 1.2.17 Vamos a calcular el máximo común divisor de 30 y 12 usando la definición

```
(%i1) D1: divisors(30);
```

$$\{1, 2, 3, 5, 6, 10, 15, 30\}$$

```
(%i2) D2: divisors(12);
```

$$\{1, 2, 3, 4, 6, 12\}$$

Con el operador `intersect` podemos determinar la intersección de los dos conjuntos, es decir, los divisores comunes.

```
(%i3) intersect(D1,D2);
```

$$\{1, 2, 3, 6\}$$

Por lo tanto, el máximo de todos los divisores es 6: $\mathrm{mcd}(30, 12) = 6$.

Disponemos igualmente de un operador que nos devuelve el máximo común divisor de dos números.

```
(%i4) gcd(30,12);
```

6 □

DEFINICIÓN 1.2.18 *Se dice que los enteros n, y m, son* primos relativos *o* coprimos *si no tienen divisores comunes, es decir, si* $\mathrm{mcd}(n,m)=1$.

EJEMPLO 1.2.19

- Los enteros 9 y 22 son coprimos, ya que $\mathrm{mcd}(9,22)=1$.
- Los enteros 9, 22 y 35 son coprimos "dos a dos" ya que

$$\mathrm{mcd}(9,22)=\mathrm{mcd}(9,35)=\mathrm{mcd}(22,35)=1$$ □

A partir del Teorema Fundamental de la Aritmética obtenemos el método de cálculo del máximo común divisor de dos números basado en la descomposición factorial de los números naturales.

COROLARIO 1.2.20 *Si* $n=p_1^{a_1}p_2^{a_2}\cdots p_n^{a_n}$, *y* $m=p_1^{b_1}p_2^{b_2}\cdots p_n^{b_n}$ *son números naturales, en donde los números* p_i *son números primos distintos, entonces*

$$\mathrm{mcd}(n,m)=p_1^{\mathrm{mín}(a_1,b_1)}p_2^{\mathrm{mín}(a_2,b_2)}\cdots p_n^{\mathrm{mín}(a_n,b_n)}$$

Obsérvese que, por simplicidad, hemos escrito la misma secuencia de primos en los dos números, lo que no resta generalidad, puesto que los exponentes pueden ser nulos.

EJEMPLO 1.2.21

$$\left.\begin{array}{rcl} 2750 & = & 2\cdot 5^3\cdot 11 \\ 1992 & = & 2^3\cdot 3\cdot 83 \end{array}\right\} \Longrightarrow \left\{\begin{array}{rcl} 2750 & = & 2^1\cdot 3^0\cdot 5^3\cdot 11^1\cdot 83^0 \\ 1992 & = & 2^3\cdot 3^1\cdot 5^0\cdot 11^0\cdot 83^1 \end{array}\right.$$

Por lo tanto,

$$\mathrm{mcd}(2750,1992)=2^1\cdot 3^0\cdot 5^0\cdot 11^0\cdot 83^0=2$$ □

DEFINICIÓN 1.2.22 *Sean* $n,m\in\mathbb{Z}^+$. *El* mínimo común múltiplo *de n y m es el menor natural que es múltiplo de n y de m. Lo denotamos por* $\mathrm{mcm}(n,m)$.

El operador `lcm` de `Maxima` devuelve el mínimo común múltiplo de dos números.

```
(%i1) lcm(60,46);
```

1380

COROLARIO 1.2.23 *Si* $n=p_1^{a_1}p_2^{a_2}\cdots p_n^{a_n}$, *y* $m=p_1^{b_1}p_2^{b_2}\cdots p_n^{b_n}$ *son números naturales, entonces*

$$\mathrm{mcm}(n,m)=p_1^{\mathrm{máx}(a_1,b_1)}p_2^{\mathrm{máx}(a_2,b_2)}\cdots p_n^{\mathrm{máx}(a_n,b_n)}$$

COROLARIO 1.2.24 *Si* $n,m\in\mathbb{Z}^+$, *entonces,* $\mathrm{mcm}(n,m)\cdot\mathrm{mcd}(n,m)=n\cdot m$.

COROLARIO 1.2.25 *Si* $n,m\in\mathbb{Z}^+$ *son coprimos, entonces* $\mathrm{mcm}(n,m)=n\cdot m$.

1.3. Ecuaciones diofánticas

1.3.1. Algoritmo de Euclides

Aunque el cálculo de los divisores comunes o la factorización de dos números, como hemos hecho en la sección anterior, permite determinar fácilmente el máximo común divisor de dos números pequeños, no son métodos eficientes cuando trabajamos con números grandes. En esta sección vamos a aprender a el algoritmo de Euclides para determinar el máximo común divisor de dos números. Este algoritmo tiene dos ventajas: es mucho más eficiente que la descomposición en factores primos y se puede completar para obtener un método eficiente para determinar la *identidad de Bézout*, que estudiaremos más adelante y que constituye la herramienta fundamental para la resolución de las *ecuaciones diofánticas* que vamos a estudiar en esta parte del tema.

El algoritmo de Euclides se basa en la siguiente propiedad, consecuencia de las propiedades algebraicas de la divisibilidad: un número d es divisor común de otros dos números n y m si y solo si es divisor de $n-m$ y de m. Por lo tanto:

LEMA 1.3.1 *Sean n y m dos números naturales tales que $n > m$, entonces*

$$\text{mcd}(n, m) = \text{mcd}(n - m, m)$$

Podemos utilizar este resultado para calcular el máximo común divisor de dos números realizando simplemente restas.

EJEMPLO 1.3.2 Calculemos el máximo común divisor de 33 y 21 utilizando tantas veces como sea necesario el lema anterior hasta llegar a un cálculo trivial:

$$\begin{aligned}\text{mcd}(33, 21) &= \text{mcd}(12, 21) = \text{mcd}(21, 12) = \text{mcd}(9, 12) = \\ &= \text{mcd}(12, 9) = \text{mcd}(3, 9) = \text{mcd}(9, 3) = \text{mcd}(6, 3) = \text{mcd}(3, 3) = 3\end{aligned}$$ □

EJEMPLO CON MAXIMA 1.3.3 Con el lema anterior podemos dar la definición recursiva más simple de mcd:

```
(%i1) mcd(m,n):= if m = n then m
                 else if m > n then mcd(m - n,n)
                 else mcd(m,n - m)$
(%i2) mcd(33,21);
```

```
3
```

Recordemos que ya habíamos visto que `Maxima` incluye el operador `gcd`. □

El método descrito en este ejemplo constituye un algoritmo eficiente para el cálculo del máximo común divisor y es la base para la demostración del siguiente resultado, que describe el algoritmo de Euclides para calcular el máximo común divisor de dos números y que será la herramienta fundamental para el resto del tema.

TEOREMA 1.3.4 (ALGORITMO DE EUCLIDES) *Sean $n, m \in \mathbb{Z}^+$ y consideremos las secuencias de enteros positivos, $q_1, \ldots, q_{k+1}$ y $r_1, \ldots, r_k$, obtenidas aplicando repetidamente el algoritmo de la división:*

$$\begin{aligned} n - m \cdot q_1 &= r_1 \\ m - r_1 \cdot q_2 &= r_2 \\ r_1 - r_2 \cdot q_3 &= r_3 \\ &\cdots\cdots \\ r_{k-3} - r_{k-2} \cdot q_{k-1} &= r_{k-1} = \operatorname{mcd}(n, m) \\ r_{k-2} - r_{k-1} \cdot q_k &= r_k = 0 \end{aligned}$$

$$\begin{array}{rcll} n & = & m \cdot q_1 + r_1 & 0 < r_1 < m \\ m & = & r_1 \cdot q_2 + r_2 & 0 < r_2 < r_1 \\ r_1 & = & r_2 \cdot q_3 + r_3 & 0 < r_3 < r_2 \\ r_2 & = & r_3 \cdot q_4 + r_4 & 0 < r_4 < r_3 \\ & \cdots & & \cdots \\ r_{k-2} & = & r_{k-1} \cdot q_k + r_k & 0 < r_k < r_{k-1} \\ r_{k-1} & = & r_k \cdot q_{k+1} + 0 & \end{array}$$

Entonces $r_k = \operatorname{mcd}(n, m)$

EJEMPLO 1.3.5 Vamos a hallar el máximo común divisor de 70 y 42 usando el teorema anterior.

$$\begin{array}{rcl} 70 & = & 42 \cdot 1 + 28 \\ 42 & = & 28 \cdot 1 + 14 \\ 28 & = & 14 \cdot 2 + 0 \end{array}$$

Por lo tanto, $\operatorname{mcd}(70, 42) = 14$. □

Ejemplo 1.3.6 Los números 21 y 13 son coprimos:

$$\begin{aligned} 21 &= 13 \cdot 1 + 8 \\ 13 &= 8 \cdot 1 + 5 \\ 8 &= 5 \cdot 1 + 3 \\ 5 &= 3 \cdot 1 + 2 \\ 3 &= 2 \cdot 1 + \boxed{1} \\ 2 &= 1 \cdot 2 + 0 \end{aligned}$$

□

Teorema 1.3.7 (Identidad de Bézout) *Si $n, m \in \mathbb{Z}^+$, entonces existen $s, t \in \mathbb{Z}$ tales que* $\mathrm{mcd}(n, m) = n \cdot s + m \cdot t$.

La demostración se basa en la secuencia de divisiones que permiten calcular el máximo común divisor y además, la demostración establece el método para calcular los números s, t.

Despejando los restos en la secuencia de divisiones, obtenemos las siguientes igualdades:

$$\begin{aligned} r_1 &= n - m \cdot q_1 \\ r_2 &= m - r_1 \cdot q_2 \\ r_3 &= r_1 - r_2 \cdot q_3 \\ &\dots \\ r_{k-1} &= r_{k-3} - r_{k-2} \cdot q_{k-1} \\ \mathrm{mcd}(n, m) = r_k &= r_{k-2} - r_{k-1} \cdot q_k \end{aligned}$$

De esta forma, si de abajo a arriba, sustituimos los números r_i hasta llegar a la primera igualdad, habremos construido la combinación lineal buscada.

Vamos a ver este proceso en el ejemplo siguiente. La única dificultad del proceso está en que trabajamos con números, y es difícil distinguir el papel que juegan en cada igualdad (resto, dividendo o divisor). Por esa razón, vamos a "recuadrar" los números iniciales y los restos para recordar que no debemos operarlos, solamente los podemos sustituir por la expresión dada por la división anterior.

Ejemplo 1.3.8 Calculamos el máximo común divisor de 136 y 26 mediante la secuencia de divisiones.

$$\boxed{136} = \boxed{26} \cdot 5 + \boxed{6} \qquad \Longrightarrow \qquad \boxed{6} = \boxed{136} - \boxed{26} \cdot 5 \tag{1}$$

$$\boxed{26} = \boxed{6} \cdot 4 + \boxed{2} \qquad \Longrightarrow \qquad \boxed{2} = \boxed{26} - \boxed{6} \cdot 4 \tag{2}$$

$$6 = 2 \cdot 3 + 0$$

Por lo tanto, $\mathrm{mcd}(136, 26) = 2$ y $\boxed{2} = \boxed{26} - \boxed{6} \cdot 4$. Como decíamos antes, los números recuadrados los sustituiremos según las igualdades previas en la secuencia,

pero no los operaremos.

$$\begin{aligned}
\boxed{2} &= \boxed{26} - \boxed{6} \cdot 4, && \text{Por (2)}\\
&= \boxed{26} - (\boxed{136} - \boxed{26} \cdot 5) \cdot 4, && \text{Por (1)}\\
&= \boxed{26} - \boxed{136} \cdot 4 + \boxed{26} \cdot 20\\
&= \boxed{136} \cdot (-4) + \boxed{26} \cdot 21
\end{aligned}$$ □

En general, no es cierto que si $n \cdot s + m \cdot t = d$, entonces $d = \text{mcd}(n, m)$. Por ejemplo, $2 \cdot 2 + 3 \cdot 2 = 10$ y $\text{mcd}(2, 3) \neq 10 \neq \text{mcd}(2, 2)$. Sin embargo, la conclusión sí es cierta si los números son coprimos.

Corolario 1.3.9 *Si existen enteros s y t tales que $n \cdot s + m \cdot t = 1$, entonces $\text{mcd}(n, m) = 1$, es decir, n y m son coprimos.*

La demostración es una consecuencia inmediata de las propiedades básicas de la divisibilidad: si $n \cdot s + m \cdot t = 1$, entonces cualquier divisor común a m y n sería divisor de 1 y por lo tanto, 1 es el único divisor común y $\text{mcd}(n, m) = 1$.

Esta propiedad es útil para demostrar propiedades relacionadas con la divisibilidad.

Ejemplo 1.3.10 Vamos a utilizar la identidad de Bézout para demostrar que, para todo número natural n, los números $n+1$ y n son coprimos: dado que $(n+1)-n = 1$ el corolario anterior establece que $n + 1$ y n son coprimos. □

Forma vectorial del algoritmo de Euclides. Vamos a obtener la identidad de Bézout del ejemplo de arriba usando un método alternativo y que nos va a permitir definir de forma recursiva el proceso. Concretamente, vamos a partir de las siguientes igualdades triviales,

$$\begin{aligned}
m &= 1 \cdot m + 0 \cdot n\\
n &= 0 \cdot m + 1 \cdot n
\end{aligned}$$

y a partir de ellas, vamos a realizar operaciones elementales en el lado izquierdo y en el derecho hasta conseguir el máximo común divisor en el lado izquierdo y la combinación de la identidad de Bézout en el lado derecho.

Ejemplo 1.3.11 Vamos a calcular el máximo común divisor de 250 y 111 y a obtener la combinación lineal establecida por la identidad de Bézout.

Empezamos con las dos igualdades triviales siguientes:

$$\begin{aligned} 250 &= 1 \cdot 250 + 0 \cdot 111 \qquad (1) \\ 111 &= 0 \cdot 250 + 1 \cdot 111 \qquad (2) \end{aligned}$$

El cociente de 250 entre 111 es 2 y $250 - 2 \cdot 111 = 28$; efectuando esta operación con las partes derechas e izquierdas de las igualdades anteriores, obtenemos:

$$28 = 250 - 2 \cdot 111 = 1 \cdot 250 - 2 \cdot 111 \qquad (3) = (1) - 2 \cdot (2)$$

Ahora dividimos 111 entre 28 y obtenemos el cociente 3, por lo que:

$$27 = 111 - 3 \cdot 28 = -3 \cdot 250 + 7 \cdot 111 \qquad (4) = (2) - 3 \cdot (3)$$

Finalmente, dividididimos 28 entre 27 y obtenemos cociente y resto iguales a 1:

$$1 = 28 - 27 = 4 \cdot 250 - 9 \cdot 111 \qquad (5) = (3) - (4)$$

Es decir, los números en el lado derecho determinan las operaciones que hacemos, pero las realizamos en ambos lados. Como podemos ver, los números 250 y 111 del lado derecho no se han operado en ningún momento y solo hemos trabajado con sus "coeficientes". Por esta razón, el proceso anterior se puede simplificar describiéndolo sobre vectores de tres coeficientes y por eso, a esta forma de determinar la identidad de Bézout la llamaremos *forma vectorial*.

$$\begin{aligned} \left\lfloor \frac{250}{111} \right\rfloor = 2 \quad &\rightsquigarrow \quad \begin{pmatrix} 250 \\ 1 \\ 0 \end{pmatrix} - 2 \cdot \begin{pmatrix} 111 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 28 \\ 1 \\ -2 \end{pmatrix} \\ \left\lfloor \frac{111}{28} \right\rfloor = 3 \quad &\rightsquigarrow \quad \begin{pmatrix} 111 \\ 0 \\ 1 \end{pmatrix} - 3 \cdot \begin{pmatrix} 28 \\ 1 \\ -2 \end{pmatrix} = \begin{pmatrix} 27 \\ -3 \\ 7 \end{pmatrix} \\ \left\lfloor \frac{28}{27} \right\rfloor = 1 \quad &\rightsquigarrow \quad \begin{pmatrix} 28 \\ 1 \\ -2 \end{pmatrix} - \begin{pmatrix} 27 \\ -3 \\ 7 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \\ -9 \end{pmatrix} \end{aligned}$$

□

La forma vectorial que hemos presentado en el ejemplo anterior, permite adaptar fácilmente el procedimiento que definimos para hallar el máximo común divisor con `Maxima` basado en las diferencias sucesivas.

```
(%i1) mcdex(n,m):= bezout_vect([n,1,0],[m,0,1])$
(%i2) bezout_vect(u,v):= if v[1] > u[1] then bezout_vect(v,u)
                         else if v[1]=0 then u
                         else bezout_vect(u-v,v)$
(%i3) mcdex(250,111);
```

$$[1, 4, -9]$$

También disponemos de varios operadores relacionados con los procesos anteriores. Por ejemplo, el operador `gcdex` devuelve los coeficientes de la identidad de Bézout y el máximo máximo común divisor, que aparece en el tercer argumento de la salida:

```
(%i3) gcdex(250,111);
```

$$[4, -9, 1]$$

Para ayudarnos a verificar los pasos intermedios, podemos usar el operador `cf`, que nos devuelve la secuencia de cocientes en el algoritmo de Euclides si lo aplicamos a la fracción determinada por los dos números:

```
(%i4) cf(250/111);
```

$$[2, 3, 1, 27]$$

1.3.2. Ecuaciones Diofánticas

Hablamos de *ecuaciones diofánticas* si estamos interesados en resolverlas dentro de $\mathbb{Z}$. Deben su nombre al matemático griego Diophantus, que escribió extensamente acerca de este tipo de ecuaciones. En este curso, solo vamos a estudiar las ecuaciones diofánticas lineales, cuya solución se puede determinar con el algoritmo de Euclides y la identidad de Bézout.

Definición 1.3.12 *Una* ecuación diofántica lineal de dos variables *es una ecuación de la forma*

$$n \cdot x + m \cdot y = k,$$

en donde $n, m, k \in \mathbb{Z}$*, y las incógnitas* x *e* y *deben resolverse en* $\mathbb{Z}$*.*

Teorema 1.3.13 *La ecuación* $n \cdot x + m \cdot y = k$ *tiene soluciones en* $\mathbb{Z}$ *si y solo si* $d = \text{mcd}(n, m)$ *divide a* k*. En tal caso, si el par* (x_0, y_0) *es una solución de la ecuación, entonces el resto de las soluciones vienen dadas por*

$$x = x_0 + \left(\frac{m}{d}\right)q, \quad y = y_0 - \left(\frac{n}{d}\right)q, \quad q \in \mathbb{Z}$$

Demostración del teorema:

- ($\Rightarrow$) Si $d = \text{mcd}(n, m)$, entonces d divide a n y a m y en consecuencia divide a $n \cdot x + m \cdot y = k$ para todo $x, y \in \mathbb{Z}$.
- ($\Leftarrow$) Si $d = \text{mcd}(n, m)$ divide a k, entonces $k = d \cdot c$ para algún $c \in \mathbb{Z}$. Además, por la identidad de Bézout, existen enteros s y t tales que

  $$d = n \cdot s + m \cdot t$$

  y en consecuencia, $k = c \cdot d = n \cdot c \cdot s + m \cdot c \cdot t$; es decir $x_0 = c \cdot s$ e $y_0 = c \cdot t$ forman una solución de la ecuación.
- Supongamos que (x_0, y_0) es una solución de la ecuación y sea

  $$x = x_0 + \frac{m}{d}q, \qquad y = y_0 - \frac{n}{d}q$$

  para algún $q \in \mathbb{Z}$; vamos a comprobar que también son solución de la ecuación

  $$\begin{aligned} n \cdot x + m \cdot y &= n \cdot \left(x_0 + \frac{m}{d}q\right) + m \cdot \left(y_0 - \frac{n}{d}q\right) = \\ &= n \cdot x_0 + \frac{n \cdot m \cdot q}{d} + m \cdot y_0 - \frac{m \cdot n \cdot q}{d} = n \cdot x_0 + m \cdot y_0 = k \end{aligned}$$

- Finalmente, tenemos que demostrar que no hay más soluciones que las descritas en el punto anterior. Supongamos que (x, y) es otra solución de la ecuación, además de (x_0, y_0):

  $$n \cdot x + m \cdot y = k, \qquad n \cdot x_0 + m \cdot y_0 = k$$

  Restamos la igualdades anteriores, miembro a miembro, y operamos como se muestra a continuación:

  $$\begin{aligned} &n(x - x_0) + m(y - y_0) = 0 \\ &\frac{n}{d}(x - x_0) + \frac{m}{d}(y - y_0) = 0 \\ &\frac{n}{d}(x - x_0) = \frac{m}{d}(y_0 - y) \end{aligned}$$

  Dado que $\frac{n}{d}$ y $\frac{m}{d}$ son coprimos, para que se verifique la igualdad anterior, $\frac{n}{d}$ debe dividir necesariamente a $(y_0 - y)$ y $\frac{m}{d}$ debe dividir a $(x - x_0)$. Por lo tanto, $\frac{x - x_0}{m/d} = \frac{y_0 - y}{n/d} = q \in \mathbb{Z}$. □

COROLARIO 1.3.14 *Si* $\text{mcdex}(n, m) = \begin{pmatrix} d \\ s \\ t \end{pmatrix}$ *y* d *divide a* c*, entonces la solución de la ecuación* $n \cdot x + m \cdot y = c$ *es:*

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{d}\left[c\begin{pmatrix} s \\ t \end{pmatrix} + q\begin{pmatrix} -m \\ n \end{pmatrix}\right], \quad \textit{para cada } q \in \mathbb{Z},$$

En la demostración anterior se describe el procedimiento de resolución de una ecuación diofántica, tal y como reproducimos en el siguiente ejemplo.

EJEMPLO 1.3.15 Vamos a estudiar la ecuación diofántica $21x + 14y = 70$. Calculamos en primer lugar mcdex$(21, 14)$:

$$\begin{pmatrix} 21 \\ 1 \\ 0 \end{pmatrix} - \left\lfloor \frac{21}{14} \right\rfloor \cdot \begin{pmatrix} 14 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 21 \\ 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 14 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 7 \\ 1 \\ -1 \end{pmatrix}$$

Dado que $7|14$, mcdex$(21, 14) = \begin{pmatrix} 7 \\ 1 \\ -1 \end{pmatrix}$ y

$$7 = 21 \cdot 1 + 14 \cdot (-1)$$

Multiplicando por 10 a ambos lados de la igualdad, encontramos una solución de la ecuación propuesta:

$$\begin{aligned} 7 \cdot 10 &= (21 \cdot 1 + 14 \cdot (-1))10 \\ 70 &= 21 \cdot 10 + 14 \cdot (-10) \end{aligned}$$

Es decir, $x_0 = 10$, $y_0 = -10$ es una solución particular de la ecuación.

Supongamos que (x, y) es otra solución.

$$\begin{aligned} 21 \cdot x + 14 \cdot y &= 70 \\ 21 \cdot 10 + 14 \cdot (-10) &= 70 \\ 21(x - 10) + 14(y + 10) &= 0 \qquad \text{(Hemos restado las dos anteriores)} \\ 3(x - 10) + 2(y + 10) &= 0 \qquad \text{(Hemos dividido por el m.c.d.)} \\ 3(x - 10) &= -2(y + 10) \\ \frac{x - 10}{-2} &= \frac{y + 10}{3} = q \in \mathbb{Z} \end{aligned}$$

Despejamos x e y en función de q para obtener la solución general.

$$x = 10 - 2 \cdot q, \qquad y = -10 + 3 \cdot q, \qquad q \in \mathbb{Z}$$

También podemos escribir la solución utilizando el corolario visto anteriormente:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{7}\left[70\begin{pmatrix} 1 \\ -1 \end{pmatrix} + q\begin{pmatrix} -14 \\ 21 \end{pmatrix}\right] = \begin{pmatrix} 10 - 2q \\ -10 + 3q \end{pmatrix}, \quad \text{para cada } q \in \mathbb{Z}, \qquad \square$$

Forma matricial del algoritmo de Euclides. La igualdad $r = n - m \cdot q$ obtenida con la división euclídea justifica la siguiente igualdad entre matrices

$$\begin{pmatrix} m \\ r \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix} \begin{pmatrix} n \\ m \end{pmatrix}$$

Por lo tanto, si $d = \text{mcd}(n, m)$ y $q_1, \ldots, q_k$, es la secuencia de los cocientes obtenidos en el algoritmo de Euclides para calcular el máximo común divisor, según aparece en el teorema del Algoritmo de Euclides, entonces

$$\begin{pmatrix} d \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_k \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -q_2 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -q_1 \end{pmatrix} \begin{pmatrix} n \\ m \end{pmatrix} = \begin{pmatrix} s & t \\ \pm m/d & \mp n/d \end{pmatrix} \begin{pmatrix} n \\ m \end{pmatrix}$$

Es decir, la primer fila de la matriz cuadrada de la derecha nos da los coeficientes cuya existencia establece el teorema de la identidad de Bézout.

Vamos a extender esta igualdad para expresar las soluciones de la ecuación diofántica $nx + my = c$, en donde $d = \text{mcd}(n, m)|c$. Basta con multiplicar ambos lados de la igualdad por la matriz $(\frac{c}{d} \;\; q)$ en donde $q \in \mathbb{Z}$ es un parámetro:

$$\begin{pmatrix} \frac{c}{d} & q \end{pmatrix} \begin{pmatrix} d \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{c}{d} & q \end{pmatrix} \begin{pmatrix} s & t \\ \pm m/d & \mp n/d \end{pmatrix} \begin{pmatrix} n \\ m \end{pmatrix}$$

$$c = \begin{pmatrix} \frac{c}{d}s \pm \frac{m}{d}q & \frac{c}{d}t \mp \frac{n}{d}q \end{pmatrix} \begin{pmatrix} n \\ m \end{pmatrix}$$

Es decir, el producto de matrices en el lado derecho determina la solución general de la ecuación diofántica. Este desarrollo, justifica el siguiente resultado

Teorema 1.3.16 *Consideremos la ecuación diofántica $nx + my = c$ tal que c es múltiplo de $d = \text{mcd}(n, m)$. Sea $q_1, \ldots, q_k$ la secuencia de cocientes obtenidos aplicando el algoritmo de Euclides a los números n y m hasta llegar al resto 0. Entonces, las soluciones de la ecuación vienen dadas por la siguiente igualdad*

$$(x \quad y) = \begin{pmatrix} \frac{c}{d} & q \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -q_k \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -q_2 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -q_1 \end{pmatrix}, \quad q \in \mathbb{Z}$$

Debemos recordar que el producto de matrices no es conmutativo y, por lo tanto, prestar especial cuidado al escribir el producto de las matrices según la secuencia de divisiones.

Ejemplo con Maxima 1.3.17 Vamos a estudiar si la siguiente ecuación tiene solución y en tal caso vamos a determinar su solución general

$$145x + 65y = 70$$

Realizamos las divisiones sucesivas del algoritmo de Euclides para determinar el máximo común divisor de 145 y 65.

$$145 = 65 \cdot \mathbf{2} + 15$$
$$65 = 15 \cdot \mathbf{4} + 5$$
$$15 = 5 \cdot \mathbf{3} + 0$$

La ecuación tiene solución, puesto que $\text{mcd}(145, 65) = 5|70$. La secuencia de cocientes que hemos obtenido es $q_1 = 2$, $q_2 = 4$, $q_3 = 3$ y por lo tanto, las soluciones de la ecuación son:

$$(x \quad y) = \left(\frac{70}{5} \quad q\right) \begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -4 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix}$$

Vamos a completar los cálculos con ayuda de `Maxima`. En primer lugar, vamos a comprobar que hemos obtenido correctamente la secuencia de cocientes en el algoritmo de Euclides y que hemos calculado correctamente el máximo común divisor.

```
(%i1) cf(145/65);
```

$$[2, 4, 3]$$

```
(%i2) gcd(145,65);
```

$$5$$

```
(%i3) 70/5;
```

$$14$$

Y finalmente obtenemos la solución general usando el producto de matrices:

```
(%i4) [x,y] = matrix([70/5,q]).
              matrix([0,1],[1,-3]).
              matrix([0,1],[1,-4]).
              matrix([0,1],[1,-2]) ;
```

$$(x, y) = (13q - 56, 126 - 29q)$$ □

Ejemplo 1.3.18 Queremos retirar 510 euros de un cajero que solo dispensa billetes de 20 y 50 euros, ¿es posible obtener esa cantidad? y en tal caso ¿de cuántas formas distintas puede hacerlo el cajero?

Si x es el número de billetes de 20 dispensados por el cajero e y es el número de billetes de 50, lo que necesitamos saber si la siguiente ecuación diofántica tiene solución

$$20x + 50y = 510$$

Además, en este caso, queremos saber soluciones son positivas y cuáles son, ya que el cajero no recoge billetes, solo los dispensa. Dado que $\mathrm{mcd}(50, 20) = 10$, podemos afirmar que el cajero puede darnos cualquier cantidad de euros múltiplo de 10 y mayor que 50, y en particular los 510 euros que le pedimos en este ejemplo. A partir de la identidad de Bézout del par 50, 20, es decir, $10 = 50 - 20 \cdot 2$, deducimos que $510 = 50 \cdot 51 + 20 \cdot (-102)$, es decir, $x_0 = -102$, $y_0 = 51$ es una solución particular. El resto de las soluciones son

$$x = -102 + 5 \cdot q, \qquad y = 51 - 2 \cdot q, \qquad \text{para cada } q \in \mathbb{Z}.$$

Si $x = -102 + 5 \cdot q \geq 0$, entonces $q \geq \frac{102}{5}$ y si $y = 51 - 2 \cdot q \geq 0$, entonces $q \leq \frac{51}{2}$. Por lo tanto, los posibles valores de q para obtener soluciones positivas son: 21, 22, 23, 24 y 25 y las correspondientes soluciones son:

$$\begin{aligned}
&x = -102 + 5 \cdot 21 = 3; \quad y = 51 - 2 \cdot 21 = 9; \quad \Rightarrow \quad 20 \cdot 3 + 50 \cdot 9 = 510\\
&x = -102 + 5 \cdot 22 = 8; \quad y = 51 - 2 \cdot 22 = 7; \quad \Rightarrow \quad 20 \cdot 8 + 50 \cdot 7 = 510\\
&x = -102 + 5 \cdot 23 = 13; \quad y = 51 - 2 \cdot 23 = 5; \quad \Rightarrow \quad 20 \cdot 13 + 50 \cdot 5 = 510\\
&x = -102 + 5 \cdot 24 = 18; \quad y = 51 - 2 \cdot 24 = 3; \quad \Rightarrow \quad 20 \cdot 18 + 50 \cdot 3 = 510\\
&x = -102 + 5 \cdot 25 = 23; \quad y = 51 - 2 \cdot 25 = 1; \quad \Rightarrow \quad 20 \cdot 23 + 50 \cdot 1 = 510
\end{aligned}$$

Es decir, 3 billetes de veinte y 9 de cincuenta, o bien 8 billetes de veinte y 7 de cincuenta, o bien 13 billetes de veinte y 5 de cincuenta, o bien 18 billetes de veinte y 3 de cincuenta, o bien 23 billetes de veinte y 1 de cincuenta. □

1.4. Aritmética Modular

1.4.1. Relación de congruencia

Dos números son *congruentes* respecto de un módulo $m \in \mathbb{N}$ si se diferencian en un múltiplo de m. Esta relación la encontramos en muchos contextos. Por ejemplo, cuando trabajamos con ángulos, si sumamos cualquier múltiplo de 360, obtenemos esencialmente el mismo ángulo, es decir, dos números representan el mismo ángulo si son congruentes módulo 360.

La medida del tiempo es otra fuente de ejemplos de relaciones de congruencia. El tiempo se divide en bloques de 12 o 24 horas de modo que, por ejemplo, las 20 horas son las 8. Lo mismo ocurre con los días de la semana y los meses.

En computación, la relación de congruencia se usa, entre otras cosas, para definir dígitos de control, es decir, números o caracteres que ayudan a detectar errores. Por ejemplo, la letra con la que terminan los números del DNI se obtienen por congruencia módulo 23, de tal forma, que dos números con la misma letra son congruentes módulo 23.

En matemáticas, la relación de congruencia es una herramienta fundamental para abordar determinados problemas, como el estudio de las reglas de divisibilidad.

DEFINICIÓN 1.4.1 (CONGRUENCIA MÓDULO m) *Sean $a, b, m \in \mathbb{Z}$ tales que $m \geq 2$. Se dice que a es* congruente con b módulo m *si $(a-b)$ es múltiplo de m. En tal caso, escribimos*

$$a \equiv b \pmod{m}$$

Otras notaciones alternativas que se pueden encontrar en la bibliografía son $a \equiv_m b$, o $a \equiv b(m)$.

EJEMPLO 1.4.2

- $23 \equiv 17 \pmod{3}$, ya que $23 - 17 = 6 = 2 \cdot 3$.
- $-12 \equiv 14 \pmod{13}$, ya que $-12 - 14 = -26 = (-2)13$. □

TEOREMA 1.4.3 *Sea $m \in \mathbb{N}$, $m \geq 2$, y $a, b \in \mathbb{Z}$.*

- *$a \equiv b \pmod{m}$ si y solo si a y b tienen el mismo resto al dividirlos entre m.*
- *Cada entero $a \in \mathbb{Z}$ es congruente módulo m con uno de los siguientes enteros: $0, 1, \ldots, m-1$.*

EJEMPLO 1.4.4 Dado que $231 = 5 \cdot 46 + 1$ y $106 = 5 \cdot 21 + 1$ podemos afirmar que $231 \equiv 106 \pmod{5}$. □

En operador `mod`(n, m) de `Maxima` determina el número entre 0 y $m-1$ congruente con n módulo m, incluso aunque n sea negativo.

```
(%i1) mod(231,17)
```

```
10
```

```
(%i2) (231-10)/17
```

```
13
```

```
(%i3) mod(-231,17)
```

```
7
```

```
(%i4) (-231-7)/17
```

$$-14$$

Si n es positivo, $\texttt{remainder}(n, m) = \texttt{mod}(n, m)$, pero no ocurre lo mismo si el primer argumento es negativo, por lo que no debemos confundir los dos operadores.

TEOREMA 1.4.5 *La relación de congruencia módulo m, verifica las siguientes propiedades:*

1. Reflexiva: $a \equiv a \pmod{m}$
2. Simétrica: *Si* $a \equiv b \pmod{m}$, *entonces* $b \equiv a \pmod{m}$.
3. Transitiva: *Si* $a \equiv b \pmod{m}$ *y* $b \equiv c \pmod{m}$, *entonces* $a \equiv c \pmod{m}$.

Denotaremos por $[a]_m$ al conjunto de enteros congruentes con a módulo m, y lo denominamos *clase de a módulo de m*:

$$[a]_m = \{b \in \mathbb{Z} \mid b \equiv a \pmod{m}\}$$

Por lo tanto, como consecuencia de las propiedades del teorema anterior,

$$a \equiv b \pmod{m} \iff [a]_m = [b]_m$$

1.4.2. Aritmética Modular

Según hemos visto, cada número es congruente a un número entre 0 y $m-1$, por lo que hay exactamente m clases módulo m y son disjuntas dos a dos. Denotaremos por $\mathbb{Z}_m$ al conjunto de las clases módulo m.

$$\mathbb{Z}_m = \{[0]_m, [1]_m, [2]_m, \ldots, [m-1]_m\}$$

EJEMPLO 1.4.6 Las cuatro clases de la congruencia módulo 4 son

$$\begin{aligned}
[0]_4 &= \{\ldots, -8, -4, 0, 4, 8, \ldots\} \\
[1]_4 &= \{\ldots, -7, -3, 1, 5, 9, \ldots\} \\
[2]_4 &= \{\ldots, -6, -2, 2, 6, 10, \ldots\} \\
[3]_4 &= \{\ldots, -5, -1, 3, 7, 11, \ldots\} \\
\mathbb{Z}_4 &= \left\{[0]_4, [1]_4, [2]_4, [3]_4\right\}
\end{aligned}$$

□

Las propiedades de la relación de congruencia, permiten realizar operaciones aritméticas entre clases de congruencia, es lo que llamamos aritmética modular. Podemos hacer esto gracias al siguiente resultado, que establece que si sumamos cualquier elemento de una clase con cualquier elemento de otra clase, el resultado estará siempre en la misma clase, independientemente de la elección; lo mismo ocurre con la diferencia y con el producto.

Teorema 1.4.7 (Aritmética de las congruencias) *Sean $a, b, c, d, m \in \mathbb{Z}$, tales que $m > 1$ y $a \equiv b \pmod{m}$ y $c \equiv d \pmod{m}$. Entonces:*

1. $a + c \equiv b + d \pmod{m}$
2. $a - c \equiv b - d \pmod{m}$
3. $a \cdot c \equiv b \cdot d \pmod{m}$

Demostración: La demostración es una simple comprobación.

$$\left.\begin{array}{l} a = b + m \cdot k_1 \\ c = d + m \cdot k_2 \end{array}\right\} \Longrightarrow \left\{\begin{array}{l} a + c = b + d + m(k_1 + k_2) \\ a - c = b - d + m(k_1 - k_2) \\ a \cdot c = b \cdot d + m(d \cdot k_1 + b \cdot k_2 + m \cdot k_1 \cdot k_2) \end{array}\right.$$ □

Por lo tanto, este teorema justifica que las siguientes operaciones están bien definidas dentro de $\mathbb{Z}_m$:

$$[a]_m + [b]_m = [a + b]_m, \quad [a]_m - [b]_m = [a - b]_m, \quad [a]_m[b]_m = [a \cdot b]_m$$

Cuando trabajamos con congruencias, lo habitual es querer expresar las clases de equivalencia utilizando un número positivo menor que la base de la congruencia, por lo que normalmente, usaremos la aritmética modular para simplificar los números que operamos.

Ejemplo 1.4.8 Vamos a determinar el resto de dividir 79^7 entre 11 ayudándonos de las operaciones con clases de equivalencia. Empezamos dividiendo 79 entre 11 para obtener la clase de 79 módulo 11:

$$79 = 11 \cdot 7 + 2$$

Por lo tanto, $[79]_{11} = [2]_{11}$. Para simplificar las potencias, las vamos operando por partes:

$$[79^7]_{11} = [2^7]_{11} = [2^4]_{11}[2^3]_{11} = [16]_{11}[8]_{11} = [5]_{11}[8]_{11} = [40]_{11} = [7]_{11}$$

Por lo tanto, $79^7 \equiv 7 \pmod{11}$. □

Obsérvese que, entre las operaciones con clases de congruencia, no hemos incluido la división, ya que, en general la división no conserva congruencias:

$$2 \cdot 9 \equiv 2 \cdot 3 \pmod{12}, \qquad \text{pero} \qquad 9 \not\equiv 3 \pmod{12}$$

Sin embargo, si disponemos de la propiedad de *cancelación*.

TEOREMA 1.4.9 (CANCELACIÓN) *Sean $a, b, c, m \in \mathbb{Z}$, con $m > 1$ y $\text{mcd}(c, m) = 1$. Si $a \cdot c \equiv b \cdot c \pmod{m}$, entonces $a \equiv b \pmod{m}$.*

La demostración de este resultado es bastante simple. Si $a \cdot c \equiv b \cdot c \pmod{m}$, entonces

$$(a - b)c = a \cdot c - b \cdot c = k \cdot m$$

Es decir, c divide a $k \cdot m$, y dado que c y m son coprimos, necesariamente c divide a k y por lo tanto $\frac{k}{c} \in \mathbb{Z}$. Por lo tanto,

$$a - b = \frac{k}{c} \cdot m \qquad \Longrightarrow \qquad a \equiv b \pmod{m}$$

1.4.3. Teorema de Euler-Fermat

En muchas aplicaciones de la aritmética modular será necesario el cálculo de potencias de clases modulares. El teorema de Euler-Fermat que vemos en esta sección, nos ayuda a simplificar potencias en determinados casos. Para enunciar el resultado, necesitamos introducir previamente la función de Euler.

DEFINICIÓN 1.4.10 *La* función de Euler $\Phi\colon \mathbb{Z}^+ \to \mathbb{Z}^+$ *se define como:* $\Phi(n)$ *es el número de enteros positivos menores que n y coprimos con n.*

La siguiente tabla, recoge los valores que toma la función de Euler sobre los primeros 12 números naturales.

n	1	2	3	4	5	6	7	8	9	10	11	12
$\Phi(n)$	1	1	2	2	4	2	6	4	6	4	10	4

No hay una forma eficiente de evaluar la función de Euler y de hecho la forma más simple de expresar su valor es utilizando la factorización del número.

TEOREMA 1.4.11 (PROPIEDADES DE Φ)

1. *Si p es primo, entonces $\Phi(p^e) = p^e - p^{e-1}$. En particular, p es primo, $\Phi(p) = p - 1$.*
2. *Si m y n son coprimos, entonces $\Phi(m \cdot n) = \Phi(m) \cdot \Phi(n)$.*
3. *Como consecuencia de los puntos anteriores, si $n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$ es la expresión factorizada de n, entonces*

$$\Phi(n) = (p_1^{e_1} - p_1^{e_1 - 1})(p_2^{e_2} - p_2^{e_2 - 1}) \dots (p_k^{e_k} - p_k^{e_k - 1})$$

Ejemplo 1.4.12

- $\Phi(11) = 11 - 1 = 10$
- $\Phi(12) = \Phi(2^2 \cdot 3) = (2^2 - 2^1)(3^1 - 3^0) = 4$
- $\Phi(180) = \Phi(2^2 \cdot 3^2 \cdot 5) = (2^2 - 2)(3^2 - 3)(5 - 1) = 48$

La función de Euler se cálcula en `Maxima` con el operador `totient`

```
(%i1) totient(504);
```

$$144$$

```
(%i2) factor(504);
```

$$2^3 3^2 7$$

```
(%i3) (2^3-2^2)*(3^2-3)*(7-1);
```

$$144$$

Ya podemos enunciar el Teorema de Euler-Fermat.

Teorema 1.4.13 (Euler-Fermat) *Sea $m \in \mathbb{Z}^+$, $m > 1$, y $a \in \mathbb{Z}$ tales que* $\text{mcd}(a, m) = 1$. *Entonces:*

$$a^{\Phi(m)} \equiv 1 \pmod{m}$$

En el siguiente ejemplo, simplificamos una potencia con la ayuda del teorema de Euler-Fermat.

Ejemplo 1.4.14 Para determinar el menor entero positivo x tal que

$$x \equiv 15^{39} \pmod{37}$$

empezamos usando el teorema de Euler-Fermat para encontrar una potencia de 15 congruente con 1:

$$1 \equiv 15^{\Phi(37)} = 15^{36} \pmod{37}$$

De esta forma, la simplificación queda:

$$15^{39} = 15^{36}15^3 \equiv 1 \cdot 15^3 = 225 \cdot 15 \equiv 3 \cdot 15 \equiv 8 \pmod{37}$$ □

El teorema de Euler-Fermat es parte de los cálculos realizados por el operador `power_mod` de `Maxima`.

```
(%i1) power_mod(15,39,37);
```

$$8$$

```
(%i2) mod(15^39,37);
```

$$8$$

1.4.4. Aplicación: criterios de divisibilidad

Una de las aplicaciones de la aritmética modular es la obtención de criterios de divisibilidad descritos sobre los dígitos de la expresión decimal. Observemos en primer lugar que si un número N es divisible por otro número p, entonces el resto de la división de N entre p es 0, es decir, la clase de N módulo p es la clase del 0. Por lo tanto, para estudiar si N es divisible por p, bastaría con simplificar la clase de N, utilizando las propiedades de la aritmética modular.

Para hacer esta simplificación de forma general, utilizamos la expresión de los números determinada por su expresión en base diez. Si $N = (\delta_\ell \delta_{\ell-1} \delta_{\ell-2} \dots \delta_1 \delta_0)_{(10}$, entonces

$$N = \sum_{i=0}^{\ell} \delta_i 10^i$$

Por lo tanto, N es divisible por p si y solo si

$$\left[\sum_{i=0}^{\ell} \delta_i 10^i\right]_p = [0]_p$$

Por lo tanto, para estudiar si N es divisible por p basta simplificar la clase anterior en $\mathbb{Z}_p$ utilizando las propiedades de las congruencias. Esta simplificación solo dependerá de los dígitos δ_i ya que las potencias de 10 son constantes. Esto nos permite obtener reglas de divisibilidad basadas en los dígitos.

Ejemplo 1.4.15 Vamos a deducir la regla de divisibilidad por $p = 3$, para ello simplificamos las potencias de 10 módulo 3.

$$[10]_3 = [1 + 3 \cdot 3]_3 = [1]_3$$
$$[10^i]_3 = [1^i]_3 = [1]_3$$

Por lo tanto,

$$\left[\sum_{i=0}^{\ell} \delta_i 10^i\right]_3 = \left[\sum_{i=0}^{\ell} \delta_i\right]_3$$

Es decir, un número N es divisible por 3 si y solo si la suma de sus dígitos es divisible por 3. Por ejemplo

$$[12473]_3 = [1 + 2 + 4 + 7 + 3]_3 = [1 + 2 + 1 + 1 + 0]_3 = [5]_3 = [2]_3$$

y en consecuencia 12473 no es múltiplo de 3. □

1.4.5. Congruencias Lineales

Un problema frecuente y con importantes aplicaciones al trabajar con congruencias, es la resolución de ecuaciones. Concretamente, en este curso vamos a resolver

ecuaciones del tipo

$$a \cdot x \equiv b \pmod{m}$$

en donde a y b son enteros cualesquiera y $m > 1$; en esta ecuación, buscamos los números x que hacen válida la relación de congruencia. Este tipo de ecuaciones se denomina *congruencia lineal*.

Realmente, ya disponemos de la herramienta fundamental para resolver las congruencias lineales, ya que son básicamente ecuaciones diofánticas: $a \cdot x \equiv b \pmod{m}$ si y solo si existe y tal que $a \cdot x + m \cdot y = b$.

Ejemplo 1.4.16 Vamos a resolver la congruencia $9x \equiv 12 \pmod{15}$, y para ello resolvemos la ecuación diofántica $9x + 15y = 12$.

$$\begin{aligned} 15 &= 9 \cdot \mathbf{1} + 6 \\ 9 &= 6 \cdot \mathbf{1} + 3 \\ 6 &= 3 \cdot \mathbf{2} + 0 \end{aligned}$$

Por lo tanto, $3 = \mathrm{mcd}(15, 9)$ y

$$(y \quad x) = \left(\frac{12}{3} \quad q\right) \begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 3q - 4 & 8 - 5q \end{pmatrix}$$

La variable y ha sido una variable auxiliar, solo nos interesan los valores de x:

$$x = 8 - 5q$$

Estamos resolviendo una congruencia lineal módulo 15 y podemos observar que el conjunto soluciones que hemos obtenido, está formado por tres clases de equivalencia módulo 15:

$$[x_1]_{15} = [8]_{15}, \quad [x_2]_{15} = [8 + 5]_{15} = [13]_{15}, \quad [x_3]_{15} = [13 + 5]_{15} = [18]_{15} = [3]_{15}$$

□

Por lo tanto, la teoría de las ecuaciones diofánticas que hemos estudiado anteriormente permite deducir el siguiente resultado.

Teorema 1.4.17 (Brahmagupta) *Sean $a, b, m \in \mathbb{Z}$, $m > 1$ y consideremos la congruencia lineal*

$$a \cdot x \equiv b \pmod{m} \tag{1.1}$$

1. *La congruencia tiene solución si y solo si* $\mathrm{mcd}(a, m)$ *divide a b.*
2. *En tal caso, el conjunto de soluciones viene dado por la unión de exáctamente* $d = \mathrm{mcd}(a, m)$ *clases de equivalencia módulo m.*

Demostración:

1. El primer apartado es consecuencia de la teoría asociada a ecuaciones diofánticas, ya que la resolución de la congruencia $a \cdot x \equiv b \pmod{m}$ es equivalente a la resolución de la ecuación $a \cdot x + m \cdot y = b$.

2. Si la congruencia lineal tiene solución, esta se determina resolviendo la ecuación diofántica $a \cdot x + m \cdot y = b$. Por lo tanto, la solución general tiene la forma

$$x_0 + \frac{m}{d}k, \quad \text{para todo } k \in \mathbb{Z},$$

en donde x_0 es una solución particular.

Por lo tanto, si $t \in \{0, \ldots, d-1\}$, las soluciones asociadas a cada $k \in [t]_d$, están en la misma clase de equivalencia módulo m:

$$\begin{aligned} x_0 + \frac{m}{d}(t + d \cdot s) &= x_0 + \frac{m}{d}t + m \cdot s \\ &\equiv x_0 + \frac{m}{d}t \pmod{m} \end{aligned}$$

En consecuencia, el conjunto de todas las soluciones está formado por la unión de las d clases de equivalencia módulo m obtenidas con $k \in \{0, \ldots, d-1\}$. □

Ejemplo 1.4.18

- Las congruencia $2x \equiv 1 \pmod{4}$ no tiene solución, ya que $2 = \text{mcd}(2, 4)$ no es divisor de 1.
- $10x \equiv 8 \pmod{15}$ no tiene solución, ya que $5 = \text{mcd}(10, 15)$ no es divisor de 8.
- El segundo apartado del teorema de Brahmagupta se expresa más brevemente diciendo que la congruencia lineal tiene *d soluciones módulo m* o *d soluciones incongruentes*. Por ejemplo, la congruencia $2x \equiv 1 \pmod{5}$ tiene solo una solución módulo 5, ya que $\text{mcd}(2, 5) = 1$; la congruencia $2x \equiv 0 \pmod{4}$ tiene 2 soluciones módulo 4; la congruencia $24x \equiv 8 \pmod{28}$ tiene 4 soluciones módulo 28. □

1.4.5.1. Inversos modulares

Ya hemos visto anteriormente que no es posible trasladar la división de enteros a una división de clases de congruencia. En términos de congruencias, para calcular el *inverso* (respecto del producto) de una clase $[a]_m$, necesitamos encontrar una clase $[x]_m$ tal que $[a \cdot x]_m = [1]_m$; es decir, necesitamos resolver la siguiente congruencia:

$$a \cdot x \equiv 1 \pmod{m}$$

Atendiendo al teorema de Brahmagupta, podemos afirmar que $[a]_m$ tiene inverso en $\mathbb{Z}_m$ si y solo si $\text{mcd}(a,m) = 1$, y en tal caso, el inverso es único. Al inverso de la clase $[a]_m$ lo denotamos por $[a]_m^{-1}$. Naturalmente, este inverso no se calcula dividiendo enteros, sino resolviendo la congruencia lineal $a \cdot x \equiv 1 \pmod{m}$.

EJEMPLO 1.4.19 La clase $[6]_{17}$ tiene inverso, ya que $\text{mcd}(17,6) = 1$. Si usamos la forma vectorial del algoritmo de Euclides, calcularemos el inverso a la vez que calculamos el máximo común divisor:

$$\left\lfloor \frac{17}{6} \right\rfloor = 2 \quad \rightsquigarrow \quad \begin{pmatrix} 17 \\ 1 \\ 0 \end{pmatrix} - 2 \begin{pmatrix} 6 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ 1 \\ -2 \end{pmatrix}$$

$$\left\lfloor \frac{6}{5} \right\rfloor = 1 \quad \rightsquigarrow \quad \begin{pmatrix} 6 \\ 0 \\ 1 \end{pmatrix} - \begin{pmatrix} 5 \\ 1 \\ -2 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 3 \end{pmatrix}$$

Recordemos que esto significa que

$$1 = 17 \cdot (-1) + 6 \cdot 3$$

y por lo tanto podemos afirmar que $\text{mcd}(17,1) = 1$ y que 3 es el inverso de 6 módulo 17.

Como podemos observar en este ejemplo, si solo estamos interesados en calcular el inverso de un número, solo necesitamos la tercera componente del vector. Esto permite simplificar el algoritmo de Euclides si lo aplicamos a esta operación:

$$\begin{pmatrix} 17 \\ 0 \end{pmatrix} - 2 \begin{pmatrix} 6 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ -2 \end{pmatrix}$$

$$\begin{pmatrix} 6 \\ 1 \end{pmatrix} - \begin{pmatrix} 5 \\ -2 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \end{pmatrix} = \begin{pmatrix} \text{m.c.d.} \\ \text{Inverso} \end{pmatrix}$$

□

El operador `inv_mod`(a,m) de `Maxima` calcula el inverso de a módulo m

```
(%i1) inv_mod(5,17)
```

$$7$$

```
(%i2) mod(5*7,17)
```

$$1$$

El uso de los inversos, nos da un forma alternativa de resolver las congruencias líneales.

Ejemplo 1.4.20 Utilizamos el inverso calculado en el ejemplo anterior para resolver la siguiente congruencia lineal:

$$\begin{aligned} 6x &\equiv 5 \pmod{17} \\ [6]_{17}[x]_{17} &= [5]_{17} \\ [x]_{17} &= [6]_{17}^{-1}[5]_{17} \\ [x]_{17} &= [3]_{17}[5]_{17} \quad \text{(ejemplo anterior)} \\ [x]_{17} &= [15]_{17} \\ x &= 15 + 17k,\ k \in \mathbb{Z} \end{aligned}$$

□

Ejemplo 1.4.21 La congruencia $12x \equiv 14 \pmod{34}$ tiene dos soluciones módulo 34, ya que $\text{mcd}(12, 34) = 2$ y 14 es múltiplo de 2. Dado que 12 y 34 no son coprimos, no podemos utilizar inversos directamente. Cuando lleguemos a ese punto, basta pasar a la ecuación diofántica para poder simplificar antes de continuar el cálculo con congruencias.

$$\begin{aligned} 12x &\equiv 14 \pmod{34} \\ 12x &= 14 + 34k \\ 6x &= 7 + 17k \\ 6x &\equiv 7 \pmod{17} \\ 3 \cdot 6x &\equiv 3 \cdot 7 \pmod{17} \\ x &\equiv 4 \pmod{17} \end{aligned}$$

Por lo tanto, las dos soluciones módulo 34 son:

$$x_1 \equiv 4 \pmod{34} \quad \text{y} \quad x_2 \equiv 4 + 17 = 21 \pmod{34}$$

□

El teorema de Euler-Fermat nos da una forma alternativa de calcular los inversos modulares. Si $\text{mcd}(a, m) = 1$, entonces $a \cdot a^{\Phi(m)-1} = a^{\Phi(m)} \equiv 1 \pmod{m}$. Por lo tanto, $[a]_m^{-1} = [a^{\Phi(m)-1}]_m$. Sin embargo, aplicar esta fórmula requiere evaluar la función de Euler y simplificar potencias, lo cual es muy costoso computacionalmente.

1.4.6. Sistemas de congruencias lineales

En muchas ocasiones, la resolución de un problema supondrá la satisfacción de más de una congruencia lineal. Por ejemplo, si nos preguntamos por un número que tenga las siguientes características: al dividirse entre 3 da por resto 1, al dividirse entre 5 da por resto 2 y al dividirse entre 7 da por resto 3, estamos buscando un número que satisfaga simultáneamente las siguientes congruencias lineales:

$$\begin{aligned} x &\equiv 1 \pmod{3} \\ x &\equiv 2 \pmod{5} \\ x &\equiv 3 \pmod{7} \end{aligned}$$

Los resultados que nos permiten determinar si estos sistemas tienen solución son los siguientes.

TEOREMA 1.4.22 *Sean $m_1, m_2, \ldots, m_k$ naturales mayores que 1 y $b_1, b_2, \ldots, b_k \in \mathbb{Z}$.*

1. *El sistema de congruencias,*

$$\begin{aligned} x &\equiv b_1 \pmod{m_1} \\ x &\equiv b_2 \pmod{m_2} \\ &\vdots \\ x &\equiv b_k \pmod{m_k} \end{aligned}$$

   tiene solución si, y solo si, los sistemas formados por cada par de congruencias, tiene solución.

2. *Cada sistema*

$$\begin{aligned} x &\equiv b_\alpha \pmod{m_\alpha} \\ x &\equiv b_\beta \pmod{m_\beta} \end{aligned}$$

   tiene solución si y solo si $b_\alpha - b_\beta$ es múltiplo de $d = \text{mcd}(m_\alpha, m_\beta)$; en tal caso, la solución es única módulo $M = \text{mcm}(m_\alpha, m_\beta)$.

Demostración: Vamos a demostrar solo la segunda parte y vamos a utilizar los subíndices 1 y 2 en lugar de α y β para facilitar la lectura.

- La necesidad de la condición $b_1 \equiv b_2 \pmod{d}$ es trivial.
  Si $x = b_1 + k_1 m_1$ y $x = b_2 + k_2 m_2$, entonces restando las dos igualdades miembro a miembro y reordenado los sumandos obtenemos:

$$b_1 - b_2 = k_2 m_2 - k_1 m_1$$

  Y como d divide a m_1 y a m_2, necesariamente debe dividir a $b_1 - b_2$.

- La demostración de que, en tal caso, podemos determinar las soluciones, describe el método de resolución.

  Utilizando el algoritmo de Ecuclides, encontramos la combinación lineal de la identidad de Bézout para m_1 y m_2:

$$d = t_1 \cdot m_1 + t_2 \cdot m_2$$

- Dado que $b_1 - b_2 = k \cdot d$, podemos realizar los siguiente pasos:

$$\begin{aligned} b_1 - b_2 = k \cdot d &= (k \cdot t_1) \cdot m_1 + (k \cdot t_2) \cdot m_2 \\ b_1 - (k \cdot t_1) \cdot m_1 &= b_2 + (k \cdot t_2) \cdot m_2 \end{aligned}$$

Y por lo tanto, este número es solución de las dos ecuaciones.

- Supongamos ahora que x y x' son soluciones del sistema y demostremos que, entonces $x \equiv x' \pmod{M}$:

$$\left.\begin{array}{l} x = b_1 + k_1 m_1 \\ x' = b_1 + k_1' m_1 \end{array}\right\} \implies x - x' = (k_1 - k_1')m_1$$

$$\left.\begin{array}{l} x = b_2 + k_2 m_2 \\ x' = b_2 + k_2' m_2 \end{array}\right\} \implies x - x' = (k_2 - k_2')m_2$$

Por lo tanto, $x - x'$ es múltiplo de m_1 y de m_2 y en consecuencia múltiplo de $M = \text{mcm}(m_1, m_2)$. □

COROLARIO 1.4.23 *Consideremos el siguiente sistema de congruencias*

$$\begin{aligned} x &\equiv a \pmod{n} \\ x &\equiv b \pmod{m} \end{aligned}$$

Si $a - b$ es múltiplo de $d = \text{mcd}(n, m)$, $d = s \cdot n + t \cdot m$, y $M = \text{mcm}(n, m)$, entonces la solución del sistema es

$$x \equiv a - \frac{s \cdot n(a-b)}{d} \pmod{M},$$

EJEMPLO 1.4.24 Vamos a resolver el siguiente sistema de congruencias siguiendo el proceso descrito en la demostración anterior

$$\begin{aligned} x &\equiv 1 \pmod{10} \\ x &\equiv 11 \pmod{15} \end{aligned}$$

Empezamos determinando la identidad de Bézout para 15 y 10:

$$\left\lfloor \frac{15}{10} \right\rfloor = 1 \quad \rightsquigarrow \quad \begin{pmatrix} 15 \\ 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 10 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ 1 \\ -1 \end{pmatrix}$$

Dado que $5|10$ ya podemos afirmar que $\text{mcd}(15, 10) = 5$ y además

$$5 = 15 \cdot 1 + 10 \cdot (-1), \qquad \text{mcm}(15, 10) = \frac{15 \cdot 10}{5} = 30$$

Dado que $11 - 1 = 10$ es múltiplo de 5, podemos afirmar que el sistema de congruencias tiene solución y

$$\begin{aligned} x &\equiv 1 - \frac{(-1)10(1-11)}{5} \pmod{30} \\ x &\equiv -19 \pmod{30} \\ x &\equiv 11 \pmod{30} \end{aligned}$$

□

También podemos resolver los sistemas usando las propiedades de las congruencias como mostramos en el siguiente ejemplo.

Ejemplo 1.4.25 Volvemos a considerar el sistema siguiente

$$\begin{aligned} x &\equiv 1 \pmod{10} \\ x &\equiv 11 \pmod{15} \end{aligned}$$

De la primera ecuación obtenemos que $x = 1 + 10k$, así que vamos a hallar el valor de k para que también se verifique la segunda congruencia.

$$\begin{aligned} x &= 1 + 10k \\ 1 + 10k &\equiv 11 \pmod{15} \quad \text{(sustituimos en la segunda congruencia)} \\ 10k &\equiv 10 \pmod{15} \\ 5k &\equiv 5 \pmod{15} \quad \text{(cancelación, ya que } \operatorname{mcd}(2,15) = 1) \\ 5k &= 5 + 15m \\ k &= 1 + 3m \\ x &= 1 + 10(1 + 3m) = 11 + 30m \\ x &\equiv 11 \pmod{30} \end{aligned}$$

□

Como consecuencia del teorema general, si los módulos son coprimos dos a dos, entonces el sistema tiene solución. Este caso particular se conoce como teorema chino del resto.

Corolario 1.4.26 (Teorema chino del resto) *Sean $m_1, m_2, \ldots, m_k$ enteros positivos mayores que 1 y coprimos dos a dos y sean $b_1, b_2, \ldots, b_k \in \mathbb{Z}$. Entonces, el siguiente sistema de congruencias, tiene solución:*

$$\begin{aligned} x &\equiv b_1 \pmod{m_1} \\ x &\equiv b_2 \pmod{m_2} \\ &\vdots \\ x &\equiv b_k \pmod{m_k} \end{aligned}$$

Además, si x y x' son soluciones, entonces $x \equiv x' \pmod{m_1 m_2 \cdots m_k}$.

Ejemplo 1.4.27 Resolvamos el sistema planteado en la introducción de esta sección:

$$\begin{aligned} x &\equiv 1 \pmod{3} \\ x &\equiv 2 \pmod{5} \\ x &\equiv 3 \pmod{7} \end{aligned}$$

1. De la primera ecuación, deducimos que $x = 3k + 1$. Utilizamos esta igualdad para sustituir x en la segunda ecuación, sobre la que trabajaremos usando la representación de clases de congruencia:

$$\begin{aligned} [x]_5 = [3k+1]_5 &= [2]_5 \\ [3k]_5 &= [1]_5 \\ [k]_5 &= [3]_5^{-1}[1]_5 = [3^3]_5 = [2]_5 \\ k &= 5m + 2 \\ x &= 3k + 1 = 3(5m+2) + 1 = 15m + 7 \end{aligned}$$

2. A continuación llevamos esta expresión de x a la última congruencia:

$$\begin{aligned} [x]_7 = [15m+7]_7 &= [3]_7 \\ [m]_7 &= [3]_7 \qquad (\text{Ya que } 15 \equiv 1 \pmod 7,\ 7 \equiv 0 \pmod 7) \\ m &= 7q + 3 \\ x &= 15m + 7 = 15(7q+3) + 7 = 105q + 52 \\ x &\equiv 52 \pmod{105} \end{aligned}$$

□

Ejemplo con Maxima 1.4.28 El operador

`chinese(coefs,mods)`

determina las soluciones de un sistema de congruencias con solución o devuelve `false` si no tiene solución. El sistema que hemos resuelto en el ejemplo anterior se escribiría:

```
(%i1) chinese([1,2,3],[3,5,7]);
```

52

Otro ejemplos con módulos que no son coprimos dos a dos:

```
(%i2) chinese([14,20,34],[15,21,35]);
```

104

```
(%i3) chinese([13,20,33],[15,21,35]);
```

false □

Ejemplo 1.4.29 Vemos un último ejemplo en el que la solución no es única módulo el mínimo común múltiplo de los módulos, pero el método aprendido sigue siendo aplicable.

$$\begin{aligned} 8x &\equiv 2 \pmod{30} \\ x &\equiv 10 \pmod{21} \end{aligned}$$

Empezamos por la segunda congruencia, que ye está resuelta, $x = 10 + 21q$, y sustituimos en la primera:

$$\begin{aligned}
8(10+21q) &\equiv 2 \pmod{30}\\
8(10-9q) &\equiv 2 \pmod{30}\\
80-72q) &\equiv 2 \pmod{30}\\
-72q &\equiv -78 \pmod{30}\\
72q &\equiv 78 \pmod{30}\\
12q &\equiv 18 \pmod{30}\\
2q &\equiv 3 \pmod{5}\\
3\cdot 2q &\equiv 3\cdot 3 \pmod{5}\\
q &\equiv 4 \pmod{5}\\
x &= 10+21(4+5m) = 94+105m\\
x &\equiv 94 \pmod{105}
\end{aligned}$$

Los módulos del sistema inicial eran 21 y 30, por lo que las soluciones se deben expresar módulo $\text{mcm}(21,30) = 210 = 2\cdot 105$. Por lo tanto, el sistema tiene dos soluciones:

$$x \equiv 94 \pmod{210}, \qquad x \equiv 94+105 = 199 \pmod{210} \qquad \square$$

1.5. Sistema de encriptación RSA

El método de encriptación de mensajes conocido como RSA recibe ese nombre por las iniciales de los investigadores Rivest, Shamir y Adleman. El funcionamiento de las claves de encriptación y desencriptación se basa en el calculo con aritmética modular que hemos aprendido. Antes de ver el fundamento matemático, vamos a presentarlo con un pequeño ejemplo.

Mark e Iván se van a comunicar con mensajes encriptados con el método RSA en un entorno determinado por el número $m = 143$. Para recibir mensajes encriptados, Mark dispone de dos claves:

$$\text{Clave pública de Mark} = e = 53, \qquad \text{Clave privada de Mark} = d = 77$$

La clave pública sirve para encriptar el mensaje y debe ser conocida por cualquiera que quiera enviarle un mensaje encriptado. Sin embargo, la clave privada, que sirve para desencriptar, solo debe ser conocida por Mark. Si Iván quiere enviar el mensaje formado por la letra K, en primer lugar deberemos convertir ese mensaje en un número. Esta conversión se hace utilizando las tablas de códigos ASCII o Unicode. En este ejemplo, utilizaremos números decimales de dos dígitos, aunque para una tabla completa tendríamos que recurrir a tres dígitos decimales. La siguiente tabla muestra parte de los códigos ASCII:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
32	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
P	Q	R	S	T	U	V	W	X	Y	Z	0	1	2	3	4
80	81	82	83	84	85	86	87	88	89	90	48	49	50	51	52
5	6	7	8	9	.	?	!	@							
53	54	55	56	57	46	63	23	54							

Por lo tanto, dado que el código ASCII de la letra K es 75, para encriptar el mensaje destinado a Mark, Iván calcula el resto de dividir 75^{53} entre 143:

```
(%i1) power_mod(75,53,143);
```

69

Cuando Mark recibe el mensaje 69, tiene que desencriptarlo y lo hace calculando el resto de dividir 69^{77} entre 143:

```
(%i2) power_mod(69,77,143);
```

75

En este entorno de comunicación segura, Iván dispone de sus propias claves publica y privada:

Clave pública de Iván $= e = 47$, Clave privada de Iván $= d = 23$

De esta forma, si Mark quiere enviarle el mensaje S, que en ASCII se corresponde con el número 83, le enviará el resto de dividir 83^{47} entre 143:

```
(%i3) power_mod(83,47,143);
```

8

E Iván lo desencriptará calculando el resto de dividir 8^{23} entre 143:

```
(%i4) power_mod(8,23,143);
```

83

Pero, ¿cómo funciona este sistema de encriptación y desencriptación? ¿en qué se basa su seguridad?

1.5.1. Determinación de las claves

En principio, el número m que define el entorno de comunicación puede ser definido arbitrariamente, aunque luego veremos la restricciones necesarias para que el

sistema resulte realmente seguro. Una vez elegido el número m, necesitaremos determinar pares de claves pública y privada para cada usuario y para ello recordamos el teorema de Euler-Fermat: si a y m son coprimos, entonces

$$\begin{aligned} a^{\phi(m)} &\equiv 1 \pmod{m} \\ a^{k\cdot\phi(m)} &\equiv 1 \pmod{m} \quad \text{para cualquier } k \in \mathbb{Z} \\ a^{1+k\cdot\phi(m)} &\equiv a \pmod{m} \quad \text{para cualquier } k \in \mathbb{Z} \end{aligned}$$

Por lo tanto, si elegimos dos números e y d tales que $e \cdot d = 1 + k \cdot \phi(m)$ para algún $k \in \mathbb{Z}$, esos números nos servirían como claves pública y privada. Es decir, las claves son número inversibles módulo $\phi(m)$, es decir, coprimos con $\phi(m)$, ya que de esta forma, los exponentes e y d funcionan efectivamente como encriptador y desencriptador del mensaje representado por el número a:

$$(a^e)^d = a^{e\cdot d} = a^{1+k\cdot\phi(m)} \equiv a \pmod{m}$$

Volviendo al ejemplo inicial para ver cómo hemos elegido los números:

$$\begin{aligned} m &= 11 \cdot 13 = 143 \\ \phi(m) &= (11-1)(13-1) = 120 \end{aligned}$$

Los números 53 y 47, elegidos como claves privadas en el ejemplo, son coprimos con 120 y por eso ha sido posible calcular sus inversos módulo 120, lo que determina los pares de claves:

```
(%i4) priv_mark: 77$
pub_mark: inv_mod(priv_mark,120);
```

53

```
(%i5) priv_ivan: 23$
pub_ivan: inv_mod(priv_ivan,120);
```

47

Teniendo en cuenta lo anterior, solo podríamos enviar mensajes representados por números coprimos con m, sin embargo, si elegimos el número m de forma adecuada, no es necesaria esta restricción. Concretamente, basta tomar un número dado por el producto de dos números primos:

TEOREMA 1.5.1 *Si p y q son números primos y $m = p \cdot q$, entonces $a^{1+k\cdot\phi(m)} \equiv a \pmod{m}$ para todo a.*

Dado que p y q son primos, $\phi(m) = \phi(p \cdot q) = (p-1)(q-1)$.

$$a^{1+k(p-1)(q-1)} = a \cdot (a^{p-1})^{k(q-1)} \equiv \begin{cases} a \pmod{p} & \text{si } \operatorname{mcd}(a,p) = 1 \\ 0 \pmod{p} & \text{si } p|a \end{cases}$$

El primer caso es consecuencia del teorema de Euler-Fermat y el segundo de la definición de congruencia. Pero en cualquiera de los dos casos, $a^{1+k(p-1)(q-1)} \equiv a$ (mód p). De la misma forma, se demuestra que $a^{1+k(p-1)(q-1)} \equiv a$ (mód q) y, en consecuencia,

$$a^{1+k(p-1)(q-1)} \equiv a \pmod{p \cdot q}$$

Atendiendo a esto, el número m se construye como producto de dos números primos p y q suficientemente grandes. Esto garantiza que todos los mensajes pueden ser encriptados y además dificulta la factorización de m, necesaria para "romper" la seguridad del sistema.

1.5.2. Descripción del sistema

En primer lugar para construir el sistema de encriptado siguiendo el método RSA, necesitamos convertir los mensajes en números. Tal y como hemos dicho, esto se hace habitualmente con las tablas de códigos ASCII o Unicode. Por ejemplo, el mensaje "HOLA MUNDO!" se traduciría como:

$$727976653277857868 7923$$

A continuación el mensaje se divide en bloques con el mismo número de dígitos, de forma que cada bloque se encriptará por separado. Además, el número máximo que aparezca en un bloque deberá ser menor que el número m que elijamos para determinar las claves. Por ejemplo, si queremos tomar bloques de cuatro dígitos, tendríamos que encriptar y enviar los siguientes números:

$$a_1 = 7279, \quad a_2 = 7665, \quad a_3 = 3277, \quad a_4 = 8578, \quad a_5 = 6879, \quad a_6 = 2300$$

Y para encriptar tendríamos que considerar un número m mayor que 10^4 y que esté dado por el producto de dos números primos. Por ejemplo:

$$m = 79 \cdot 127 = 10033$$

Las claves podrán ser cualquier número coprimo con $\phi(m) = (79-1)(127-1) = 9828 = 2^2 \cdot 3^3 \cdot 7 \cdot 13$. Por ejemplo, podemos elegir $e = 817$ como clave pública y $d = 409$ sería la correspondiente clave privada:

```
(%i6) privada: 409$
publica: inv_mod(privada,9828);
```

817

Vamos a definir en Maxima las funciones que nos permiten encriptar y desencriptar mensajes:

```
(%i7) encriptar(m):= power_mod(m, publica,10033)$
(%i8) desencriptar(m):= power_mod(m, privada,10033)$
```

De esta forma, encriptamos el mensaje “HOLA MUNDO!” con:

```
(%i9) mensaje: [7279,7665,3277,8578,6879,2300]$
(%i10) map(encriptar,mensaje);
```

$$[5720, 4681, 8471, 2751, 3465, 8809]$$

Y comprobamos que al desencriptar obtenemos la misma lista de números:

```
(%i11) map(desencriptar,[5720,4681,8471,2751,3465,8809]);
```

$$[7279, 7665, 3277, 8578, 6879, 2300]$$

Ejercicios 1

1. Halla la representación decimal de los siguientes números expresados en las bases indicadas:

   $a)$ $10\ 011\ 101_{(2}$ $\qquad$ $b)$ $1231_{(7}$

2. Halla la representación en las bases $2, 8$ y 11 de los siguientes números expresados en base decimal:

   $a)$ 237 $\qquad$ $b)$ 2002

3. Halla el valor de $x \in \mathbb{N}$ e $y \in \mathbb{N}$ para que se verifiquen las siguientes igualdades:

   $a)$ $331_{(x} = 106_{(11}$ $\qquad$ $b)$ $274_{(8} = y_{(2}$

4. Da ejemplos de enteros a, b y c tales que

   $a)$ a es divisor de $b \cdot c$, pero a no es divisor de b ni de c.

   $b)$ a y b son divisores de c, pero $a \cdot b$ no es divisor de c.

5. Demuestra, sin calcularlo, que $2020^4 + 2020^3 - 2020 - 1$ no es un número primo.

6. (`Maxima`) Define de forma recursiva, con recursión de cola, la función `suma`(n) que calcula la suma de los números naturales del 1 a n.

7. (`Maxima`) Define de forma recursiva `reves`(ℓ) que devuelva la lista ℓ pero con el orden de sus elementos invertido. Por ejemplo: `reves`$([a, b, c]) = [c, b, a]$.

Ejercicios 2

1. En cada uno de los siguientes apartados, expresa el $\mathrm{mcd}(a, b)$ como combinación lineal de a y b.

   $a)$ $a = 16,\ b = 135$ $\quad$ $b)$ $a = 55,\ b = 34$ $\quad$ $c)$ $a = 107,\ b = 23$

2. Estudia si las siguientes ecuaciones tienen solución y en tal caso, encuentra todas las soluciones.

   $$(1)\quad 42x + 312y = 834, \qquad\qquad (2)\quad 144x + 702y = 9$$

3. Para tender un tramo de vía de 122 m. se dispone de barras de 30 m. y de 16 m. de largo. ¿Es posible cubrir el tramo utilizando solamente ese tipo de barras? Si es posible, determina cuántas barras de cada longitud se necesitan para cubrir los 122m.

4. Enviamos por correo dos tipos de paquetes A y B. Por enviar los del tipo A nos cobran 15 céntimos de euro más que por los del tipo B. Sabiendo que hemos enviado más paquetes del tipo B que del tipo A, que en total hemos enviado 12 paquetes y que nos han cobrado un total de 13 euros con 20 céntimos, ¿cuántos hemos enviado de cada tipo y qué nos han cobrado por cada uno?

5. (`Maxima`) Resuelve la ecuación diofántica $4x+6y+7z = 12$ siguiendo el siguiente procedimiento:

   $a)$ Aplica el cambio de variable $u = 2x+3y$ y resuelve, con ayuda de Maxima, la ecuación diofántica en u y z obtenida.

   $b)$ Para cualquier solución u obtenida en el apartado anterior, resuelve con ayuda de Maxima, la ecuación diofántica $2x + 3y = u$ en x e y.

   $c)$ Razona la posibilidad de utilizar los pasos anteriores para resolver cualquier ecuación de 3 o más incógnitas; ¿qué otros cambios de variable podríamos haber utilizado?

6. (`Maxima`) Hemos definido la función `mcdex` utilizando las diferencias sucesivas en lugar de los cocientes, tal y como aparece en la descripción del algoritmo de Euclides: da una definición recursiva (con recursión de cola) para `mcdex` en la que se utilicen los cocientes sucesivos.

7. (`Maxima`) Da una definición recursiva de la función

   $$\texttt{bezout\_mat}(n, m) = \begin{pmatrix} s & t \\ \pm m/d & \mp n/d \end{pmatrix}$$

   en donde $n \cdot s + m \cdot t = d = \mathrm{mcd}(n, m)$ es la identidad de Bézout para n y m.

Ejercicios 3

1. Resuelve, si es posible, las siguientes congruencias lineales:

   $a)\ 3x \equiv 1 \pmod{12}$ $b)\ 3x \equiv 1 \pmod{11}$ $c)\ 64x + 11 \equiv 43 \pmod{84}$

2. En los apartados siguientes, calcula el menor entero positivo x que verifique la relación:

   $a)\ 3^{201} \equiv x \pmod{11}$ $b)\ 2^{11} \cdot 3^{13} \equiv x \pmod{7}$

3. Calcula el resto de dividir 100^{101} entre 7.

4. Resuelve, cuando sea posible, los sistemas:

   $$a)\begin{cases} x \equiv -2 \pmod{5} \\ 2x \equiv 1 \pmod{7} \\ x \equiv 3 \pmod{8} \end{cases} \quad b)\begin{cases} x \equiv 5 \pmod{6} \\ x \equiv 3 \pmod{10} \\ x \equiv 8 \pmod{15} \end{cases} \quad c)\begin{cases} x \equiv 2 \pmod{14} \\ x \equiv 10 \pmod{30} \\ x \equiv 6 \pmod{21} \end{cases}$$

5. Encuentra el menor entero positivo cuyo resto cuando se divide por 11 es 8, que tiene el último dígito igual a 4 y es divisible por 27.

6. Un tesoro escondido de monedas de oro pasa a "ser propiedad" de una banda de 15 piratas. Cuando empiezan a repartirse las monedas, les sobran 3 monedas. La discusión por el reparto se "anima" y sólo quedan 7 piratas, pero, cuando se reparten las monedas entre ellos, sobran 2. La discusión continua y el número de piratas se reduce a 4, que sí consiguen repartirse todas las monedas. ¿Cúal es el mínimo número de monedas que podía haber en el tesoro?

7. Utiliza la congruencia módulo 9 para encontrar el dígito x en el producto: $89878 \cdot 58965 = 5299x56270$.

8. Determina la máxima potencia de 2 que divide a cada uno de los enteros siguientes:

   $a)\ 1423408$ $b)\ 41578912246$

9. La letra asociada al DNI en el NIF indica la clase del número módulo 23, teniendo en cuenta la siguiente correspondencia:

0	1	2	3	4	5	6	7	8	9	10	11
T	R	W	A	G	M	Y	F	P	D	X	B
12	13	14	15	16	17	18	19	20	21	22	
N	J	Z	S	Q	V	H	L	C	K	E	

   En cada apartado, determina si existe un dígito α que haga que el NIF resultante sea válido

   $a)\ 24\alpha 67890A$ $b)\ 2\alpha 041327Y$ $c)\ 4459\alpha 203D$

10. Sea p un número primo. Demuestra que $n^p \equiv n \pmod{p}$ para todo entero positivo n.

11. Demuestra la siguiente propiedad: si $\text{mcd}(n, m) = 1$, entonces

$$x \equiv a \pmod{n \cdot m} \qquad \text{si y solo si} \qquad \begin{cases} x \equiv a \pmod{n} \\ x \equiv a \pmod{m} \end{cases}$$

TEMA 2

Conjuntos, funciones, recuento

2.1. Conjuntos y funciones

2.1.1. Teoría intuitiva de conjuntos

La *teoría de conjuntos* es el fundamento de la matemáticas y de diversas áreas de las Ciencias de la Computación. Formalmente, se desarrolla sobre un *Sistema Axiomático*, pero queda fuera de los objetivos del curso trabajar estas teorías de forma general. Sin embargo, sí es necesario conocer sus elementos básicos, lo que se denomina *Teoría Intuitiva de Conjuntos*. En esta sección introducimos los conceptos y operadores básicos para trabajar con conjuntos.

Podemos decir de forma sintética que la Teoría de Conjuntos es el área de las matemáticas que estudia la *relación de pertenencia* entre *elementos* y *conjuntos*.

> $x \in A$, leído "x *pertenece* a A", indica que A es un *conjunto* (colección de objetos) y x es uno de sus *elementos*.

También utilizamos la notación $x \notin A$ para indicar que x no es elemento de A.

Ejemplo 2.1.1

✎ $\in \{✂, ✎, 👁\}$ ☞ $\notin \{✂, ✎, 👁\}$

□

Por lo tanto, todas las operaciones, relaciones y propiedades dentro de la teoría de conjuntos se definen a partir de la relación de pertenencia.

Definición 2.1.2 (Axioma de extensión) *Dos conjuntos A y B son* iguales *si tienen los mismos elementos; es decir:*

$$A = B \quad \overset{def.}{\iff} \quad \forall x \Big(x \in A \ \leftrightarrow \ x \in B \Big)$$

Es decir, un conjunto queda completamente especificado por sus elementos. Si describimos un conjunto dando la lista de sus elementos encerrada entre llaves, decimos que se define por *extensión*, como los que hemos presentado en el ejemplo anterior.

Debemos tener en cuenta que, atendiendo a la definición de igualdad, en las representaciones por extensión, no importa el orden en que se escriban los elementos ni las posibles repeticiones.

Ejemplo 2.1.3

$$\{a,b,c\} = \{b,a,c\}; \qquad \{a,b,a\} = \{a,b,b\} = \{a,b\}$$ □

Ejemplo 2.1.4 Conjuntos numéricos:

- Conjunto de los números *naturales*:

$$\mathbb{N} = \{0,1,2,3,\dots\}$$

- Conjunto de los números *enteros*:

$$\mathbb{Z} = \{\dots,-3,-2,-1,0,1,2,3,\dots\}$$

- Conjunto de los números *enteros positivos*:

$$\mathbb{Z}^+ = \{1,2,3,\dots\}$$

- Conjuntos de los números *racionales*:

$$\mathbb{Q} = \left\{\pm\frac{p}{q};\ p,q \in \mathbb{N}, p, q \text{ primos entre sí}\right\}$$ □

Ejemplo con Maxima 2.1.5 Los conjuntos se definen en `Maxima` tal y como lo hemos hecho antes, delimitando sus elementos entre llaves.

```
(%i1) A: {2,3,5,5,3,-1,2,1};
```

$$\{-1,1,2,3,5\}$$

En este ejemplo podemos observar que `Maxima` elimina los elementos repetidos aunque nosotros los hayamos escrito. También observamos que ha escrito los elementos, numéricos en este caso, de forma ordenada. Tal y como hemos dicho, este orden no es importante en la estructura del conjunto, pero sí es utilizado en los algoritmos internos de `Maxima` para trabajar de forma más eficiente.

También podemos definir conjuntos con el operador `set()`, que será útil en combinación con otros operadores.

```
(%i2) set(2,2,3,1,5);
```

$$\{1,2,3,5\}$$

Si queremos definir un conjunto de cadenas de caracteres (strings), debemos delimitarlas por comillas, en caso contrario, `Maxima` lo interpretará como un parámetro. En el siguiente ejemplo vamos a utilizar el parametro x y también la cadena de caracteres "x".

```
(%i3) B: set(x,"x");
```

$$\{x,x\}$$

Aparentemente, el conjunto está formado por un único elemento repetido. Pero si ahora le damos un valor al parámetro x, vemos que el conjunto está formado por la cadena 'x' y el valor que tome el parámetro x.

```
(%i4) ev(B,x=1);
```

$$\{1,x\}$$

Cuando trabajamos con conjuntos grandes, puede ser útil el operador `elementp` que analiza si un elemento pertenece o no a un conjunto.

```
(%i5) C: {0,1,-1,3,5,-7};
```

$$\{-7,-1,0,1,3,5\}$$

```
(%i6) elementp(3,C);
```

```
true
```

```
(%i7) elementp(2,C);
```

```
false
```

□

La otra relación básica en la teoría de conjuntos es la relación de inclusión, que no debe confundirse con la relación de pertenencia.

Definición 2.1.6 (Inclusión) *Dados los conjuntos A y B, se dice que A está* contenido *en B si todo elemento de A es tambien un elemento de B; es decir:*

$$A \subseteq B \quad \overset{def.}{\iff} \quad \forall x(x \in A \ \rightarrow \ x \in B)$$

Si $A \subseteq B$, también decimos que A es un *subconjunto* de B. Si $A \subseteq B$ y $A \neq B$, decimos que A es un *subconjunto propio* de B y lo denotamos $A \subset B$.

Teorema 2.1.7 *$A = B$ si y solo si $A \subseteq B$ y $B \subseteq A$.*

Debemos tener mucho cuidado y no confundir la relación de pertenencia $\in$, y la relación de inclusión $\subseteq$; por ejemplo:

$$a \in \{a,b,c\}, \qquad a \not\subset \{a,b,c\}, \qquad \{a\} \subset \{a,b,c\}, \qquad \{a\} \not\in \{a,b,c\}$$

Definición 2.1.8 (Axioma del conjunto vacío) *Existe un conjunto que no tiene elementos. Este conjunto se denomina* conjunto vacío *y se denota por* $\varnothing$.

Teorema 2.1.9 *Para todo conjunto* A*, se verifica que* $\varnothing \subseteq A$.

La demostración de este resultado es bastante simple: no es posible encontrar ningún elemento de $\varnothing$ que no esté en A, ya que $\varnothing$ no contiene elementos.

Ejemplo con Maxima 2.1.10 En `Maxima`, disponemos de los operadores `subsetp` y `setequalp` para analizar la relación de inclusión o igualdad de dos conjuntos,

```
(%i1) A: {-2,-1,0,2,4,6}$
      B: {-1,0,2}$
      C: {-1,0,1}$
      D: {0,-1,-2,2,4,6}$

(%i2) subsetp(B,A);

                              true

(%i3) subsetp(C,A);

                              false

(%i4) setequalp(A,D);

                              true
```

El conjunto vacío lo podemos introducir con un par de llaves o con `set()`.

```
(%i5) setequalp({},set());

                              true

(%i6) subsetp({},A);

                              true
```

□

Definición 2.1.11 (Axioma de la unión) *Si* A *y* B *son conjuntos, existe otro conjunto formado exactamente por los elementos de* A *y* B*:*

$$x \in A \cup B \quad \overset{def.}{\iff} \quad x \in A, \quad \textit{o bien} \quad x \in B$$

Ejemplo 2.1.12

- $\{a, b\} \cup \{c, d\} = \{a, b, c, d\}$
- $\{a, b, c\} \cup \{b, c, d\} = \{a, b, c, d\}$ □

Teorema 2.1.13 (Caracterización de la unión) *Sean A, B y C tres conjuntos: $A \cup B \subseteq C$ si, y solo si, $A \subseteq C$ y $B \subseteq C$.*

Mostramos la demostración de este teorema como ejemplo de prueba de un enunciado basada en enunciados (definiciones y teoremas) previos.

Ejemplo 2.1.14 Demostración del teorema anterior. Dado que el enunciado de este teorema establece la equivalencia de dos enunciados, tenemos que probar que a partir de cada uno de ellos podemos demostrar el otro. Separamos las dos direcciones de la prueba.

- ($\Rightarrow$) Supongamos que $A \cup B \subseteq C$. Si $x \in A$, entonces $x \in A \cup B$ (definición de $\cup$) y por lo tanto, $x \in C$ (ya que $A \cup B \subseteq C$). En consecuencia, $A \subseteq C$. Análogamente probamos que $B \subseteq C$.
- ($\Leftarrow$) Supongamos que $A \subseteq C$ y $B \subseteq C$. Si $x \in A \cup B$, entonces, o bien $x \in A$ o bien $x \in B$. En el primer caso, $x \in A$, tenemos que $x \in C$ (ya que $A \subseteq C$); en el otro caso, $x \in B$, también tenemos que $x \in C$ (ya que $B \subseteq C$). Por lo tanto, en cualquier caso $x \in C$ y podemos afirmar que $A \cup B \subseteq C$. □

Ejemplo con Maxima 2.1.15 En `Maxima`, la unión de dos conjuntos se determina con el operador `union`.

```
(%i1) A: {-1,1,2,3,5}$
      B: {2,3}$

(%i2) union(A,B)
```

$$\{-1, 1, 2, 3, 5\}$$ □

Otra forma de definir conjuntos es por *compresión*, es decir, mediante una propiedad que caracteriza los elementos de un conjunto.

Definición 2.1.16 (Axioma de comprensión) *Si A es un conjunto y P es una propiedad aplicable a elementos de A, existe un subconjunto de A formado por los elementos de A que verifican la propiedad P. Este conjunto se escribe como*

$$\{x \in A \mid P(x)\}$$

Obsérvese, que en este caso, lo que definimos realmente es un subconjunto de un conjunto: $\{x \in A \mid P(x)\} \subseteq A$.

Ejemplo 2.1.17

- $\{x \in \mathbb{N} \mid x \text{ es primo }\}$
- $\{x \in \mathbb{R} \mid x^2 - 5x + 6 = 0\} = \{2, 3\}$
- $\{x \in \mathbb{R} \mid x^2 + 1 = 0\} = \varnothing$
- $\{x \in \mathbb{C} \mid x^2 + 1 = 0\} = \{\mathrm{i}, -\mathrm{i}\}$ □

La operación de *intersección* entre conjuntos se define usando comprensión.

Definición 2.1.18 (Intersección) *La* intersección *de A y B es el conjunto formado por los elementos de A que también son elementos de B:*

$$A \cap B = \{x \in A \mid x \in B\}$$

Por lo tanto, en particular $A \cap B \subseteq A$. Además, equivalentemente, podemos definir $A \cap B = \{x \in B \mid x \in A\} \subseteq B$.

Teorema 2.1.19 (Caracterización de la intersección) *Sean A, B y C tres conjuntos: $C \subseteq A \cap B$ si, y solo si, $C \subseteq A$ y $C \subseteq B$.*

Ejemplo 2.1.20 Vamos a demostrar el teorema anterior como un ejemplo más de prueba.

- ($\Rightarrow$) Supongamos que $C \subseteq A \cap B$. Si $x \in C$, entonces $x \in A \cap B$ y por lo tanto, $x \in A$ y $x \in B$. En consecuencia, $C \subseteq A$ y $C \subseteq B$.
- ($\Leftarrow$) Supongamos que $C \subseteq A$ y $C \subseteq B$. Si $x \in C$, entonces, por la definición de inclusión, $x \in A$ y $x \in B$ y por lo tanto, $x \in A \cap B$. En consecuencia $C \subseteq A \cap B$. □

Ejemplo con Maxima 2.1.21 Podemos definir conjuntos por comprensión en `Maxima` de dos formas, con los operadores `makeset` y `subset`. El operador `makeset` tiene tres argumentos: el primero es una expresión que determina los elementos del conjunto a partir de unos parámetros; el segundo argumento es la lista de los parámetros que se usan en el argumento anterior y el tercer argumento es una lista con los distintos valores que toman los parámetros para construir los elementos del conjunto.

```
(%i1) makeset(j/k,
              [j,k],
              [[1,a],[1,b],[2,b],[3,c]]);
```

$$\left\{\frac{1}{a},\frac{1}{b},\frac{2}{b},\frac{3}{c}\right\}$$

```
(%i2) makeset(x^2,[x],{[2],[-2],[3]});
```

$$\{4,9\}$$

El operador `subset` es más parecido a la definición matemática de la construcción por compresión. Ese operado tiene dos argumentos: el primero es un conjunto previamente definido y el segundo es una propiedad, es decir, un operador de `Maxima` que actúe sobre un argumento y cuya salida sea `true` o `false`. Por ejemplo, el operador `evenp` es uno de estos operadores; aplicado a un número, nos devuelve `true` o `false` según el número sea par o impar:

```
(%i3) evenp(4);
```

```
true
```

```
(%i4) evenp(5);
```

```
false
```

Vamos a utilizar este operador para determinar el subconjunto de los números pares de un conjunto de números.

```
(%i5) A: {1,27,8,9,12}$
```

```
(%i6) subset(A,evenp);
```

$$\{8,12\}$$

En `Maxima` hay varios operadores que podemos usar como propiedades para definir subconjuntos, pero también podemos definir otras propiedades con el operador `is`() cuya sintaxis vemos en el siguiente ejemplo en el que definimos una propiedad que analiza si un número es o no múltiplo de 3.

```
(%i7) prop(x):= is(remainder(x,3)=0)$
```

```
(%i8) subset(A,prop);
```

$$\{9,12,27\}$$

Ya hemos dicho anteriormente que `Maxima` trabaja internamente con los conjuntos como si fueran listas, es decir, utilizando algún orden intrínseco de los elementos. De hecho, para definir y trabajar con nuestros propios conjuntos también es preferible utilizar la estructura de lista, es decir, con el operador `makelist` en lugar de `makeset` y con el operador `setify` que convierte listas en conjuntos.

```
(%i9) C: makelist(k^2,k,-5,5);
```

$$[25, 16, 9, 4, 1, 0, 1, 4, 9, 16, 25]$$

```
(%i10) setify(C);
```

$$\{0, 1, 4, 9, 16, 25\}$$

Vemos otro ejemplo, en el que además hacemos uso de la propiedad `primep`, para determinar los números primos menores que 50.

```
(%i11) D: setify(makelist(i,i,1,50))$
```

```
(%i12) subset(D,primep);
```

$$\{2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47\} \qquad \square$$

Definición 2.1.22 (Diferencia) *La* diferencia *de dos conjuntos A y B es el conjunto*

$$A - B = \{x \in A \mid x \notin B\}$$

También es frecuente encontrar el símbolo $\setminus$ para representar la diferencia de conjuntos,

$$A \setminus B = A - B$$

y evitar la confusión con la diferencia de números. Por otra parte, $A-B$ se denomina igualmente *complemento (relativo)* de B en A.

Ejemplo 2.1.23

$$\{1, 2, 8, 9\} - \{2, 6, 7\} = \{1, 8, 9\}; \qquad \{2, 6, 7\} - \{1, 2, 8, 9\} = \{6, 7\} \qquad \square$$

Ejemplo con Maxima 2.1.24 El operador de `Maxima` para calcular la intersección de dos conjuntos es `intersection`.

```
(%i1) A: {-1,1,2,3,5}$
      B: {2,4,5}$
```

```
(%i2) intersection(A,B);
```

$$\{2, 5\}$$

Y `setdifference` determina la diferencia de dos conjuntos

```
(%i3) setdifference({1, 2, 8, 9}-{2, 6, 7});
```

$$\{1, 8, 9\}$$

```
(%i4) setdifference({2, 6, 7}-{1, 2, 8, 9});
```

$$\{6, 7\}$$ □

En muchas aplicaciones concretas, es habitual trabajar dentro de un *Universo*, es decir, un conjunto que contiene a todos los conjuntos con los que trabajamos en dicha aplicación. En estos casos, algunas notaciones se pueden simplificar. Por ejemplo, si trabajamos en un universo $\mathcal{U}$, hablaremos simplemente de conjunto *complementario* de A

$$\overline{A} = \mathcal{U} - A$$

Ejemplo 2.1.25 Vamos a demostrar que si A y B son conjuntos en un universo $\mathcal{U}$, entonces

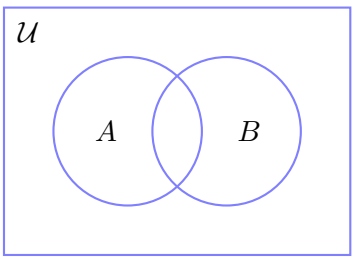
$$A - B = A \cap \overline{B}$$

Concretamente, usamos la caracterización de la intersección y demostramos que $A - B \subseteq A \cap \overline{B}$ y $A \cap \overline{B} \subseteq A - B$.

- ($\subseteq$) Supongamos que $x \in A - B$. Entonces, por la definición de diferencia: $x \in A$ y $x \notin B$. Si $x \notin B$, por la definición del complementario $x \in \overline{B}$ Por lo tanto, $x \in A$ y $x \in \overline{B}$, es decir, $x \in A \cap \overline{B}$.
- ($\supseteq$) La demostración es similar (se propone como ejercicio). □

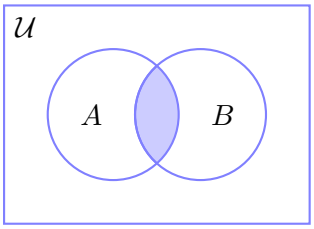
De hecho, en adelante consideraremos la igualdad $A - B = A \cap \overline{B}$ como la definición de la diferencia de conjuntos.

Digramas de Venn. Los diagramas de Venn son una representación esquemática de conjuntos. Estos diagramas están pensados para visualizar las relaciones y operaciones entre dos o más conjuntos. Representamos con un rectángulo el universo y dentro dibujamos dos o más círculos para representar conjuntos dentro de ese universo.

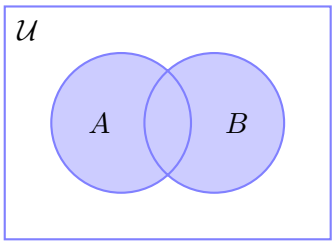


De esta forma, coloreando o sombreando regiones, podemos visualizar las operaciones que hemos definido hasta ahora.

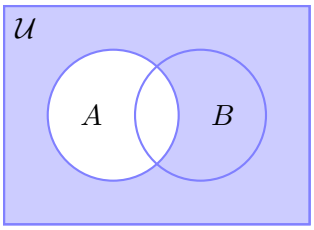
La región sombreada en la figura, corresponde a $A \cap B$:



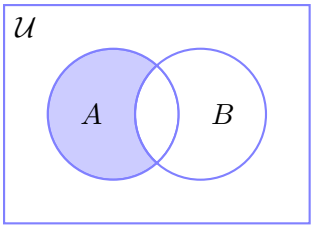
La región sombreada en la figura, corresponde a $A \cup B$:



La región sombreada en la figura corresponde con el complementario de A en el universo, $\overline{A} = \mathcal{U} - A$:



La región sombreada en la figura corresponde con el complementario de B en A, es decir $A - B = A \cap \overline{B}$:



TEOREMA 2.1.26 *Si A, B y C son conjuntos, se verifican las siguientes propiedades:*

- *Conmutativa:* $A \cup B = B \cup A$; $\quad A \cap B = B \cap A$
- *Asociativa:* $A \cup (B \cup C) = (A \cup B) \cup C$; $\quad A \cap (B \cap C) = (A \cap B) \cap C$
- *Absorción:* $A \cup (A \cap B) = A$; $\quad A \cap (A \cup B) = A$
- *Idempotencia:* $A \cup A = A$; $A \cap A = A$
- *Identidad:* $\varnothing \cup A = A$
- *Dominancia:* $\varnothing \cap A = \varnothing$
- *Distributiva:* $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$; $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
- *Cota* $(\cup)$*:* $A \subseteq B$ *si, y solo si,* $A \cup B = B$
- *Cota* $(\cap)$*:* $A \subseteq B$ *si, y solo si,* $A \cap B = A$

TEOREMA 2.1.27 *Si A y B son conjuntos dentro de un universo $\mathcal{U}$, entonces:*

- *Complemento:* $A \cup \overline{A} = \mathcal{U}; \quad A \cap \overline{A} = \varnothing$
- *De Morgan:* $\overline{A \cup B} = \overline{A} \cap \overline{B}; \quad \overline{A \cap B} = \overline{A} \cup \overline{B}$
- *Involución:* $\overline{\overline{A}} = A$

Las propiedades recogidas en los teoremas anteriores se denominan *propiedades básicas*. Junto con las definiciones, se podrán usar para demostrar cualquier otro las propiedad.

La propiedad asociativa de la unión y de la intersección permite prescindir de los paréntesis cuando aplicamos estos operadores a más de dos conjuntos

$$A \cup B \cup C = (A \cup B) \cup C = A \cup (B \cup C)$$
$$A \cap B \cap C = (A \cap B) \cap C = A \cap (B \cap C)$$

También podremos utilizar expresiones del tipo

$$\bigcup_{i=1}^{n} A_i, \qquad \bigcap_{i=1}^{n} A_i$$

para representar la unión y la intersección, respectivamente, de una familia de conjuntos.

EJEMPLO 2.1.28 Vamos a demostrar la igualdad

$$A \cap (B - C) = (A \cap B) - (A \cap C),$$

y para ello usaremos propiedades básicas:

$$\begin{aligned}
(A \cap B) - (A \cap C) &= (A \cap B) \cap \overline{(A \cap C)} = && \text{(Def. de } -) \\
&= (A \cap B) \cap (\overline{A} \cup \overline{C}) = && \text{(De Morgan)} \\
&= \Big((A \cap B) \cap \overline{A}\Big) \cup \Big((A \cap B) \cap \overline{C}\Big) = && \text{(Distribución)} \\
&= \varnothing \ \cup \ \Big((A \cap B) \cap \overline{C}\Big) = && \text{(Complementación)} \\
&= (A \cap B) \cap \overline{C} = && \text{(Identidad)} \\
&= A \cap (B \cap \overline{C}) = && \text{(Asociatividad)} \\
&= A \cap (B - C) && \text{(Def. de } -)
\end{aligned}$$

DEFINICIÓN 2.1.29 (AXIOMA DEL CONJUNTO DE LAS PARTES) *Si A es un conjunto, existe otro conjunto cuyos elementos son los subconjuntos de A:*

$$B \in \wp(A) \quad \overset{def.}{\Longleftrightarrow} \quad B \subseteq A$$

En la bibliografía también puede encontrarse la notación 2^A para el conjunto de las partes de A.

Ejemplo 2.1.30 Si $A = \{a, b\}$, entonces

$$\wp(A) = \Big\{\varnothing, \{a\}, \{b\}, \{a, b\}\Big\}$$ □

Ejemplo con Maxima 2.1.31 El operador de `Maxima` para calcular la el conjunto de las partes o conjunto potencia es `powerset`.

```
(%i1) powerset({0,1,2,3});
```

$$\{\{\{\}, \{0\}, \{0,1\}, \{0,1,2\}, \{0,1,2,3\}, \{0,1,3\}, \{0,2\}, \{0,2,3\}, \{0,3\}, \{1\}, \{1,2\}, \{1,2,3\}, \{1,3\}, \{2\}, \{2,3\}, \{3\}\}$$ □

2.1.2. Producto cartesiano, relaciones y funciones

Las *funciones* son, junto a los conjuntos, los elementos más básicos en los que se sustentan las matemáticas. Formalmente, las funciones son una clase particular de *relaciones*. A su vez, las *relaciones* son la herramienta matemática para formalizar las conexiones entre elementos de dos o más conjuntos. Funciones y relaciones son importantes en matemáticas y en computación tanto por sus aplicaciones teóricas cómo prácticas. El estudio de las relaciones es el objetivo del siguiente tema de la asignatura, en este, nos vamos a centrar en las funciones.

Definición 2.1.32 (Producto cartesiano de dos conjuntos) *El* producto cartesiano *de los conjuntos A y B es el conjunto:*

$$A \times B = \{(x, y);\ x \in A, y \in B\}$$

Los elementos del producto cartesiano de dos conjuntos se denominan *pares ordenados*. En ese caso, la palabra orden se refiere a la posición que ocupan los elementos dentro de la pareja. Por ejemplo, los pares $(1, 2)$ y $(2, 1)$ son elementos de $\mathbb{N} \times \mathbb{N}$, y aunque contienen las mismos elementos, son pares ordenados distintos: $(1, 2) \neq (2, 1)$

Si los conjuntos del producto son iguales, se puede utilizar una notación abreviada $A \times A = A^2$.

Definición 2.1.33 (Relación binaria) *Una* relación binaria *en los conjuntos A_1, A_2 es cualquier subconjunto $\mathcal{R}$ del producto cartesiano $A_1 \times A_2$*

$$\mathcal{R} \subseteq A_1 \times A_2$$

El símbolo utilizado para representar una relación binaria se suele escribir habitualmente de forma *infija*, es decir, si $(x, y) \in \mathcal{R}$, escribimos $x\mathcal{R}y$ y se lee, *x está relacionado con y mediante $\mathcal{R}$.*

DEFINICIÓN 2.1.34 *Si A y B son conjuntos, una* función de A en B *es una relación binaria f, de A en B tal que: para cada $x \in A$ existe un único $y \in B$ tal que $(x, y) \in f$.*

- *En este caso, escribimos $f\colon A \to B$*
- *Si $(x, y) \in f$ escribimos $f(x) = y$, es decir, representamos por $f(x)$ al único elemento que está relacionado con x mediante f.*
- *En la función $f\colon A \to B$, el conjunto A se denomina* dominio de f *y lo escribimos $A = \mathrm{Dom}(f)$. El conjunto B se denomina* codominio de f.
- *La imagen de la función $f\colon A \to B$ es un subconjunto de B definido como*

$$\mathrm{Im}(f) = \{y \in B \mid \text{ existe } x \in A \text{ tal que } y = f(x)\} = \{f(x); x \in A\}$$

Las funciones se denominan igualmente *aplicaciones*. Además de la notación $f\colon A \to B$ es habitual usar $A \xrightarrow{f} B$. También escribimos $x \overset{f}{\mapsto} y$ para indicar que $y = f(x)$.

Obsérvese que, cuando trabajamos con funciones, el dominio coincide con el conjunto A, es decir $\mathrm{Dom}(f) = A$. Si el dominio de la función es distinto de A, hablamos de *funciones parciales*, pero su estudio queda fuera de los objetivos del curso.

EJEMPLO 2.1.35

- La relación $\mathcal{R}_1 = \{(x, y) \in \mathbb{R} \times \mathbb{R} \mid x^2 + y^2 = 1\}$ no es una función, ya que $0\mathcal{R}_1 1$ y $0\mathcal{R}_1(-1)$ y además, el dominio de la relación no coincide con $\mathbb{R}$.
- La relación $\mathcal{R}_2 = \{(x, y) \in [-1, 1] \times \mathbb{R}^+ \mid x^2 + y^2 = 1\}$ sí es una función y $\mathcal{R}_2(x) = \sqrt{1 - x^2}$.
- La relación $\mathcal{R}_3 = \{(1, b), (2, a), (3, d)\} \subset \{1, 2, 3\} \times \{a, b, c, d\}$ es una función.

□

EJEMPLO 2.1.36

- Si $A \subseteq B$, la *función inclusión* es la función $i\colon A \to B$ definida por $i(x) = x$
- En particular, si $A = B$ la función inclusión se denomina *identidad de A* y se denota 1_A, es decir, $1_A\colon A \to A$ esta definida por $1_A(x) = x$ □

EJEMPLO 2.1.37 Si el dominio de la función es infinito, la asociación del elemento de la imagen correspondiente a cada elemento del dominio se hará mediante una regla.

- $f_1\colon \mathbb{Z} \to \mathbb{N}$, siendo $f_1(x) = x^2$
- $f_2\colon \mathbb{R} \to \mathbb{R}^+$, siendo $f_2(x) = 2^x$
- $f_3\colon \mathbb{R} \to \mathbb{R}$, siendo $f_3(x) = \text{sen}(x)$
- $f_4\colon (0,1) \subset \mathbb{R} \to \mathbb{R}$, siendo $f_4(x) = \dfrac{2-x}{x(1-x)}$
- $f_5\colon \mathbb{R} \to \mathbb{R}$, siendo $f_5(x) = \begin{cases} x & \text{si } \ x \geq 0 \\ -x & \text{si } \ x < 0 \end{cases}$ □

Ejemplo 2.1.38 Si el dominio es un conjunto finito, también podemos utilizar una tabla o una lista de emparejamientos para definir las funciones.

$$A = \{a_1, a_2, a_3, a_4, a_5\}, \qquad B = \{b_1, b_2, b_3\}$$

$$\begin{array}{l} f\colon A \to B \\ \quad a_1 \mapsto b_3 \\ \quad a_2 \mapsto b_1 \\ \quad a_3 \mapsto b_3 \\ \quad a_4 \mapsto b_2 \\ \quad a_5 \mapsto b_1 \end{array} \qquad \begin{array}{|c|ccccc|} \hline A & a_1 & a_2 & a_3 & a_4 & a_5 \\ \hline B & b_3 & b_1 & b_3 & b_2 & b_1 \\ \hline \end{array}$$ □

Ejemplo 2.1.39 Si el dominio de la función es $\mathbb{N}$, también podemos definir una función de forma *recursiva*. Por ejemplo, la sucesión de Fibonacci es una función $F\colon \mathbb{N} \to \mathbb{N}$ definida por:

$$\begin{aligned} F(0) &= 0 \\ F(1) &= 1 \\ F(n+1) &= F(n) + F(n-1) \end{aligned}$$

En este caso, para conocer la imagen de la función en un número, debemos calcular previamente las imágenes de todos los números menores.

$$F(0) = 0, \quad F(1) = 1, \quad F(2) = 1, \quad F(3) = 2, \quad F(4) = 3,$$

$$F(5) = 5, \quad F(6) = 8, \quad F(7) = 13, \quad F(8) = 21, \quad F(9) = 34, \ldots$$ □

Definición 2.1.40 *Sea $f\colon A \to B$ una función, $A_1 \subseteq A$ y $B_1 \subseteq B$*

- *Llamamos* imagen (directa) *de A_1 al conjunto:*

$$f(A_1) = \{y \in B \mid \textit{existe } \ x \in A_1 \textit{ tal que } f(x) = y\} = \{f(x); x \in A_1\}$$

Naturalmente, $f(A_1) \subseteq B$ y en particular $f(A) = \text{Im}(f)$.

- *Llamamos* preimagen *de* B_1 *al conjunto:*

$$f^{-1}(B_1) = \{a \in A \mid f(a) \in B_1\}$$

Naturalmente, $f^{-1}(B_1) \subseteq A$

Las operaciones de cálculo de la imagen y de la preimagen no son una inversa de la otra, tal y como vemos en los siguientes ejemplos.

EJEMPLO 2.1.41 Para $A = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ y $B = \{b_1, b_2, b_3, b_4, b_5\}$ definimos la función:

$$\begin{aligned} f\colon \quad & A \to B \\ & f(a_1) = b_2 \\ & f(a_2) = b_1 \\ & f(a_3) = b_3 \\ & f(a_4) = b_2 \\ & f(a_5) = b_3 \\ & f(a_6) = b_4 \end{aligned}$$

Si consideramos $A_1 = \{a_2, a_3, a_4\}$, tenemos que

$$\begin{aligned} f(A_1) &= f\big(\{a_2, a_3, a_4\}\big) = \{b_1, b_3, b_2\} \\ f^{-1}(f(A_1)) &= f^{-1}(\{b_1, b_3, b_2\}) = \{a_1, a_2, a_3, a_4, a_5\} \supset A_1 \end{aligned}$$ □

EJEMPLO 2.1.42 Para $A = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ y $B = \{b_1, b_2, b_3, b_4, b_5\}$, se define la función

$$\begin{aligned} f\colon \quad & A \to B \\ & f(a_1) = b_2 \\ & f(a_2) = b_1 \\ & f(a_3) = b_3 \\ & f(a_4) = b_2 \\ & f(a_5) = b_3 \\ & f(a_6) = b_4 \end{aligned}$$

Para el subconjunto $B_1 = \{b_3, b_4, b_5\}$, tenemos

$$\begin{aligned} f^{-1}(B_1) &= \{a_3, a_5, a_6\} \\ f(f^{-1}(B_1)) &= \{b_3, b_4\} \end{aligned}$$ □

DEFINICIÓN 2.1.43 *Dadas las funciones* $f\colon A \to B$ *y* $g\colon B \to C$, *definimos la composición* $g \circ f\colon A \to C$ *como* $g \circ f(x) = g(f(x))$ *para todo* $x \in A$.

EJEMPLO 2.1.44 La composición de funciones no es conmutativa. Por ejemplo, consideremos las funciones $f\colon \mathbb{R} \to \mathbb{R}$ y $g\colon \mathbb{R} \to \mathbb{R}$ tales que $f(x) = x + 1$ y $g(x) = x^2$. Las composiciones $f \circ g$ y $g \circ f$ son dos funciones de $\mathbb{R}$ en $\mathbb{R}$, pero las reglas que las definen son distintas

$$\begin{aligned} (f \circ g)(x) &= f(g(x)) = f(x^2) = x^2 + 1 \\ (g \circ f)(x) &= g(f(x)) = g(x+1) = (x+1)^2 = x^2 + 2x + 1 \end{aligned}$$ □

Ejemplo 2.1.45 Entre los conjuntos

$$A = \{a_1, a_2, a_3, a_4, a_5\}, \quad B = \{b_1, b_2, b_3, b_4\}, \quad C = \{c_1, c_2, c_3, c_4, c_5, c_6\}$$

definimos las funciones:

f

A	a_1	a_2	a_3	a_4	a_5
B	b_3	b_1	b_2	b_3	b_2

g

B	b_1	b_2	b_3	b_4
C	c_2	c_3	c_2	c_1

La composición $g \circ f$ será:

$g \circ f$

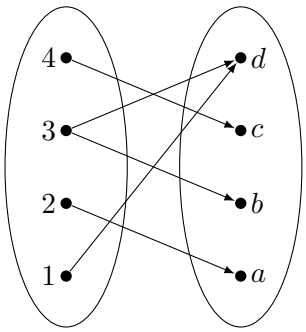
A	a_1	a_2	a_3	a_4	a_5
C	c_2	c_2	c_3	c_2	c_3

□

Ejemplo 2.1.46 Consideremos la relación $\mathcal{R} \subseteq \{1, 2, 3, 4\} \times \{a, b, c, d\}$:

$$\mathcal{R} = \{(1, d), (3, d), (3, b), (2, a), (4, c)\}$$

Representando los conjuntos con diagramas de Venn, los pares de la relación se pueden representar mediante flechas:



Sin embargo, la representación más habitual, por sus aplicaciones prácticas y computacionales es mediante tablas. La siguiente tabla muestra visualmente los elementos de la relación anterior:

	a	b	c	d
1				×
2	×			
3		×		×
4			×	

□

Matrices booleanas. Aunque los conjuntos son colecciones "no ordenadas" de elementos, cuando trabajamos con conjuntos finitos es conveniente representar sus elementos mediante secuencias ordenadas, de tal forma que ese orden permanezca fijo a lo largo del problema o de la aplicación. De esta forma, podremos definir procedimientos y algoritmos de forma más eficiente. Por ejemplo, para trabajar con subconjuntos de un conjunto finito, podemos aprovechar el orden para representarlos mediante secuencias de números 0 y 1.

Ejemplo 2.1.47 Para trabajar en el universo $\mathcal{U} = \{c, g, f, b, e, a, d\}$ vamos a utilizar el orden *lexicográfico*: $\mathcal{U} = [a, b, c, d, e, f, g]$. En la siguiente tabla, vemos cómo podemos representar mediante secuencias de dígitos 0-1, subconjuntos y operaciones entre ellos. Un 1 en la cadena significa que el elemento correspondiente está en el subconjunto, mientras que un 0 significa que no lo está:

		$abcdefg$
$\mathcal{U} = \{a, b, c, d, e, f, g\}$	1	1111111
$B = \{b, c, f, g\}$	x	0110011
$C = \{a, b, e, g\}$	y	1100101
$\overline{B} = \{a, d, e\}$	$1 - x$	1001100
$B \cap C = \{b, g\}$	mín(x, y)	0100001
$B \cup C = \{a, b, c, e, f, g\}$	máx(x, y)	1110111

Como vemos, la secuencia que representa la intersección de dos conjuntos queda determinada por la secuencia que se obtiene al calcular el mínimo de los dígitos posición a posición; la unión de dos conjuntos queda determinada por la secuencia que se obtiene al calcular el máximo de los dígitos posición a posición; y el complementario se obtiene intercambiando ceros y unos. □

Las operaciones que hemos utilizado en el ejemplo anterior, dentro del conjunto $\{0, 1\}$, se conocen como operaciones *booleanas* y se denominan *producto de Boole* $\wedge$, *suma de Boole* $\vee$, y *complemento de Boole* $\bar{\cdot}$:

$\wedge$	1	0
1	1	0
0	0	0

$\vee$	1	0
1	1	1
0	1	0

	1	0
$\bar{\cdot}$	0	1

$$x \wedge y = \text{mín}\{x, y\} \qquad x \vee y = \text{máx}\{x, y\} \qquad \overline{x} = 1 - x$$

La representación booleana de conjuntos finitos puede ser utilizada igualmente para representar relaciones entre conjuntos finitos. Sin embargo, en este caso, es preferible usar matrices de ceros y unos en lugar de secuencias, es decir, matrices *booleanas*. La *matriz de adyacencia* de una relación es la matriz booleana que la representa.

Definición 2.1.48 (Matriz de adyacencia) *Sea $\mathcal{R}$ es una relación binaria entre dos conjuntos finitos A y B. Supongamos que $[x_1, x_2, \ldots, x_n]$ es una ordenación de los elementos de A e $[y_1, y_2, \ldots, y_m]$ una ordenación de los elementos de B. Llamamos* matriz (de adyacencia) asociada *a $\mathcal{R}$ a la matriz $\mathcal{M}_\mathcal{R} = (m_{ij})$ de tamaño $n \times m$ (es decir, con n filas y m columnas) dada por:*

$$m_{ij} = \begin{cases} 1 & si\ (x_i, y_j) \in \mathcal{R} \\ 0 & si\ (x_i, y_j) \notin \mathcal{R} \end{cases}$$

Ejemplo 2.1.49 Consideremos la relación $\mathcal{R} \subseteq \{1,2,3,4\} \times \{a,b,c,d\}$ definida por la siguiente tabla:

	a	b	c	d
1				×
2	×			
3		×		×
4			×	

La matriz de adyacencia construida utilizando el orden en el que aparecen escritos los elementos en la tabla de arriba es:

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

□

2.1.3. Tipos de funciones

Definición 2.1.50 *Sea $f\colon A \to B$ una función.*

- *f se dice* inyectiva *si elementos distintos tienen imágenes distintas:*

$$f(x) = f(y) \quad \Longrightarrow \quad x = y$$

- *f se dice* sobreyectiva *si todo elemento del codominio tiene preimagen:*

$$\textit{para todo } y \in B, \ \textit{existe } x \in A \textit{ tal que } f(x) = y$$

- *f se dice* biyectiva *si es inyectiva y sobreyectiva.*

Ejemplo 2.1.51

1. Las inclusiones $i\colon A \to B$ son funciones inyectivas.
2. Las identidades 1_A son funciones biyectivas.
3. La función $f\colon \mathbb{N} \to \mathbb{N}$, definida por $f(x) = 2x$ es inyectiva:

$$f(x) = f(y) \quad \Rightarrow \quad 2x = 2y \quad \Rightarrow \quad x = y$$

4. La función $g\colon \mathbb{R} \to [-1,1]$, definida $g(x) = \operatorname{sen} x$, es sobreyectiva, pero no es inyectiva.
5. La función $h\colon [-\pi/2, \pi/2] \to [-1,1]$, definida $h(x) = \operatorname{sen} x$, es biyectiva. □

TEOREMA 2.1.52 *Consideremos las funciones $f\colon A \to B$ y $g\colon B \to C$.*

1. *Si f y g son inyectivas, entonces $g \circ f$ también es inyectiva.*
2. *Si $g \circ f$ es inyectiva, entonces f es inyectiva.*
3. *Si f y g son sobreyectivas, entonces $g \circ f$ también es sobreyectiva.*
4. *Si $g \circ f$ es sobreyectiva, entonces g es sobreyectiva*
5. *Si f y g son biyectivas, entonces $g \circ f$ también es biyectiva.*
6. *Si $g \circ f$ es biyectiva, entonces f es inyectiva y g es sobreyectiva.*

EJEMPLO 2.1.53 Consideremos las funciones $f\colon A \to B$ y $g\colon B \to C$; como ejemplo de demostración, vamos a probar que si $g \circ f$ es sobreyectiva, entonces g es sobreyectiva.

Suponemos entonces que $g \circ f$ es sobreyectiva y queremos probar g es sobreyectiva, y para ello, tenemos que encontrar una preimagen de cada elemento de C. Si $y \in C$, y dado que $g \circ f$ es sobreyectiva, existe $x \in A$ tal que $(g \circ f)(x) = y$; es decir, $g(f(x)) = y$. Esto termina la demostración, ya que hemos encontrado $f(x) \in B$ que es preimagen de y por g. □

DEFINICIÓN 2.1.54 *Dada la función $f\colon A \to B$, se dice que $g\colon B \to A$ es* inversa *de f si $g \circ f = 1_A \quad y \quad f \circ g = 1_B$. Si existe una función inversa de f, decimos que f es* inversible.

TEOREMA 2.1.55 *Si una función es inversible, entonces solo tiene una función inversa: denotaremos por f^{-1} a la función inversa de f.*

Hemos utilizado la notación f^{-1} para dos operaciones distintas, el contexto resolverá la ambigüedad, ya que en un caso aplicamos el operador a un conjunto (preimagen) y en el otro a un elemento (función inversa). Además, los dos operadores están relacionados, ya que si f es inversible, $\{f^{-1}(y)\} = f^{-1}(\{y\})$.

EJEMPLO 2.1.56 La función $f\colon \mathbb{Z} \to \mathbb{Z}$ definida por $f(x) = x + 7$ es inversible:

$$x + 7 = y \iff y - 7 = x$$

Por lo tanto, $f^{-1}(y) = y - 7$, lo que comprobamos a continuación:

$$(f^{-1} \circ f)(x) = f^{-1}(x + 7) = (x + 7) - 7 = x = 1_{\mathbb{Z}}(x)$$
$$(f \circ f^{-1})(y) = f(y - 7) = (y - 7) + 7 = y = 1_{\mathbb{Z}}(y)$$ □

TEOREMA 2.1.57 *Una función $f\colon A \to B$ es inversible si y sólo si es biyectiva.*

Ejemplo 2.1.58 Vamos a demostrar en este ejemplo el teorema anterior.

- ($\Leftarrow$) Supongamos que $f\colon A \to B$ es biyectiva. Entonces, para cada $y \in B$ existe exactamente un $x \in A$ tal que $f(x) = y$; llamemos $g(y)$ al elemento de A así definido: $x = g(y)$. Entonces, $f(g(y)) = f(x) = y$.

  Para demostrar la otra igualdad, basta observar que dado que f es una inyectiva, para $f(x) \in B$, x es el único elemento de A con imagen $f(x)$, por lo que $g(f(x)) = x$.

- ($\Rightarrow$) Supongamos que f tiene inversa f^{-1}. Entonces, para cada $y \in B$, se verifica que $f(f^{-1}(y)) = y$, por lo que $f^{-1}(y)$ es preimagen de y y podemos afirmar que f es sobreyectiva.

  Para demostrar que f es inyectiva, supongamos que $f(x) = f(y)$. Entonces:

$$\begin{aligned} f(x) &= f(y) \\ f^{-1}(f(x)) &= f^{-1}(f(y)) \\ x &= y \end{aligned}$$

  y en consecuencia podemos afirmar que f es inyectiva. □

El principio de Dirichlet que enunciamos a continuación debe su nombre a Peter Gustav Lejeune Dirichlet (1805-1859), que lo enunció por primera vez de manera formal. También se conoce en la literatura como *principio de distribución de Dirichlet*, *principio de las cajas de Dirichlet* o *principio del palomar*.

Teorema 2.1.59 (Principio de Dirichlet) *Sean A y B conjuntos finitos tales que $|A| > |B|$. Entonces no es posible definir una función inyectiva de A en B.*

En la descripción informal, los elementos de B son los nidos y los elementos de A son las palomas que quieren entrar en los nidos. Dado que hay más palomas que nidos, necesariamente en algún nido debe de haber más de una paloma.

Ejemplo 2.1.60 Vamos a demostrar que cualquier subconjunto de seis elementos de $D = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ debe contener dos elementos cuya suma es 10.

Los pares de elementos de D que suman 10 son:

$$\{1, 9\}, \{2, 8\}, \{3, 7\} \text{ y } \{4, 6\}$$

Estos subconjuntos constituyen los nidos y las palomas son los 6 números del subconjunto analizado. Cada uno de los seis números lo tendremos que coger de algún nido y dado que solo hay 4 nidos, necesariamente tendremos que coger dos de un mismo nido; esos dos números elegidos del mismo nido suman 10. □

Ejemplo 2.1.61

- En cualquier grupo de 13 personas podemos encontrar dos que han nacido en el mismo mes.
- Un cuestionario consta de 10 preguntas de seleción múltiple, con cinco alternativas cada una. Entonces, el mínimo número de alumnos para el cual podamos garantizar que, por lo menos, dos de ellos tendrán exactamente las mismas respuestas para todas las preguntas es $5^{10} + 1 = 9765626$. □

Teorema 2.1.62 (Principio de Dirichlet generalizado) *Sean A y B conjuntos finitos tales que $|A| = r$, $|B| = n$ y $r > k \cdot n$ para algún k. Entonces cada función $f\colon A \to B$, verifica que la preimagen de algún elemento b de B tiene más de k elementos:*

$$|f^{-1}(b)| > k$$

Por ejemplo, si queremos repartir r objetos distintos en n cajas distintas, siendo $r > n \cdot k$ para algún k, entonces necesariamente tendremos que poner más de k objetos en alguna de ellas. Esto es lo que se conoce como el *principio de las cajas generalizado.*

Ejemplo 2.1.63

- En un grupo de 22 personas hay al menos cuatro que han nacido el mismo día de la semana.
- En un grupo de 25 personas hay al menos tres que han nacido el mismo mes.
- Necesitaremos tirar a lo sumo siete veces un dado para garantizar que un resultado se repite al menos dos veces.
- Necesitaremos tirar a lo sumo 13 veces un dado para garantizar que un resultado se repite al menos tres veces.
- En general, necesitaremos tirar $6(n-1)+1$ veces un dado para garantizar que un resultado se repite al menos n veces.
- Si elegimos $n+1$ números del conjunto $\{1, 2, 3, \ldots, 2n\}$, necesariamente dos de ellos serán coprimos. Para demostrar esta afirmación, tomamos los conjuntos $\{1, 2\}$, $\{3, 4\}$, $\{5, 6\}$, $\ldots \{2n-1, 2n\}$ como 'nidos'; dado que solo hay n 'nidos' y queremos coger $n+1$ números, necesariamente tendremos que coger dos del mismo nido; como cada nido esta formado por un par de números consecutivos, estos son coprimos, lo que demuestra la afirmación. □

2.2. Recuento

En matemáticas y en computación, es frecuente la necesidad de conocer la cantidad de elementos u objetos de un determinado dominio. Para comparar el coste de aplicar dos algoritmos, debemos determinar el 'número de operaciones' que ejecuta cada uno. Para estimar el coste de utilizar una determinada 'estructura de datos', debemos conocer cuántos ítemes pueden ser almacenados en dicha estructura de datos. Para establecer la probabilidad de un acontecimiento, necesitamos conocer la cantidad de posibilidades.

Si la cantidad de elementos que queremos determinar es pequeña, puede ser suficiente el recorrido uno a uno de estos elementos para determinar el tamaño; pero este método no es posible si la cantidad es muy grande. La teoría que desarrollamos en este tema consiste en describir diferentes 'modelos' teóricos que describen conjuntos cuyo número de elementos sabemos determinar. Para aplicar lo que se denomina teoría de recuento, debemos buscar el modelo que se ajuste al conjunto cuyo tamaño queremos determinar; aunque debemos tener en cuenta que un mismo problema se podrá abordar con varios modelos.

Ejemplo 2.2.1 ¿Cuántos partidos se necesita programar para determinar el campeón de un torneo de tenis en el que hay 64 participantes?

- **Análisis 1:** Necesitamos 32 partidos para reducir los jugadores a la mitad; a continuación, necesitaremos 16 partidos para reducirlos de nuevo a la mitad; siguiendo este razonamiento, deducimos que el número de partidos será:

$$32 + 16 + 8 + 4 + 2 + 1 = 63$$

- **Análisis 2:** Cada partido elimina exactamente un jugador, y dado que necesitamos eliminar 63 jugadores, necesitaremos programar 63 partidos.

Aunque los dos análisis son válidos, el segundo es más simple y conlleva un proceso de cálculo más sencillo. En el primer análisis hemos pensado en los 'ganadores', mientras que en el segundo hemos pensado en los 'perdedores'. En términos conjuntistas, en el segundo análisis hemos hallado el tamaño del conjunto complementario. □

En términos matemáticos, lo que vamos a hacer en este tema es describir lo que queremos contar como un conjunto y determinar el número de elementos de ese conjunto: si A es un conjunto finito, $|A| \in \mathbb{N}$ denota el número de elementos del conjunto A.

2.2.1. Operaciones entre conjuntos

Dado que estamos interesados en conocer el tamaño de un conjunto, los modelos que estudiamos determinan el comportamiento del tamaño de un conjunto respecto a las distintas operaciones que podemos realizar entre conjuntos.

Empezamos estableciendo la relación del tamaño de un conjunto respecto de las operaciones entre conjuntos: unión y producto cartesiano.

TEOREMA 2.2.2 (REGLA DE LA SUMA) *Si A y B son conjuntos finitos disjuntos, entonces $A \cup B$ es finito y*

$$|A \cup B| = |A| + |B|$$

COROLARIO 2.2.3 *Si $A_1, \ldots, A_k$ son conjuntos finitos y disjuntos dos a dos, entonces*

$$\left| \bigcup_{i=1}^{k} A_i \right| = |A_1| + \cdots + |A_k|$$

La regla o el principio de la suma se traduce en la técnica de recuento **por casos**: si un procedimiento se puede separar en n casos mutuamente excluyentes y hay s_i posibles resultados para el i-ésimo caso, entonces el número total de resultados es

$$s_1 + \cdots + s_t$$

Este procedimiento es el usamos en el primer análisis del primer ejemplo de esta sección. Pero también es el que hemos utilizado en el segundo análisis, ya que A y $\mathcal{U} - A$ son conjuntos disjuntos.

COROLARIO 2.2.4 *Si A es un conjunto finito, entonces*

$$B \subseteq A \quad \Longrightarrow \quad |A - B| = |A| - |B|$$

TEOREMA 2.2.5 (REGLA DEL PRODUCTO) *Si A y B son conjuntos finitos, entonces:*

$$|A \times B| = |A| \cdot |B|$$

COROLARIO 2.2.6 *Si $A_1, \ldots, A_k$ son conjuntos finitos, entonces:*

$$|A_1 \times \cdots \times A_k| = |A_1| \cdot \ldots \cdot |A_k|$$

La regla o el principio del producto se traduce en la técnica de recuento **secuencial**: si una actividad se puede realizar en n pasos sucesivos de forma que el paso 1 se realiza de r_1 formas, el segundo paso se realiza de r_2 formas y sucesivamente el último paso se realiza de r_n formas, entonces el número total de formas posibles de realizar esa actividad es

$$r_1 \cdot r_2 \cdot \ldots \cdot r_t$$

Al aplicar este modelo, debemos tener cuidado y asegurarnos de que cada secuencia nos da un resultado diferente, ya que en caso contrario estaríamos contando un mismo objeto o evento varias veces.

Ejemplo 2.2.7

1. El número de formas en que se puede responder a un cuestionario de diez preguntas del tipo 'Verdadero' o 'Falso' es 2^{10}.

2. El número de enteros de cuatro cifras que no tienen cifras repetidas es $9 \cdot 9 \cdot 8 \cdot 7$: dado que no puede empezar por 0, solo tenemos 9 dígitos posibles en las unidades de millar; para las centenas también tenemos 9 dígitos posibles, ya que no podemos utilizar el que hemos puesto la unidad de millar; para las decenas disponemos de 8 dígitos, ya que tenemos que excluir los usados en la unidad de millar y en la centena y finalmente para las unidades solo disponemos de 7 dígitos.

Ejemplo 2.2.8 Si A es un conjunto finito, denotamos por $|A| \in \mathbb{N}$ el número de elementos que contiene. En ese caso, el conjunto $\wp(A)$ también es finito y

$$|\wp(A)| = 2^{|A|}$$

Si disponemos los elementos de A en una lista, podemos identificar los subconjuntos de A con una lista binaria (de ceros y unos) de la misma longitud en donde los unos indican que el elemento de A en la misma posición pertenece al subconjunto y los ceros indican que el elemento no pertenece. De esta forma, contar los subconjuntos de A es contar las listas binarias de longitud $|A| = n$. Dado que en cada posición podemos colocar dos posibles elementos, 0 o 1, el recuento por casos nos dice que el número total de listas es

$$\underbrace{2 \cdot 2 \cdots 2}_{n} = 2^n \qquad \square$$

2.2.2. Principio de Inclusión-Exclusión

Solo podemos aplicar la regla de la suma si los conjuntos son disjuntos dos a dos, el principio de inclusión-exclusión generaliza este resultado si no se verifica esa restricción.

Proposición 2.2.9 *Si A_1 y A_2 son conjuntos finitos (no necesariamente disjuntos), entonces*

$$|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2|$$

Este resultado es consecuencia de la regla de la suma, teniendo en cuenta que $A_1 - A_2$, $A_2 - A_1$ y $A_1 \cap A_2$ son conjuntos disjuntos dos a dos y

$$A_1 = (A_1 - A_2) \cup (A_1 \cap A_2)$$
$$A_2 = (A_2 - A_1) \cup (A_1 \cap A_2)$$
$$A_1 \cup A_2 = (A_1 - A_2) \cup (A_1 \cap A_2) \cup (A_2 - A_1)$$

Por lo tanto:

$$\begin{aligned}|A_1 \cup A_2| &= |A_1 - A_2| + |A_1 \cap A_2| + |A_2 - A_1| = \\ &= |A_1 - A_2| + |A_1 \cap A_2| + |A_2 - A_1| + |A_1 \cap A_2| - |A_1 \cap A_2| = \\ &= |(A_1 - A_2) \cup (A_1 \cap A_2)| + |(A_2 - A_1) \cup (A_1 \cap A_2)| - |A_1 \cap A_2| = \\ &= |A_1| + |A_2| - |A_1 \cap A_2|\end{aligned}$$

La asociatividad de la unión de conjuntos nos permite extender fácilmente esta propiedad a la unión de tres o más conjuntos.

COROLARIO 2.2.10 *Si A_1, A_2 y A_3 son conjuntos finitos, entonces*

$$|A_1 \cup A_2 \cup A_3| = |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - |A_1 \cap A_3| - |A_2 \cap A_3| + |A_1 \cap A_2 \cap A_3|$$

Intuitivamente, al sumar los elementos de todos los conjuntos, estamos contando dos veces los que están en la intersección de dos conjuntos, y por eso restamos los tamaños de las intersecciones dos a dos. Pero al hacer esto, estamos restando dos veces los elementos que están en la intersección de los tres conjuntos, y por eso debemos sumar el tamaño de la intersección de los tres conjuntos.

Formalmente, la demostración de esta igualdad hace uso solamente del principio de inclusión-exclusión para dos conjuntos y de la distributividad de la intersección respecto de la unión:

$$\begin{aligned}|A_1 \cup A_2 \cup A_3| &= |A_1| + |A_2 \cup A_3| - |A_1 \cap (A_2 \cup A_3)| = \\ &= |A_1| + |A_2| + |A_3| - |A_2 \cap A_3| - |(A_1 \cap A_2) \cup (A_1 \cap A_3)| = \\ &= |A_1| + |A_2| + |A_3| - |A_2 \cap A_3| - |A_1 \cap A_2| - |A_1 \cap A_3| + |A_1 \cap A_2 \cap A_1 \cap A_3| = \\ &= |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - |A_1 \cap A_3| - |A_2 \cap A_3| + |A_1 \cap A_2 \cap A_3|\end{aligned}$$

Enunciamos finalmente el principio de inclusión-exclusión para una familia finita de conjuntos finitos. No incluimos la demostración formal, pero el razonamiento intuitivo es el mismo que hemos utilizado para una familia de tres conjuntos.

TEOREMA 2.2.11 (PRINCIPIO DE INCLUSIÓN-EXCLUSIÓN) *Sean $A_1, A_2, \ldots, A_k$ conjuntos finitos no vacíos. Entonces*

$$\Big|\bigcup_{j=1}^{k} A_j\Big| = \sum_{j=1}^{k} (-1)^{j+1} \sum_{1 \le i_1 < \ldots < i_j \le k} \Big|A_{i_1} \cap \cdots \cap A_{i_j}\Big|$$

En la expresión de este teorema, el sumatorio $\sum_{1\leq i_1<\ldots<i_j\leq k}$ para cada $j \in \{1,\ldots,k\}$, recorre todas las intersecciones de, exactamente, j conjuntos.

En el siguiente ejemplo, vamos a utilizar la función *parte entera* o *suelo*, que se denota por $\lfloor\cdot\rfloor$ y se define como sigue: para cada $x \in \mathbb{R}$, $\lfloor x\rfloor$ es el mayor entero menor o igual que x; por ejemplo, $\lfloor\frac{10}{3}\rfloor = 3$, $\lfloor\sqrt{5}\rfloor = 2$.

Ejemplo 2.2.12 ¿Cuántos enteros positivos menores o iguales a 1000 son divisibles por 7 o por 11?:

Si consideramos el conjunto A_1 de los múltiplos de 7 menores que 1000 y el conjunto A_2 de los múltiplos de 11, entonces queremos saber el tamaño de $A_1 \cup A_2$. En general, el número de enteros positivos divisibles por d y menores que 1000 es $\lfloor\frac{1000}{d}\rfloor$, en donde $\lfloor\cdot\rfloor$ es la función parte entera, es decir, $\lfloor x\rfloor$ es el mayor entero menor o igual que x. Entonces:

$$|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2| = \\ = \lfloor\frac{1000}{7}\rfloor + \lfloor\frac{1000}{11}\rfloor - \lfloor\frac{1000}{7\cdot 11}\rfloor = 142 + 90 - 12 = 220$$

Hemos utilizado que la intersección $A_1 \cap A_2$ está formada por los números que son divisibles por 7 y por 11, es decir, que son divisibles por 77. □

No hemos establecido hasta ahora un resultado que nos permita hallar el tamaño de la intersección. Para ello, basta tener en cuenta que la ley de De Morgan nos permite expresar la intersección de conjuntos en función de la unión y los conjuntos complementarios. Si $A_1 \subseteq \mathcal{U}, \ldots, A_n \subseteq \mathcal{U}$, entonces

$$A_1 \cap \cdots \cap A_n = \overline{\overline{A}_1 \cup \cdots \cup \overline{A}_n}$$
$$|A_1 \cap \cdots \cap A_n| = |\mathcal{U}| - |\overline{A}_1 \cup \cdots \cup \overline{A}_n|$$

2.2.3. Conjuntos de funciones

Los conjuntos de funciones entre dos conjuntos dan diferentes modelos de recuento.

Proposición 2.2.13 *Si A y B son conjuntos finitos, entonces el número de funciones definibles de A en B es $|B|^{|A|}$.*

Este resultado es una consecuencia del principio del producto. Una función de A en B se define asignando exactamente un elemento de B a cada elemento de A. Por lo tanto, para cada elemento de A tenemos exactamente $|B|$ posibilidades, lo que nos da un total de

$$\underbrace{|B|\ldots|B|}_{(|A|)} = |B|^{|A|}$$

Un caso particular lo obtenemos al calcular el número de subconjuntos de un conjunto. Cada subconjunto A se determina a partir de una aplicación de A en $\{0,1\}$, por lo que el número de subconjuntos de A es $2^{|A|}$.

COROLARIO 2.2.14 *Si A es un conjunto finito, entonces hay $2^{|A|}$ subconjuntos de A:*

$$|\wp(A)| = 2^{|A|}$$
$$|2^A| = 2^{|A|}$$

Por el principio de Dirichlet, para definir funciones inyectivas entre dos conjuntos A y B necesitamos que $|A| \leq |B|$, pero, en tal caso ¿cuántas funciones distintas podemos definir? Para determinar una aplicación inyectiva vamos asignando imágenes de forma secuencial a cada elemento de A, de forma que en cada paso podemos elegir entre un elemento menos; de esta forma deducimos el siguiente resultado.

PROPOSICIÓN 2.2.15 *Sean A y B conjuntos finitos tales que, si $r = |A|$, $n = |B|$, $r \leq n$. Entonces, el número de funciones inyectivas de A en B es*

$$n(n-1)\dots(n-r+1) = \frac{n!}{(n-r)!}$$

2.2.4. Permutaciones

Si A es un conjunto de n objetos distintos y $r \leq n$, una **r-permutación** (o **variación** de n elementos tomados de r en r) de A es una secuencia de r elementos de A. En las permutaciones "no podemos repetir" elementos y hablamos de secuencias porque el orden en que disponemos los elementos determina permutaciones distintas. Por ejemplo, *acd*, *cda*, *adc*, *bca*,..., son 3-permutaciones distintas de elementos de $\{a,b,c,d\}$.

El número de r-permutaciones de un conjunto A de n elementos se denota $P(n,r)$ y coincide con el número de funciones inyectivas del conjunto $\{1,2,\dots,r\}$ en el conjunto A, es decir,

$$P(n,r) = n(n-1)(n-2)\dots(n-r+1) = \frac{n!}{(n-r)!}$$

Las n-permutaciones de un conjunto de n elementos se denominan simplemente permutaciones y en este caso, $P(n,n) = n!$

EJEMPLO 2.2.16 ¿Cuántos números comprendidos entre 1 y 9999 tienen cifras distintas?

En primer lugar, dividimos el problema en cuatro casos, es decir, dividimos el conjunto en cuatro subconjuntos disjuntos:

X_1: Enteros de una cifra sin cifras repetidas.

X_2: Enteros de dos cifras sin cifras repetidas.

X_3: Enteros de tres cifras sin cifras repetidas.

X_4: Enteros de cuatro cifras sin cifras repetidas.

Los elementos de cada subconjunto son permutaciones (los dígitos deben ser distintos y órdenes distintos dan números distintos), por lo que

$$|X_1| + |X_2| + |X_3| + |X_4| = 9 + 9 \cdot 9 + 9 \cdot 9 \cdot 8 + 9 \cdot 9 \cdot 8 \cdot 7 \qquad \square$$

EJEMPLO CON MAXIMA 2.2.17 En `Maxima`, disponemos de algunos operadores relacionados con los modelos de recuento que vamos a estudiar en este tema. Por ejemplo, el operador `permutations` permite obtener el conjunto de todas las permutaciones de los elementos de un conjunto.

```
(%i1) perm3: permutations({1,2,3});
```

$$\{[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]\}$$

Naturalmente, el resultado es un conjunto de listas. En este caso, estamos obteniendo permutaciones de todos los elementos del conjunto y por lo tanto, el número total coincide con el factorial del cardinal del conjunto.

```
(%i2) is(cardinality(perm3)=3!);
```

```
True
```

Utilizando los operadores para trabajar con conjuntos que hemos aprendido, también podemos obtener variaciones. Por ejemplo, las 2-permutaciones de $A = \{1,2,3\}$ son los elementos de $A \times A$ con las dos componentes distintas y por lo tanto, podemos obtener este conjunto como subconjunto del producto cartesiano.

```
(%i3) distintos(l):= is(l[1]#l[2])$
(%i4) distintos([1,1]);
```

```
False
```

```
(%i5) distintos([2,1]);
```

```
True
```

```
(%i6) perm23: subset(cartesian_product({1,2,3},{1,2,3}),distintos);
```

$$\{[1,2],[1,3],[2,1],[2,3],[3,1],[3,2]\}$$

El cardinal de este conjunto es igual a $3 \cdot 2 = 6$:

```
(%i7) cardinality(perm23);
```

6 $\square$

2.2.5. Permutaciones generalizadas

Las permutaciones de n elementos distintos son reordenaciones de estos elementos, pero ¿qué ocurre si algunos de estos objetos son indistinguibles? ¿de cuántas formas podremos reordenarlos?

Consideremos una 'colección' de n objetos de k tipos diferentes, de forma que los objetos de cada tipo son indistinguibles. Una *permutación generalizada* es cualquier secuencia ordenada de los n objetos. Si n_i es el número de objetos del tipo i para cada $i = 1 \dots k$, y $n = n_1 + \cdots + n_k$, denotamos por $P(n; n_1, \dots, n_k)$ al número de permutaciones generalizadas.

TEOREMA 2.2.18 $P(n; n_1, \dots, n_k) = \dfrac{n!}{n_1! \cdot n_2! \cdots n_k!}$

EJEMPLO 2.2.19 ¿Cuantas palabras (con o sin significado) se pueden obtener a partir de las letras de la palabra IMPRIMIR?

Es decir, tenemos que hallar de cuantas formas podemos reordenar 3 letras I, 2 letras M, 1 letra P y 2 letras R. Según el teorema anterior, podemos formar

$$P(8; 3, 2, 1, 2) = \frac{8!}{3! \cdot 2! \cdot 1! \cdot 2!} = \frac{8 \cdot 7 \cdot 6 \cdot 5 \cdot \cancel{4} \cdot \cancel{3!}}{\cancel{3!} \cdot \cancel{2} \cdot \cancel{2}} = 1680$$

palabras distintas. □

EJEMPLO CON MAXIMA 2.2.20 En las permutaciones generalizas puede haber elementos repetidos, lo que significa que estamos permutando elementos de una lista. De hecho, el operador `permutations` de `Maxima` se puede aplicar a una lista y en ese caso, nos genera las permutaciones generalizadas.

```
(%i1) perm222: permutations([1,1,2,2,3,3]);
```

$$\{[1,1,2,2,3,3],[1,1,2,3,2,3],\dots,[3,3,2,1,2,1],[3,3,2,2,1,1]\}$$

La expresión que determina el cardinal de este conjunto se denomina coeficiente multinomial y esa denominación es la que da nombre al operador en `Maxima`.

$$P(n; n_1, n_2, ..., n_r) = \texttt{multinomial\_coeff}(n_1, n_2, ..., n_r) = \frac{n!}{n_1! \cdot n_2! \cdot \cdots \cdot n_r!}$$

```
(%i2) cardinality(perm222)=multinomial_coeff(2,2,2)
                                        =6!/(2!*2!*2!);
```

$$90 = 90 = 90$$ □

2.2.6. Combinaciones

Anteriormente hemos calculado el número de subconjuntos de un conjunto, ahora vamos a determinar el número de subconjuntos de un tamaño fijo, es lo que se denominan *combinaciones*. Concretamente, si X es un conjunto de n elementos distintos y $r \leq n$, llamamos **r-combinación** a cualquier subconjunto de r elementos de X. Hablamos de subconjuntos porque los elementos no pueden repetirse y porque no importa el orden en el que dispongamos los elementos de la combinación (a diferencia de las permutaciones, en las que el orden sí importa). El número de r-combinaciones de n elementos se denota por $C(n,r)$ y

$$C(n,r) = \frac{P(n,r)}{r!} = \frac{n!}{r!(n-r)!} = \binom{n}{r}$$

La demostración es bastante simple. Hay $P(n,r)$ formas de disponer de manera ordenada r elementos del conjunto X; dado que no nos importa el orden, debemos considerar iguales las $r!$ formas de reordenar cada elección, por lo que el total de combinaciones es $\dfrac{P(n,r)}{r!}$.

Ejemplo con Maxima 2.2.21 Ya habíamos visto que `binomial` es el operador de `Maxima` que determina los número combinatorios: `binomial`$(n,m) = \binom{n}{m}$. Pero, ¿cómo determinamos los subconjuntos de un conjunto con un determinado número de elementos? Nuevamente, podemos recurrir a los operadores que hemos aprendido hasta ahora y, concretamente, a la forma de determinar subconjuntos definidos por una propiedad. Si queremos obtener los subconjuntos con 3 elementos, podemos usar la siguiente propiedad:

```
(%i1) card3(x):=is(cardinality(x)=3)$
```

De esta forma, los subconjuntos con 3 elementos del conjunto $\{1,2,3,4,5,6\}$ se pueden determinar como sigue:

```
(%i2) subconj63: subset(powerset({1,2,3,4,5,6}),card3);
```

$$\{\{1,2,3\},\{1,2,4\},\{1,2,5\},\{1,2,6\},\{1,3,4\},\{1,3,5\},\{1,3,6\},\{1,4,5\},\{1,4,6\},\{1,5,6\},\{2,3,4\},\{2,3,5\},\{2,3,6\},\{2,4,5\},\{2,4,6\},\{2,5,6\},\{3,4,5\},\{3,4,6\},\{3,5,6\},\{4,5,6\}\}$$

Naturalmente, el cardinal de este conjunto coincide con $\binom{6}{3}$:

```
(%i3) cardinality(subconj63)=binomial(6,3);
```

$$20 = 20$$ □

2.2.7. Funciones sobreyectivas

Con los resultados anteriores hemos aprendido a determinar el número total de aplicaciones entre dos conjuntos y el número de ellas que son inyectivas (y como caso particular las biyectivas). En el siguiente resultado, determinamos el número de aplicaciones sobreyectivas entre dos conjuntos $|A|$ y $|B|$ tales que $|A| \geq |B|$; esta restricción también es consecuencia del principio de Dirichlet.

COROLARIO 2.2.22 (FUNCIONES SOBREYECTIVAS) *Si $|A| = r$ y $|B| = n$ y $n \leq r$, entonces el número de funciones sobreyectivas de A en B es*

$$n^r - \binom{n}{1} \cdot (n-1)^r + \binom{n}{2} \cdot (n-2)^r - \cdots + (-1)^{n-1} \binom{n}{n-1} \cdot 1^r = \\ = n^r - \sum_{j=1}^{n-1} (-1)^{j+1} \binom{n}{j} \cdot (n-j)^r$$

Este resultado es una consecuencia del principio de inclusión-exclusión. Para cada $b \in B$, definimos el conjunto F_b:

$$F_b = \{f \colon A \to B \mid f(x) \neq b, \text{ para todo } x \in A\}$$

Es decir, F_b está formado por las funciones de A en B que dejan al elemento b sin preimagen. Por lo tanto, el conjunto de las aplicaciones sobreyectivas coincide con

$$\overline{\bigcup_{b \in B} F_b}$$

y por complementariedad y el principio de inclusión-exclusión, el número de elementos de este conjunto es

$$\left| \overline{\bigcup_{b \in B} F_b} \right| = n^r - \left| \bigcup_{b \in B} F_b \right| = n^r - \sum_{j=1}^{n-1} (-1)^{j+1} \sum_{\substack{J \subseteq B \\ |J|=j}} \left| \bigcap_{b \in J} F_b \right|$$

En esta expresión, para cada $j \in \{1, \ldots, n-1\}$, el sumando $\displaystyle\sum_{\substack{J \subseteq B \\ |J|=j}} \left| \bigcap_{b \in J} F_b \right|$ coincide con el número de funciones que dejan al menos a j elementos sin preimagen. Es decir, para cada elección de j elementos, el número de funciones de un conjunto de $n-j$ elementos en otro conjunto de r elementos:

$$\sum_{\substack{J \subseteq B \\ |J|=j}} \left| \bigcap_{b \in J} F_b \right| = \binom{n}{j} \cdot (n-j)^r,$$

de donde se deduce la expresión que aparece en el teorema.

La expresión que determina el número de aplicaciones sobreyectivas se puede expresar con los *números de Stirling de Segunda Clase* que se pueden calcular fácilmente de forma recursiva.

DEFINICIÓN 2.2.23 *Se denominan* números de Stirling de Segunda Clase*, y se denotan, $S(r,n)$ a los números de la forma*

$$S(r,n) = \frac{1}{n!}\left(n^r - \sum_{j=1}^{n-1}(-1)^{j+1}\binom{n}{j}\cdot(n-j)^r\right) =$$
$$= \frac{1}{n!}\left(n^r - \binom{n}{1}\cdot(n-1)^r + \binom{n}{2}\cdot(n-2)^r - \cdots + (-1)^{n-1}\binom{n}{n-1}\cdot 1^r\right), \quad si\ r \geq n$$

TEOREMA 2.2.24 *Sean r y n dos enteros positivos tales que $r \geq n$. Entonces*

$$S(r,1) = 1$$
$$S(r,r) = 1$$
$$S(r+1,n) = S(r,n-1) + n\cdot S(r,n)$$

Más adelante, veremos la demostración de este resultado utilizando el significado de los números de Stirling dentro de la teoría de recuento.

COROLARIO 2.2.25 *Si $|A| = r$ y $|B| = n$ y $n \leq r$, entonces el número de funciones sobreyectivas de A en B es $n!\cdot S(r,n)$.*

EJEMPLO 2.2.26 En un departamento de n profesores se asigna la docencia de n asignaturas: ¿de cuántas maneras se puede hacer el reparto, si hay un profesor de baja y todos los profesores deben impartir alguna asignatura?

El proceso de asignación se corresponde con la definición de aplicaciones sobreyectivas del conjunto de n asignaturas en el conjunto de $n-1$ profesores. Por lo tanto, el número de asignaciones posibles es $(n-1)!S(n,n-1)$.

Otra forma de pensarlo es la siguiente: dado que hay n asignaturas y $n-1$ profesores, un profesor deberá elegir dos asignaturas. Los posibles pares de asignaturas son $\binom{n}{2}$. Cada par de asignaturas y las $n-2$ restantes se pueden repartir de $(n-1)!$ formas posibles, por lo que el resultado final es $(n-1)!\binom{n}{2}$. De lo anterior deducimos que $S(n,n-1) = \binom{n}{2}$. □

EJEMPLO 2.2.27 Si $|A| = n \geq 2$, ¿cuántas aplicaciones biyectivas $f\colon A \to A$ tales que $f(x) \neq x$ para todo $x \in A$ podemos definir?

Las funciones así definidas se denominan **desarreglos** o **permutaciones completas**. Consideremos los conjuntos F_a formados por las aplicaciones de A en A tales que $f(a) = a$. Entonces, el conjunto de las aplicaciones biyectivas definidas en la pregunta se puede escribir como

$$\overline{\bigcup F_a}$$

y por complementariedad y el principio de inclusión-exclusión, el tamaño de este conjunto es

$$\left|\overline{\bigcup F_a}\right| = n! - \left|\bigcup F_a\right| = n! - \sum_{j=1}^{n}(-1)^{j+1}\sum_{\substack{J\subseteq A\\|J|=j}}\left|\bigcap_{a\in J} F_a\right|$$

En este caso, para cada $j \in \{1,\ldots,n\}$, el sumando $\displaystyle\sum_{\substack{J\subseteq A\\|J|=j}}\left|\bigcap_{a\in J} F_a\right|$ coincide con el número de funciones que se comportan como la identidad para al menos j elementos, es decir $P(n, n-j) = \dfrac{n!}{j!}$. Por lo tanto, el número de desarreglos es el siguiente:

$$n! - \sum_{j=1}^{n}(-1)^{j+1}\frac{n!}{j!} = \sum_{j=0}^{n}(-1)^{j}\frac{n!}{j!} = \sum_{j=2}^{n}(-1)^{j}\frac{n!}{j!} \qquad \square$$

2.2.8. Combinaciones con repetición

Las combinaciones que hemos definido en la sección anterior se pueden entender como la forma de seleccionar (o asignar) un número determinado de objetos distintos sin importar el orden en el que se hace la selección (o asignación). Pero que ocurre si disponemos de muchos objetos del mismo tipo y podemos seleccionar o asignar un mismo objeto varias veces, de forma repetida. En este caso, hablamos de *combinaciones con repetición.*

Teorema 2.2.28 *El número de combinaciones de r elementos, posiblemente repetidos, elegidos de un conjunto con n tipos de elementos distintos se denota $CR(n,r)$ y:*

$$\begin{aligned} CR(n,r) &= P(r+n-1; r, n-1)\\ CR(n,r) &= \frac{(r+n-1)!}{r!(n-1)!}\\ CR(n,r) &= \binom{r+n-1}{r} \end{aligned}$$

Ejemplo 2.2.29 ¿Cuántas soluciones enteras no negativas tiene la siguiente ecuación?

$$x + y + z = 5$$

Realmente, esta pregunta es otra forma de representar el modelo de las combinaciones con repetición. Buscamos de cuantas formas podemos tomar x objetos de un tipo, y objetos de un segundo tipo y z elementos de un tercer tipo de forma que en total elijamos 5 objetos. Por lo tanto, el número de soluciones es

$$CR(3,5) = \frac{(5+3-1)!}{5!(3-1)!} = \frac{7\cdot 6\cdot \cancel{5!}}{\cancel{5!}\cdot 2} = \frac{42}{2} = 21 \qquad \square$$

COROLARIO 2.2.30 *El número de soluciones enteras no negativas de la ecuación* $x_1 + x_2 + \cdots + x_n = r$ *es* $CR(n, r)$.

En el resultado anterior, consideramos que las variables x_i pueden ser mayores o iguales que 0 y no están restringidas superiormente. En problemas reales, nos encontraremos que estas variables estarán restringidas superiormente e inferiormente. Las combinaciones con repetición serán el modelo básico para determinarlas, pero tendremos que recurrir al principio de inclusión-exclusión para realizar el cálculo.

EJEMPLO 2.2.31 Para la ecuación $x_1 + x_2 + x_3 = 11$, ¿cuántas soluciones verifican que $0 \leq x_1 \leq 3$, $0 \leq x_2 \leq 4$, $0 \leq x_3 \leq 6$?

Dado que tenemos restricciones adicionales para cada variable, tenemos que utilizar el modelo de las combinaciones con repetición y el principio de inclusión-exclusión. Consideramos los siguientes conjuntos:

$$\begin{aligned} A &= \{(x_1, x_2, x_3) \in \mathbb{N}^3 \colon x_1 + x_2 + x_3 = 11\} \\ A_1 &= \{(x_1, x_2, x_3) \in A \colon x_1 > 3\} = \{(x_1, x_2, x_3) \in A \colon x_1 \geq 4\} \\ A_2 &= \{(x_1, x_2, x_3) \in A \colon x_2 > 4\} = \{(x_1, x_2, x_3) \in A \colon x_2 \geq 5\} \\ A_3 &= \{(x_1, x_2, x_3) \in A \colon x_3 > 6\} = \{(x_1, x_2, x_3) \in A \colon x_3 \geq 7\} \end{aligned}$$

Entonces, el conjunto de las soluciones que buscamos es

$$\overline{A}_1 \cap \overline{A}_2 \cap \overline{A}_3$$

y por el principio de inclusión exclusión:

$$\begin{aligned} &|\overline{A}_1 \cap \overline{A}_2 \cap \overline{A}_3| = |A| - |A_1 \cup A_2 \cup A_3| = \\ &= |A| - \big(|A_1| + |A_2| + |A_3|\big) + \big(|A_1 \cap A_2| + |A_1 \cap A_3| + |A_2 \cap A_3|\big) - |A_1 \cap A_2 \cap A_3| \end{aligned}$$

Podemos determinar el tamaño de cada uno de los conjuntos de esta expresión usando combinaciones con repetición:

$$|A| = CR(3, 11) = \binom{3 + 11 - 1}{2} = 13 \cdot 6 = 78$$

El tamaño de A_1 es $CR(3, 7)$ ya que el número de soluciones de la ecuación $x_1 + x_2 + x_3 = 11$ si $x_1 \geq 4$ coincide con el número de soluciones de la ecuación $(y + 4) + x_2 + x_3 = 11$, es decir, de la ecuación $y + x_2 + x_3 = 7$. El mismo razonamiento se

utiliza en el resto de los conjuntos cuya tamaño tenemos que calcular:

$$\begin{aligned}
|A_1| = CR(3,7) &= \binom{3+7-1}{2} = 9 \cdot 4 = 36 \\
|A_2| = CR(3,6) &= \binom{3+6-1}{2} = 4 \cdot 7 = 28 \\
|A_3| = CR(3,4) &= \binom{3+4-1}{2} = 3 \cdot 5 = 15 \\
|A_1 \cap A_2| = CR(3,2) &= \binom{2+3-1}{3-1} = 2 \cdot 3 = 6 \\
|A_1 \cap A_3| = CR(3,0) &= \binom{0+3-1}{3-1} = 1 \\
|A_2 \cap A_3| &= 0 \\
|A_1 \cap A_2 \cap A_3| &= 0
\end{aligned}$$

Y por lo tanto:

$$|\overline{A}_1 \cap \overline{A}_2 \cap \overline{A}_3| = 78 - (36 + 28 + 15) + (6 + 1 + 0) - 0 = 6 \qquad \square$$

EJEMPLO CON MAXIMA 2.2.32 No disponemos de un operador en `Maxima` que genere las combinaciones con repetición, pero no es difícil hacerlo con los operadores que hemos aprendido hasta ahora. Concretamente, usaremos las soluciones de las ecuaciones lineales como ejemplo básico.

Por ejemplo, las combinaciones con repetición $CR(3,7)$ son las soluciones positivas de $x_1 + x_2 + x_3 = 7$ y por lo tanto, necesitamos generar listas de tres números positivos que sumen 7. Para que el proceso sea fácilmente generalizable, definiremos los operadores sin tener en cuenta la longitud de las misma. Empezamos por definir un operador que determina la suma de los elementos de una lista.

```
(%i1) suma_list(l):=lreduce("+",l)$
(%i2) suma_list([2,3,1,4]);
```

9

Para seleccionar las listas adecuadas, definimos una propiedad sobre listas, en este caso, que la suma sea igual a 7:

```
(%i3) suma7Q(l):=is(suma_list(l)=7)$
```

Ya podemos construir el subconjunto de las combinaciones con repetición.

```
(%i4) A: {0,1,2,3,4,5,6,7};
(%i5) comb_rep37:=subset(cartesian_product(A,A,A),suma7Q);
```

$$\{[0,0,7],[0,1,6],[0,2,5],[0,3,4],[0,4,3],[0,5,2],[0,6,1],[0,7,0],$$
$$[1,0,6],[1,1,5],[1,2,4],[1,3,3],[1,4,2],[1,5,1],[1,6,0],[2,0,5],$$
$$[2,1,4],[2,2,3],[2,3,2],[2,4,1],[2,5,0],[3,0,4],[3,1,3],[3,2,2],$$
$$[3,3,1],[3,4,0],[4,0,3],[4,1,2],[4,2,1],[4,3,0],$$
$$[5,0,2],[5,1,1],[5,2,0],[6,0,1],[6,1,0],[7,0,0]\}$$

```
(%i6) cardinality(comb_rep37);
```

$$36$$

```
(%i7) binomial(3+7-1,7);
```

$$36$$ □

2.2.9. Particiones. Números de Stirling

Una situación que no hemos contemplado en los modelos anteriores es la siguiente: disponemos de r objetos distintos y los queremos repartir en n cajas indistinguibles, de forma que ninguna quede vacía. Estas divisiones se conocen como *particiones*. Concretamente, queremos calcular el número de *particiones de un conjunto de r elementos en n partes*.

Definición 2.2.33 *Una* partición *de un conjunto S en k partes es una familia de subconjuntos, $\mathcal{P} = \{S_1, ..., S_k\}$ tales que:*

- $S_i \neq \varnothing$ *para todo i.*
- $S_i \cap S_j = \varnothing$ *para todo i, j (es decir, son* disjuntos dos a dos*).*
- $S_1 \cup S_2 \cup ... \cup S_k = S$

Ejemplo 2.2.34

- $\mathcal{P} = \{\{1,3\},\{2,5\},\{4\}\}$ es una partición del conjunto $S = \{1,2,3,4,5\}$ en tres partes.
- $\mathcal{P} = \{\{x \in \mathbb{N} \mid x \text{ es par }\},\ \{x \in \mathbb{N} \mid x \text{ es impar }\}\}$ es una partición de $\mathbb{N}$ en dos partes. □

Esta forma de distribución de los elementos coincide con establecer aplicaciones sobreyectivas del conjunto de objetos al conjunto de cajas, pero dado que las posiciones son indistinguibles, debemos dividir por $n!$, que son las formas de reordenar

las n posiciones. Por lo tanto, el número de particiones es

$$\frac{1}{n!}\left(n^r - \sum_{j=1}^{n-1}(-1)^{j+1}\binom{n}{j}\cdot(n-j)^r\right) = S(r,n)$$

Como hemos visto anteriormente, los números de Stirling de Segunda Clase se pueden definir también de forma rucursiva: si n y r son enteros positivos tales que $n \leq r$, entonces

$$\begin{aligned} S(r,1) &= 1 \\ S(r,r) &= 1 \\ S(r+1,n) &= S(r,n-1) + n\cdot S(r,n) \end{aligned}$$

Vamos a utilizar el modelo de recuento de las particiones de un conjunto para demostrar estas tres igualdades. Las dos primeras son triviales: solo hay una forma de distribuir cualquier número de objetos en una caja, y solo hay una forma de distribuir r objetos en r cajas (sin dejar ninguna vacía). El paso recursivo se puede razonar como sigue. Para hacer una distribución de $r+1$ objetos en n cajas, podemos empezar repartiendo los primeros r objetos; si con estos r objetos ocupamos todas las cajas, para lo cual tenemos $S(r,n)$ posibilidades, podremos colocar el último en cualquiera de ellas, lo que nos da un total de $n\cdot S(r,n)$ posibilidades; si con los primeros r objetos dejamos una caja libre, tendremos $S(r,n-1)$ posibilidades y el último objeto tendrá que ocupar necesariamente la caja vacía; de esta forma, hemos demostrado que $S(r+1,n) = S(r,n-1) + n\cdot S(r,n)$.

Ejemplo 2.2.35 Visualicemos algunos números de Stirling de segunda clase determinando explícitamente todas las particiones de un conjunto de cuatro elementos.

$S(4,1)=1$	$S(4,2)=7$	$S(4,3)=6$	$S(4,4)=1$
$\{a,b,c,d\}$	$\{a,b\},\{c,d\}$	$\{a,b\},\{c\},\{d\}$	$\{a\},\{b\},\{c\},\{d\}$
	$\{a,c\},\{b,d\}$	$\{a,c\},\{b\},\{d\}$	
	$\{a,d\},\{b,c\}$	$\{a,d\},\{b\},\{c\}$	
	$\{a,b,c\},\{d\}$	$\{b,c\},\{a\},\{d\}$	
	$\{a,b,d\},\{c\}$	$\{b,d\},\{a\},\{c\}$	
	$\{a,c,d\},\{b\}$	$\{c,d\},\{a\},\{b\}$	
	$\{b,c,d\},\{a\}$		

Ejemplo con Maxima 2.2.36 En `Maxima` disponemos tanto del operador que calcula los números de Stirling de segunda especie, `stirling2`, como del operador que determina las particiones de un conjunto con un determinado número de partes, `set_partitions`.

```
(%i1) partes74: set_partitions({1,2,3,4,5,6,7},4)$
(%i2) cardinality(partes74)=stirling2(7,4);
```

$$350 = 350$$ □

EJEMPLO 2.2.37 ¿De cuántas formas puede factorizarse 30 030 en tres factores? (cada factor es mayor que 1 y naturalmente no importa el orden de los factores)

Dado que $30\,030 = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13$, los tres factores se obtienen agrupando los 6 factores primos en tres grupos no vacíos; por lo tanto, hay $S(6,3)$ ternas de factores:

$$\begin{aligned} S(6,3) &= S(5,2) + 3 \cdot S(5,3) = \\ &= S(4,1) + 2 \cdot S(4,2) + 3(S(4,2) + 3 \cdot S(4,3)) = \\ &= S(4,1) + 5 \cdot S(4,2) + 9 \cdot S(4,3) = 1 + 5 \cdot 7 + 9 \cdot 6 = 90 \end{aligned}$$ □

2.2.10. Funciones generadoras

Para motivar la herramienta que vamos introducir en esta sección, empezamos por un ejemplo que ya hemos abordado en la sección anterior usando combinaciones con repetición.

EJEMPLO 2.2.38 Vamos a determinar el número de soluciones enteras de la ecuación

$$x_1 + x_2 + x_3 = 17$$

tales que: $2 \leq x_1 \leq 5$, $3 \leq x_2 \leq 6$, $4 \leq x_3 \leq 7$

Este ejercicio lo hemos aprendido a hacer utilizando combinaciones con repetición y el principio de inclusión-exclusión. Vamos a utilizar otro método que nos va a permitir resolverlo más fácilmente.

Vamos a considerar el siguiente producto de polinomios en z:

$$P(z) = (z^2 + z^3 + z^4 + z^5)(z^3 + z^4 + z^5 + z^6)(z^4 + z^5 + z^6 + z^7)$$

La variable z no tiene ningún significado en el problema, es la variable del polinomio, que es la verdadera herramienta que usamos en este ejemplo. En los tres factores anteriores hemos tomado como coeficientes de los polinomios el número 1 y las potencias de los términos son los números que pueden ser solución de la ecuación del enunciado. En el primer factor utilizamos los exponentes del 2 al 5, ya que $2 \leq x_1 \leq 5$; en el segundo factor utilizamos los exponentes del 3 al 6, ya que $3 \leq x_2 \leq 6$; en el tercer factor utilizamos los exponentes del 4 al 7, ya que $4 \leq x_3 \leq 7$.

Eliminando los paréntesis obtenemos la forma expandida:

$$\begin{aligned} P(z) &= (z^2 + z^3 + z^4 + z^5)(z^3 + z^4 + z^5 + z^6)(z^4 + z^5 + z^6 + z^7) = \\ &= z^{18} + 3z^{17} + 6z^{16} + 10z^{15} + 12z^{14} + 12z^{13} + 10z^{12} + 6z^{11} + 3z^{10} + z^9 \end{aligned}$$

¿Qué significado tienen los coeficientes que hemos obtenido? Para expandir el producto, multiplicamos un sumando de cada uno de los tres factores, de forma que el coeficiente resultantes siempre es 1 y el grado del término es la suma de los tres

exponentes. Por lo tanto, al sumar todos los términos con el mismo grado, estamos sumando una unidad por cada producto de factores cuya suma coincide con ese grado, es decir, el coeficiente coincide con el número de productos cuya suma de exponentes coincide con ese grado. Por ejemplo, el coeficiente de z^9 es 1, ya que solo hay un producto que genere ese grado, el $z^2z^3z^4$. Sin embargo, el coeficiente de z^{10} es tres, ya que hay tres productos que generan ese grado, $z^2z^3z^5$, $z^2z^4z^4$ y $z^3z^3z^4$.

Por lo tanto, cada coeficiente c_n del polinomio $P(z)$, es el número de soluciones enteras de la ecuación $x_1 + x_2 + x_3 = n$ si $2 \leq x_1 \leq 5$, $3 \leq x_2 \leq 6$, y $4 \leq x_3 \leq 7$. En particular, $c_{17} = 3$ es la respuesta a la pregunta que iniciaba este ejemplo. □

En el ejemplo anterior, la secuencia de coeficientes c_n, para $9 \leq n \leq 18$, nos da las soluciones de una familia de problemas, soluciones que han sido *generadas* por el polinomio $P(z)$. Aunque la expansión del polinomio supone en realidad "contar una a una" las soluciones, el proceso es puramente mecánico y las propiedades de los polinomios evitan tener que establecer estrategias para el recorrido exhaustivo de las soluciones.

Por otra parte, aunque en el ejemplo anterior las restricciones solo permitían generar una familia finita de problemas y soluciones, podremos utilizar la misma técnica aunque necesitemos generar una familia infinita. En estos casos, necesitaremos trabajar con *series de potencias*. Cuando trabajamos con polinomios y series de potencias para generar soluciones de familias de problemas, hablamos de *funciones generadoras*. En este tema, las vamos a utilizar para resolver problemas de recuento, como en el ejemplo anterior, y para determinar el término general de sucesiones definidas de forma recursiva.

DEFINICIÓN 2.2.39 *Sea a_n una sucesión de números reales. Llamamos* ***función generadora (ordinaria)*** *de la sucesión a_n a la serie de potencias*

$$\sum_{n=0}^{\infty} a_n \cdot z^n = a_0 + a_1 z + a_2 z^2 + \dots$$

Las funciones generadoras se denominan igualmente *generatrices*. Aunque vamos a trabajar con series de potencias, nunca necesitaremos conocer su campo de convergencia, ya que la validez de los resultados obtenidos sobre los coeficientes es independiente de dicho campo.

1. $\displaystyle\sum_{n=0}^{\infty} a_n \cdot z^n + \sum_{n=0}^{\infty} b_n \cdot z^n = \sum_{n=0}^{\infty} (a_n + b_n) \cdot z^n$

2. $\displaystyle c \cdot \sum_{n=0}^{\infty} a_n \cdot z^n = \sum_{n=0}^{\infty} c \cdot a_n \cdot z^n$

3. $\dfrac{1-z^{m+1}}{1-z} = \sum_{n=0}^{m} z^n = 1 + z + z^2 + z^3 + \cdots + z^m$

4. $\dfrac{1}{1-z} = \sum_{n=0}^{\infty} z^n$

5. $(1+z)^\alpha = \sum_{n=0}^{\infty} \binom{\alpha}{n} z^n = \sum_{n=0}^{\infty} \dfrac{\alpha(\alpha-1)...(\alpha-n+1)}{n!} z^n$

6. Si $m \in \mathbb{N}$, $\quad (1+z)^m = \sum_{n=0}^{m} \binom{m}{n} z^n$

7. Si $m \in \mathbb{N}$, $\quad (1-z)^{-m} = \sum_{n=0}^{\infty} \binom{n+m-1}{n} z^n = \sum_{n=0}^{\infty} \binom{n+m-1}{m-1} z^n$

Vamos a ver en el siguiente ejemplo, las importantes consecuencias prácticas de estas propiedades. Concretamente, la igualdad 3 nos permite trabajar con polinomios y multiplicarlos de forma más sencilla; las igualdades 6 y 7 nos ayudarán a generar más fácilmente los coeficientes tras agrupar numeradores y denominadores. El objetivo será transformar un producto de más de dos polinomios o series y reducirlo a dos factores.

Ejemplo 2.2.40 Para cada n, vamos a calcular el número c_n, de soluciones de la ecuación

$$x_1 + x_2 + x_3 = n$$

tales que $2 \leq x_1 \leq 5$, $3 \leq x_2 \leq 6$, y $4 \leq x_3$.

Procedemos como en el ejemplo anterior, construyendo un polinomio que represente las restricciones de cada variable.

$$\begin{aligned} 2 \leq x_1 \leq 5 \quad &\Longrightarrow \quad (z^2 + z^3 + z^4 + z^5) \\ 3 \leq x_2 \leq 6 \quad &\Longrightarrow \quad (z^3 + z^4 + z^5 + z^6) \\ 4 \leq x_3 \quad &\Longrightarrow \quad \sum_{k=4}^{\infty} z^k \end{aligned}$$

Dado que la variable x_3 no está acotada superiormente, la suma que representa esta restricción es infinita. La función generadora de la sucesión c_n que buscamos en este ejemplo, es el producto de estas tres expresiones.

$$P(z) = \sum_{n=0}^{\infty} c_n z^n = (z^2 + z^3 + z^4 + z^5)(z^3 + z^4 + z^5 + z^6) \sum_{k=4}^{\infty} z^k$$

Utilizando las propiedades que hemos visto antes, vamos a transformar esta expresión

en el cociente de un polinomio entre una potencia de $(1-z)$.

$$\begin{aligned}
P(z) = \sum_{n=0}^{\infty} c_n z^n &= (z^2+z^3+z^4+z^5)(z^3+z^4+z^5+z^6)\sum_{k=4}^{\infty} z^k = \\
&= z^9(1+z+z^2+z^3)(1+z+z^2+z^3)\sum_{k=0}^{\infty} z^k = && \text{(Factor común)} \\
&= z^9(1+z+z^2+z^3)^2\sum_{k=0}^{\infty} z^k = \\
&= z^9\left(\frac{1-z^4}{1-z}\right)^2\sum_{k=0}^{\infty} z^k = && \text{(Propiedad 3)} \\
&= z^9\left(\frac{1-z^4}{1-z}\right)^2\frac{1}{1-z} = && \text{(Propiedad 4)} \\
&= z^9(1-z^4)^2\frac{1}{(1-z)^3}
\end{aligned}$$

Finalmente, con la propiedad 7 escribimos la función generatriz como el producto de un polinomio por una serie

$$P(z) = z^9(1-z^4)^2\frac{1}{(1-z)^3} = (z^{17}-2z^{13}+z^9)\sum_{k=0}^{\infty}\binom{k+2}{2}z^k = \sum_{n=0}^{\infty} c_n z^n$$

A partir de esa igualdad es más fácil determinar el valor de cada coeficiente c_n

- Si $0 \leq n \leq 8$, entonces $c_n = 0$
- Para n entre 9 y 12, el coeficiente de z^n se obtiene del producto de z^9 por los primeros términos de la serie, de donde

$$\begin{aligned}
c_9 &= \binom{0+2}{2} = 1, \qquad c_{10} = \binom{1+2}{2} = 3, \\
c_{11} &= \binom{2+2}{2} = 6, \qquad c_{12} = \binom{3+2}{2} = 10
\end{aligned}$$

- Para n entre 13 y 16, los coefientes de z^n se obtienen del producto de z^9 por los siguientes sumandos de la serie y del producto de z^{13} por los primeros términos,

$$\begin{aligned}
c_{13} &= -2\binom{0+2}{2} + \binom{4+2}{2} = 13, \qquad c_{14} = -2\binom{1+2}{2} + \binom{5+2}{2} = 15, \\
c_{15} &= -2\binom{2+2}{2} + \binom{6+2}{2} = 16, \qquad c_{16} = -2\binom{3+2}{2} + \binom{7+2}{2} = 16
\end{aligned}$$

- Los coeficientes a partir de c_{17} se obtienen de forma similar como suma de tres

posibles productos, aunque el valor resultante siempre es el mismo.

$$\begin{aligned} c_{17+m} &= \binom{m+2}{2} - 2\binom{m+4+2}{2} + \binom{m+8+2}{2} = \\ &= \frac{1}{2}(m+2)(m+1) - (m+6)(m+5) + \frac{1}{2}(m+10)(m+9) = \\ &= \frac{m^2}{2} + \frac{3m}{2} + 1 - m^2 - 11m - 30 + \frac{m^2}{2} + \frac{19m}{2} + 45 = 16 \end{aligned}$$ □

EJEMPLO CON MAXIMA 2.2.41 Naturalmente, si utilizamos `Maxima`, no necesitaremos realizar las transformaciones del ejemplo anterior. Por ejemplo, vamos a determinar el número de soluciones de la ecuación $x_1 + x_2 + x_3 = 30$ teniendo en cuenta que $1 \leq x_1 \leq 10$, $2 \leq x_2 \leq 7$, $1 \leq x_3$. Vamos a hacerlo de dos formas. En la primera vamos a utilizar que, dado que la suma de los tres sumandos es 30, ninguno de ellos puede ser mayor que 30, es decir, $x_3 \leq 30$. De esta forma, la función generadora de este problema es el siguiente polinomio:

```
(%i1) gen: sum(z^n,n,1,10)*sum(z^n,n,2,7)*sum(z^n,n,1,30),
      expand;
```

$$\begin{aligned} z^{47} + 3z^{46} + 6z^{45} + 10z^{44} + 15z^{43} + 21z^{42} + 27z^{41} + \\ + 33z^{40} + 39z^{39} + 45z^{38} + 50z^{37} + 54z^{36} + 57z^{35} + 59z^{34} + \\ + 60z^{33} + 60z^{32} + 60z^{31} + 60z^{30} + 60z^{29} + 60z^{28} + 60z^{27} + \\ + 60z^{26} + 60z^{25} + 60z^{24} + 60z^{23} + 60z^{22} + 60z^{21} + 60z^{20} + \\ + 60z^{19} + 60z^{18} + 59z^{17} + 57z^{16} + 54z^{15} + 50z^{14} + 45z^{13} + \\ + 39z^{12} + 33z^{11} + 27z^{10} + 21z^9 + 15z^8 + 10z^7 + 6z^6 + 3z^5 + z^4 \end{aligned}$$

Aunque podemos ver fácilmente cual es el coeficiente de z^{30}, también podemos determinarlo con el operador `coeff`. Hay que tener en cuenta que para que el resultado sea correcto, debemos aplicar este operador a una expresión polinómica en su forma expandida, tal y como estamos haciendo en este ejemplo.

```
(%i2) coeff(gen,z,30);
```

$$60$$

La función generadora anterior solo nos sirve para encontrar el numero de soluciones de $x_1+x_2+x_3 = m$ si $m \geq 30$, ya que hemos añadido la restricción $x_3 \geq 30$. Si queremos hallar la función generadora que podamos utilizar para cualquier valor de m con las restricciones que habíamos indicado al principio, $1 \geq x_1 \geq 10$, $2 \geq x_2 \geq 7$, $1 \geq x_3$., tendremos que usar series:

```
(%i3) gen1: sum(z^n,n,1,10)*sum(z^n,n,2,7)*sum(z^n,n,1,inf)$
```

En este caso, no podemos recurrir directamente el operador `coeff`, ya que la expresión no está expandida:

```
(%i4) coeff(gen1(z),z,30);
```

$$0$$

La alternativa a la opción `expand` cuando trabajamos con series es el operador `taylor`, que determina el polinomio de Taylor de la función generadora hasta el grado que deseemos y que en este caso coincide con la forma expandida truncada en ese grado:

```
(%i5) taylor(gen1,z,0,35);
```

$$\begin{aligned} z^4+3z^5+6z^6+10z^7+15z^8+21z^9+27z^{10}+33z^{11}+39z^{12}+45z^{13}+50z^{14}+\\ +54z^{15}+57z^{16}+59z^{17}+60z^{18}+60z^{19}+60z^{20}+60z^{21}+60z^{22}+60z^{23}+\\ +60z^{24}+60z^{25}+60z^{26}+60z^{27}+60z^{28}+60z^{29}+60z^{30}+60z^{31}+60z^{32}+\\ +60z^{33}+60z^{34}+60z^{35}+\ldots \end{aligned}$$

```
(%i6) coeff(%,z,30);
```

$$60$$ □

2.2.11. Ecuaciones en recurrencia

En el siguiente ejemplo, vemos cómo las funciones generadoras se pueden utilizar para obtener la expresión explícita de una sucesión definida de forma recursiva.

Ejemplo 2.2.42 Vamos a encontrar la expresión explícita o término general de la sucesión a_n que verifica: $a_0 = 0$, $a_1 = 1$ y $a_n - 3a_{n-1} + 2a_{n-2} = 0$.

Concretamente, vamos a tomar la función generadora de la sucesión e intentaremos utilizar las propiedades que hemos visto anteriormente para deducir una expresión de cada coeficiente. Sea $G(z)$ la función generadora

$$G(z) = a_0 + a_1 \cdot z + a_2 \cdot z^2 + \cdots + a_n \cdot z^n + \cdots = \sum_{n=0}^{\infty} a_n z^n$$

A continuación, vamos operar sobre esta expresión para poder utilizar la igualdad $a_n - 3a_{n-1} + 2a_{n-2} = 0$ y simplificar la expresión resultante; multiplicamos entonces la función por $-3z$ y por $2z^2$ y sumaremos las tres igualdades obtenidas

$$\begin{array}{rccccccc} G(z) = & a_0 & +a_1 z & +a_2 z^2 & +a_3 z^3 & +a_4 z^4 & \ldots \\ -3zG(z) = & & -3za_0 & -3a_1 z^2 & -3a_2 z^3 & -3a_3 z^4 & \ldots \\ +2z^2 G(z) = & & & +2a_0 z^2 & +2a_1 z^3 & +2a_2 z^4 & \ldots \end{array}$$

Si en lado de la derecha, sumamos las columnas con tres sumandos, el resultado es 0; por ejemplo,

$$(a_2 - 3a_1 + 2a_0)z^2 = 0 \cdot z^2, \qquad (a_3 - 3a_2 + 2a_1)z^3 = 0 \cdot z^3, \ldots$$

Por lo tanto

$$\begin{array}{rlllllll}
G(z) = & a_0 & +a_1 z & +a_2 z^2 & +a_3 z^3 & +a_4 z^4 & \ldots \\
-3zG(z) = & & -3za_0 & -3a_1 z^2 & -3a_2 z^3 & -3a_3 z^4 & \ldots \\
+2z^2 G(z) = & & & +2a_0 z^2 & +2a_1 z^3 & +2a_2 z^4 & \ldots \\
\hline
(1 - 3z + 2z^2)G(z) = & a_0 + a_1 z - 3za_0 & & & & & \\
(1 - 3z + 2z^2)G(z) = & z & & & & &
\end{array}$$

y en consecuencia

$$G(z) = \frac{z}{1 - 3z + 2z^2}$$

A partir de aquí, podemos utilizar las propiedades de las series de potencias para encontrar el valor de los coeficientes de G.

$$G(z) = \frac{z}{1 - 3z + 2z^2} = \frac{-1}{1 - z} + \frac{1}{1 - 2z} = -\sum_{n=0}^{\infty} z^n + \sum_{n=0}^{\infty} (2z)^n = \sum_{n=0}^{\infty} (2^n - 1)z^n$$

Por lo tanto, $a_n = 2^n - 1$.

Los dos sumando obtenidos en la forma explícita son exponenciales de base 1 y 2, que son los inversos de las soluciones de la ecuación polinómica que se obtiene a partir del denominador de la función generadora, $1 - 3z + 2z^2 = 0$. Si realizamos el cambio de variable $z = 1/r$, obtenemos una ecuación polinómica equivalente:

$$0 = 1 - \frac{3}{r} + \frac{2}{r^2} = \frac{r^2 - 3r + 2}{r^2} \quad \Rightarrow \quad 0 = r^2 - 3r + 2 = (r - 1)(r - 2)$$

Los coeficientes de esta ecuación son los coeficientes de la ecuación en recurrencia y sus soluciones son la base de las exponenciales que determinan la solución de la ecuación en recurrencia. A la ecuación numérica $r^2 - 3r + 2 = 0$ se la denomina ecuación característica y como vemos a continuación es la herramienta para la resolución de ecuaciones en recurrencia lineales de coeficientes constantes. □

Definición 2.2.43 *Una* relación de recurrencia *para la sucesión a_n es una igualdad que determina el término a_n en función de los términos anteriores.*

Para caracterizar una sucesión mediante una relación de recurrencia, es necesario considerar tantas *condiciones iniciales* como el número de términos anteriores que intervienen en la definición recursiva. Por ejemplo, la sucesión $a_n = 3n$ verifica la relación de recurrencia $a_n = 2a_{n-1} - a_{n-2}$. La sucesión constante $b_n = 5$ también verifica una relación de recurrencia, $b_n = 2b_{n-1} - b_{n-2}$. Sin embargo, los primeros términos de las dos sucesiones son distintos: $a_0 = 0$, $b_0 = 5$, $a_1 = 3$, $b_0 = 5$,...

Definición 2.2.44 *Una* relación de recurrencia lineal homogénea de orden k *y con* ***coeficientes constantes*** *es una relación de recurrencia de la forma*

$$a_n = c_1 \cdot a_{n-1} + c_2 \cdot a_{n-2} + \cdots + c_k \cdot a_{n-k}$$

en donde $c_1, \ldots, c_k \in \mathbb{R}$ *y* $c_k \neq 0$.

Ejemplo 2.2.45

1. $f_n = f_{n-1} + f_{n-2}$, $n \geq 2$ es una relación de recurrencia lineal y homogénea de orden 2.
2. La relación $a_n = a_{n-4}$ es una relación de recurrencia lineal y homogénea de orden 4.
3. La relación $a_n = a_{n-1} + a_{n-2}^2$ no es lineal.
4. La relación $q_{n+1} = 2q_n + 4^n$ es lineal, pero no es homogénea. □

Podemos determinar el término general de una sucesión definida por una recurrencia lineal homogénea utilizando funciones generadoras, tal y como hemos visto en la sección anterior. Vamos a ver, sin embargo, un resultado que simplifica este proceso.

Definición 2.2.46 *Dada la recurrencia lineal homogénea*

$$a_n = c_1 \cdot a_{n-1} + c_2 \cdot a_{n-2} + \cdots + c_k \cdot a_{n-k}$$

Llamamos ecuación característica *de esta relación, a la siguiente ecuación polinómica (númerica) en* r*:*

$$r^k - c_1 \cdot r^{k-1} - c_2 \cdot r^{k-2} - \cdots - c_k = 0$$

Las soluciones de la ecuación característica determinan la forma de la expresión explícita de la sucesión definida por la relación de recurrencia, tal y como establece el siguiente resultado.

Teorema 2.2.47 *Supongamos que*

$$r^k - c_1 r^{k-1} - c_2 r^{k-2} - \ldots - c_k = 0$$

es la ecuación característica de una recurrencia lineal homogénea y que r_1, r_2, …, r_t, *son las soluciones (reales) de esta ecuación con multiplicidades* m_1, m_2,…, m_t *respectivamente. Entonces* a_n *es solución de la recurrencia lineal si y solo si*

$$a_n = p_1(n) \cdot r_1^n + p_2(n) \cdot r_2^n + \ldots + p_t(n) \cdot r_t^n,$$

en donde cada p_i *es un polinomio de grado estrictamente menor que* m_i.

Ejemplo 2.2.48 $a_0 = 0$, $a_1 = 1$, $a_n - 3a_{n-1} + 2a_{n-2} = 0$.

La ecuación característica es $r^2 - 3r + 2 = 0$ y sus soluciones son $r_1 = 2$, $r_2 = 1$, ambas con multiplicidad 1. Por lo tanto, los polinomios p_1 y p_2 definidos por el teorema anterior son constantes y podemos afirmar que la sucesión tiene la forma

$$a_n = A \cdot 2^n + B \cdot 1^n$$

en donde A y B son dos constantes, que calculamos a partir de los primeros términos de la sucesión.

$$\left.\begin{array}{l} 0 = a_0 = A \cdot 2^0 + B = 0 \\ 1 = a_1 = A \cdot 2^1 + B = 1 \end{array}\right\} \Longrightarrow \left\{\begin{array}{rl} A + B &= 0 \\ 2A + B &= 1 \end{array}\right\} \Longrightarrow \left\{\begin{array}{l} A = 1 \\ B = -1 \end{array}\right.$$

Por lo tanto, $a_n = 2^n - 1^n = 2^n - 1$, $n \geq 0$. □

Ejemplo 2.2.49 $a_0 = 1$, $a_1 = -2$, $a_2 = -1$, $a_n = -3a_{n-1} - 3a_{n-2} - a_{n-3}$.

La ecuación característica de esta recurrencia es $r^3 + 3r^2 + 3r + 1 = 0$. Dado que $r^3 + 3r^2 + 3r + 1 = (r+1)^3$, la ecuación tiene a $r_1 = -1$ como única solución con multiplicidad $m_1 = 3$. Por el teorema de caracterización de las soluciones, esta tiene la forma

$$a_n = (A + Bn + Cn^2)(-1)^n$$

Y utilizando los tres primeros términos de la sucesión, obtenemos el siguiente sistema

$$\left.\begin{array}{ll} a_0 \to & A = 1 \\ a_1 \to & -A - B - C = -2 \\ a_2 \to & A + 2B + 4C = -1 \end{array}\right\} \quad \Longrightarrow \quad \left\{\begin{array}{l} A = 1 \\ B = 3 \\ C = -2 \end{array}\right.$$

Por lo tanto, $a_n = (1 + 3n - 2n^2) \cdot (-1)^n$. □

Ejemplo con Maxima 2.2.50 En este primer ejemplo con `Maxima` vamos a utilizar el paquete `solve_rec`, que resuelve directamente ecuaciones en recurrencia lineales (homogéneas o no).

```
(%i1) load(solve_rec)$
```

Primero introducimos la ecuación, en donde la sucesión a determinar se escribe como una lista. Si escribimos solo una expresión involucrando elementos de la lista, `Maxima` entiende que la expresión está igualada a 0. Por ejemplo, la ecuación en recurrencia de orden 3 dada por $18a_n - 21a_{n-1} + 8a_{n-2} - a_{n-3} = 0$ se introduce como sigue:

```
(%i2) ecrec: 18*a[n]-21*a[n-1]+8*a[n-2]-a[n-3]$
```

El operador que resuelve la ecuación es `solve_rec` y toma como argumentos la ecuación y la incógnita.

```
(%i3) solve_rec(ecrec,a[n]);
```

$$a[n] = \frac{\%k_3 * n + \%k_2}{3^n} + \frac{\%k_1}{2^n}$$

En este caso, nos devuelve la solución general en función de tres parámetros, $\%k_1$, $\%k_2$ y $\%k_3$, ya que la ecuación es de orden 3.

También podemos añadir, como argumentos, las condiciones iniciales de la ecuación y, en tal caso, nos devolverá la solución particular correspondiente.

```
(%i1) solve_rec(ecrec,a[n],a[0]=1,a[1]=5/6,a[2]=17/36);
```

$$a[n] = \frac{n}{3^n} + \frac{1}{2^n}$$ □

Ejemplo con Maxima 2.2.51 En este ejemplo, vamos resolver una ecuación en recurrencia usando la ecuación característica. Concretamente, vamos a resolver la ecuación que determina la sucesión de Fibonacci:

$$f[n] - f[n-1] - f[n-2] = 0, \quad f[0] = 0, \quad f[1] = 1.$$

La ecuación característica es $r^2 - r - 1 = 0$:

```
(%i1) eccar: r^2-r-1$
(%i2) solve(eccar,r);
```

$$r = -\frac{\sqrt{5}-1}{2}, \quad r = \frac{\sqrt{5}+1}{2}$$

Por lo tanto, el esquema de la solución de la ecuación en recurrencia es:

```
(%i3) sol: A*(-(sqrt(5)-1)/2)^n+B*((sqrt(5)+1)/2)^n$
```

Para hallar los valores de A y B evaluamos esta expresión con $n = 0$ y $n = 1$ e igualamos los resultados a los correspondientes valores iniciales:

```
(%i4) ec1: ev(sol,n=0)=0$
(%i5) ec2: ev(sol,n=1)=1$
```

El operador `solve` nos da los valores de A y B:

```
(%i6) solve([ec1,ec2],[A,B]);
```

$$A = \frac{-1}{\sqrt{5}}, \quad B = \frac{1}{\sqrt{5}}$$

Por lo tanto, la solución de la ecuación en recurrencia que nos da la forma explícita de la sucesión de Fibonacci:

$$a[n] = \frac{-1}{\sqrt{5}} \cdot \left(\frac{-\sqrt{5}+1}{2}\right)^n + \frac{1}{\sqrt{5}} \cdot \left(\frac{\sqrt{5}+1}{2}\right)^n$$

El resultado obtenido con el operador `solve_rec` es naturalmente el mismo.

```
(%i7) load(solve_rec)$
(%i8) solve_rec(f[n]-f[n-1]-f[n-2]=0, f[n], f[0]=0, f[1]=1.);
```

$$f[n] = \frac{(\sqrt{5}+1)^n}{\sqrt{5}\cdot 2^n} - \frac{(\sqrt{5}-1)^n(-1)^n}{\sqrt{5}\cdot 2^n} \qquad \square$$

2.2.12. Recurrencias lineales no homogéneas

DEFINICIÓN 2.2.52 *Una relación de* recurrencia lineal no homogénea con coeficientes constantes *es una relación de recurrencia de la forma*

$$a_n = c_1 \cdot a_{n-1} + c_2 \cdot a_{n-2} + \cdots + c_k \cdot a_{n-k} + F(n)$$

en donde $c_1, \ldots, c_k \in \mathbb{R}$ *y la función* $F(n)$ *no es nula.*

En este curso, solo vamos a trabajar con recurrencias no homogéneas en las cuales la función F es de la forma $F(n) = p(n)b^n$, en donde $b \in \mathbb{R}$ y $p(n)$ es un polinomio. Para este tipo de funciones, las recurrencias se resuelven utilizando el mismo método de las ecuaciones características.

TEOREMA 2.2.53 *Sea* $b \in \mathbb{R}$ *y* $p(n)$ *un polinomio de grado* q*. Entonces, las soluciones de recurrencia lineal no homogénea*

$$a_n = c_1 \cdot a_{n-1} + c_2 \cdot a_{n-2} + \cdots + c_k \cdot a_{n-k} + b^n \cdot p(n)$$

coinciden con las soluciones de las recurrencia lineal homogénea cuya ecuación característica es

$$(r^n - c_1 r^{n-1} - \ldots - c_k)(r-b)^{q+1} = 0$$

EJEMPLO 2.2.54 $a_0 = 3$, $a_1 = -2$, $a_n - a_{n-1} - 2a_{n-2} = 2$

Dado que $2 = 2 \cdot 1^n$, la ecuación característica de esta recurrencia es

$$(r^2 - r - 2)(r-1)^{0+1} = 0$$

y $r_1 = -1$, $r_2 = 2$ y $r_3 = 1$ son sus soluciones, todas ellas con multiplicidad 1. Por lo tanto, la solución tiene la siguiente forma

$$a_n = A(-1)^n + B \cdot 2^n + C \cdot 1^n,$$

Los valores de las constantes A, B y C los calculamos utilizando los tres primeros términos de la sucesión.

$$\left.\begin{array}{rrrrr} a_0 \to & A & + & B & + C = 3 \\ a_1 \to & -A & + 2B & + C = -2 \\ a_2 \to & A & + 4B & + C = 6 \end{array}\right\} \quad \Rightarrow \quad \left\{\begin{array}{l} A = 3 \\ B = 1 \\ C = -1 \end{array}\right.$$

Por lo tanto, $a_n = 3 \cdot (-1)^n + 1 \cdot 2^n + (-1) \cdot 1^n = 3(-1)^n + 2^n - 1$, para todo $n \geq 0$.

También podemos hallar la solución con `Maxima`.

```
(%i1) load(solve_rec)$
(%i2) solve_rec(a[n] - a[n-1] - 2*a[n-2] = 2,
                                    a[n], a[0]=3, a[1]=-2);
```

$$a[n] = 2^n + 3(-1)^n - 1 \qquad \square$$

EJEMPLO 2.2.55 $a_0 = 5$, $a_1 = 6$, $a_n = 3a_{n-1} - 2a_{n-2} - 2^{n-1}$

Dado que $2^n = 1 \cdot 2^n$, la ecuación característica de esta recurrencia es

$$(r^2 - 3r + 2)(r - 2)^{0+1} = (r^2 - 3r + 2)(r - 2) = 0$$

Las soluciones de la ecuación característica son: $r_1 = 1$, con multiplicidad 1 y $r_2 = 2$ una multiplicidad 2. Por lo tanto, la solución tiene la forma

$$a_n = A \cdot 1^n + (B + Cn)2^n,$$

y los coeficientes se determinan utilizando los tres primeros términos de la sucesión

$$\left.\begin{array}{rl} a_0 \to & A + B = 5 \\ a_1 \to & A + 2B + 2C = 6 \\ a_2 \to & A + 4B + 8C = 6 \end{array}\right\} \quad \Rightarrow \quad \left\{\begin{array}{l} A = 2 \\ B = 3 \\ C = -1 \end{array}\right.$$

Por lo tanto, $a_n = 2 + (3 - n)2^n$, para todo $n \geq 0$.

También podemos hallar la solución con `Maxima`.

```
(%i1) load(solve_rec)$
(%i2) solve_rec(a[n] = 3*a[n-1] - 2*a[n-2] - 2^(n-1),
                                    a[n], a[0]=5, a[1]=6);
```

$$a[n] = -n \cdot 2^n + 3 \cdot 2^n + 2 \qquad \square$$

2.2.13. Recuento recursivo

Las funciones generadoras y las ecuaciones en recurrencia nos permiten plantear y resolver problemas de recuento enunciados en función de un parámetro. En primer lugar, encontraríamos una regla recursiva que describa el problema y utilizaremos la técnica adecuada para resolverla.

EJEMPLO 2.2.56 Un robot sube escaleras de forma errática. A veces sube dos peldaños de golpe, a veces sólo uno. Vamos a hallar una fórmula para expresar el número de maneras distintas de subir n peldaños.

Sea b_n, la sucesión que queremos determinar, es decir, b_n es el número de maneras distintas de subir n peldaños. Solo hay una forma de que el robot suba un único peldaño, es decir, $b_1 = 1$. Para subir dos peldaño, tenemos dos posibilidades, subirlos de una sola vez, o en dos pasos; es decir, $b_2 = 2$.

Para alcanzar el escalón $n+2$, puede alcanzar el escalón $n+1$ y entonces subir el siguiente, o puede haber llegado al escalón n y desde ahí subir dos escalones de golpe. Por lo tanto, las formas de alcanzar el $n+2$ es la suma de las formas de alcanzar el $n+1$ mas las formas de alcanzar el n, es decir, $b_{n+2} = b_{n+1} + b_n$, si $n \geq 1$.

Por lo tanto, la solución del problema (o familia de problemas) de recuento se obtiene como solución de la ecuación de recurrencia $b_{n+2} = b_{n+1} + b_n$, con condiciones iniciales $b_1 = 1$ y $b_2 = 2$. □

Ejemplo 2.2.57 Sea q_n el número de palabras de longitud n que se pueden formar con símbolos del alfabeto $\{0, 1\}$ y tales que no contienen dos ceros consecutivos. Vamos a dar una descripción recursiva de q_n.

Las dos posibles cadenas de longitud 1 verifican trivialmente la condición, y por lo tanto, $q_1 = 2$. Hay tres palabras de longitud 2 que verifican la condición, 11, 10, y 01, por lo tanto, $q_2 = 3$.

Para construir una palabra de longitud n, sin dos ceros consecutivos, podemos añadir un 1 a una palabra de longitud $n-1$ o bien añadir la secuencia 10 a una palabra de longitud $n-2$. Por lo tanto: $q_n = q_{n-1} + q_{n-2}$. □

Relación de ejercicios 4

1. Establece si son verdaderas o falsas las siguientes relaciones:

I) $a \in \{a\}$	II) $\{a\} \in \{a\}$	III) $\{a,b\} \in \{a,\{a,b\}\}$
IV) $a \subseteq \{a\}$	V) $\{a\} \subseteq \{a\}$	VI) $\{a,b\} \subseteq \{a,\{a,b\}\}$

2. Sean los conjuntos $A_1 = \{-2,-1,0,1,2\}$, $A_2 = \{0,1,2\}$, $A_3 = \{-1,0,1\}$ y sea el conjunto de índices $I = \{1,2,3\}$.

   - Determina los siguientes conjuntos: *a)* $\bigcup_{i\in I} A_i$ *b)* $\bigcap_{i\in I} A_i$
   - Tomando $\mathbb{Z}$ como conjunto universal, determina:

   c) $\bigcup_{i\in I} \overline{A_i}$ *d)* $\bigcap_{i\in I} \overline{A_i}$

3. En el conjunto $\mathbb{N}$ de los números naturales se consideran los subconjuntos siguientes:
   P: conjunto de números naturales primos; D: conjunto de múltiplos de dos; T: conjunto de múltiplos de tres; I: conjunto de números impares y S: conjunto de múltiplos de seis.

   - Determina o escribe de forma alternativa:
     a) $P \cap I$, *b)* $P \cap D$, *c)* $D \cap T$, *d)* $D \cap S$, *e)* $I \cap S$.
   - Determina o escribe de forma alternativa el complementario de:
     f) P, *g)* I, *h)* D.
   - Determina o escribe de forma alternativa:
     i) $P \cup I$, *j)* $P - I$, *k)* $\overline{D \cap I}$.

4. Demuestra la validez de las siguientes igualdades para cualquier terna de conjuntos A, B y C.

   a) $A - (B \cup C) = (A - B) \cap (A - C)$

   b) $A \cup (B - C) = (A \cup B) - (\overline{A} \cap C)$

5. Da un contraejemplo que demuestre que la siguiente igualdad no es válida para cualesquiera conjuntos A, B y C:

$$A - (B - C) = (A - B) - C$$

6. La operación $\triangle$, *diferencia simétrica* de los conjuntos A y B se define:

$$A \triangle B \stackrel{def}{=} (A - B) \cup (B - A)$$

   a) Demuestra la igualdad $A \cap (B \triangle C) = (A \cap B) \triangle (A \cap C)$

b) Da un contraejemplo para demostrar que no se verifican la igualdad $A \triangle (B \cup C) = (A \triangle B) \cup (A \triangle C)$

7. Consideramos la siguiente relación:

$$\mathcal{R} = \{(0,1),(1,2),(2,4),(3,8),(4,8),(5,4),(6,2)\}$$

a) Escribe la matriz de adyacencia de la relación.

b) Estudia si la relación es una función y, en tal caso, determina su dominio, su codominio, su rango y estudia si es inyectiva y sobreyectiva.

8. Consideramos las conjuntos $A = \{0,1,2,3,4\}$, $B = \{a,b,c,d\}$ y la siguiente relación de A en B:

$$\mathcal{R} = \{(0,b),(1,d),(3,a),(4,c)\}$$

Escribe su matriz de adyacencia, estudia si es una función y, en tal caso, sus elementos característicos y sus propiedades.

9. Consideramos las conjuntos $A = \{a,b,c,d,e\}$, $B = \{0,1,2,3,4\}$ y la siguiente relación de A en B:

$$\mathcal{R} = \{(a,0),(b,2),(c,3),(d,4),(e,2)\}$$

Escribe su matriz de adyacencia, estudia si es una función y, en tal caso, sus elementos característicos y sus propiedades.

10. Consideramos las conjuntos $A = \{a,b,c,d,e\}$, $B = \{0,1,2,3,4\}$ y la siguiente relación de A en B:

$$\mathcal{R} = \{(c,3),(a,0),(e,2),(b,2),(a,1),(d,4)\}$$

Escribe su matriz de adyacencia, estudia si es una función y, en tal caso, sus elementos característicos y sus propiedades.

11. Sea el conjunto $X = \{a,b,c,d\}$ y $f \subseteq X \times X$ la relación binaria dada por la matriz

$$\mathcal{M}_f = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & - & 1 \\ - & - & - & - \end{pmatrix}$$

Completa la matriz $\mathcal{M}_f$ sabiendo que f es una función inyectiva.

12. Dados los conjuntos $A = \{1,2,3,4,5\}$ y $B = \{6,7,8,9\}$, se define la relación $\mathcal{R} \subseteq A \times B$

$$\mathcal{R} = \{(1,8),(2,6),(5,y),(5,7),(x,z),(t,8)\}$$

En cada uno de los apartados, encuentra todos los posibles valores de las variables $x,t \in A$, $y,z \in B$ de tal forma que:

a) $\mathcal{R}$ sea una función.

b) $\mathcal{R}$ sea una función inyectiva.

c) $\mathcal{R}$ sea una función, pero no sea sobreyectiva.

13. Sea la función $f : \mathbb{Z}_{51} \to \mathbb{Z}_{51}$ definida $f([x]_{51}) = [3x]_{51}$.

    a) Halla la imagen de $[20]_{51}$.

    b) Encuentra, si existe, la preimagen de $[21]_{51}$ y la de $[22]_{51}$.

    c) Analiza si f es inyectiva, sobreyectiva y biyectiva.

14. Demuestra que si escogemos cinco números cualesquiera entre el 1 y el 8, dos de ellos suman 9.

15. Demuestra que si $g \circ f$ es sobreyectiva, entonces g también es sobreyectiva.

Relación de ejercicios 5

1. ¿Cuántas permutaciones de las letras $ABCDEF$ contienen las letras DEF juntas en cualquier orden?

2. Consideramos el siguiente conjunto de símbolos $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

   a) Calcula cuántas cadenas de cuatro elementos de Σ el número 7 aparece exactamente una vez.

   b) Calcula cuántas cadenas de cuatro elementos de Σ el número 7 aparece a lo sumo una vez.

   c) Calcula cuántas cadenas de cuatro elementos de Σ el número 7 aparece al menos una vez.

3. Consideramos el número $29\,338\,848\,000 = 2^8 3^5 5^3 7^3 11$

   a) ¿Cuántos divisores positivos tiene este número?

   b) ¿Cuántos son múltiplos de 99?

   c) ¿Y de 39?

4. Un autobús de 32 plazas (16 a la derecha y 16 a la izquierda) transporta a 28 alumnos de la E.T.S. Ingeniería Informática en su viaje de fin de carrera. ¿De cuántas formas pueden sentarse si tres de ellos sólo pueden ir a la derecha y cinco de ellos sólo a la izquierda?

5. Se ha producido un robo y la policía interroga a dos testigos sobre la matrícula del vehículo utilizado para la huida (cuatro dígitos y dos letras de un alfabeto de 26).

   El primer testigo asegura que la segunda letra de la matrícula era una O o una Q y que el último dígito era un 3 o un 8.

   El segundo testigo asegura que la primera letra era una C o una G y que el primer dígito era definitivamente un 7.

   a) ¿Cuántas placas diferentes tendrá que verificar la policía?

   b) Si en investigaciones posteriores la policía obtiene además que la matrícula no termina en 53 ni empieza en 78, ¿cuántas comprobaciones se tendrán que hacer en este caso?

6. Un comité de seis personas A, B, C, D, E, F debe escoger un presidente, un secretario y un tesorero. ¿De cuántas formas se puede hacer la elección? ¿De cuántas si el presidente debe ser A ó B? ¿De cuántas si E debe ocupar uno de los cargos? ¿De cuántas si A y F deben ocupar un cargo?

7. De un grupo de 9 personas se quiere elegir un comité con 4 miembros, pero hay dos personas que no podrían estar juntas en él ¿de cuántas formas se puede constituir el comité?

8. Un examen consta de 10 preguntas. Halla de cuántas maneras se puede responder al examen si

   a) hay que contestar al menos 6 preguntas.

   b) hay que contestar al menos 6, de las cuales dos son obligatorias entre las cinco primeras y al menos dos deben ser de las cinco últimas.

9. Halla cuántos números naturales entre mil y cien mil tienen la propiedad de que la suma de sus dígitos es 9 y son todos distintos de cero.

10. Se dispone de una gran cantidad de bolas rojas, azules y verdes. ¿De cuántas formas se puede seleccionar nueve bolas si se debe tener al menos una de cada color?

11. Una clase de 43 estudiantes vota para elegir la fecha de un examen. Cada uno vota por uno de los cinco posibles días. Determina cuántos resultados se pueden obtener en la votación en los siguientes casos:

    a) Cada día obtiene al menos un voto.

    b) Al menos un día recibe más de ocho votos.

12. De un total de 20 personas se deben elegir tres comisiones de 4, 5 y 6 personas respectívamente.

    a) ¿De cuántas maneras es posible formar las comisiones?

    b) Y si cada persona solo puede pertenecer a una comisión.

13. Siete amigos llegan a un hotel y sólo hay disponibles dos habitaciones dobles y una triple. ¿De cuántas maneras pueden repartirse?

14. En el plano XY se consideran caminos que avanzan un paso cada vez (a la derecha o hacia arriba).

    a) ¿Cuántos caminos distintos hay desde $(0, 0)$ a $(7, 7)$?

    b) Cuántos caminos distintos hay desde $(2, 7)$ a $(9, 14)$.

    c) ¿Cuántos de estos caminos pasan por $(4, 10)$?

15. Se consideran los siguientes números: -5, -4, -3, -2, -1, 1, 2, 3, 4. ¿De cuántas maneras se pueden seleccionar cuatro números de modo que su producto sea positivo?

16. Una red de ordenadores está formada por seis equipos. Cada ordenador puede estar conectado a varios equipos o estar desconectado. Demuestra que hay al menos dos ordenadores en la red que tienen el mismo número de conexiones.

17. ¿Cuántas cadenas de ocho bits hay? ¿Cuántas de ellas comienzan por 101? ¿Cuántas de ellas comienzan por 101 o tienen el cuarto bit igual a 1?

18. Halla cuántos enteros no superiores a 100 son primos.

19. Determina el número de enteros positivos menores que 600 que son coprimos con 600.

20. ¿Cuántos números de cuatro cifras tienen al menos un dígito que sea 0, al menos un dígito que sea 1 y al menos un dígito que sea 2?

21. Calcula el número de soluciones de enteros no negativos tiene la ecuación

$$x_1 + x_2 + x_3 + x_4 = 29$$

¿Cuántas de ellas satisfacen $x_1 > 0, \quad x_2 > 2, \quad x_3 < 9, \quad x_4 < 7$?

22. ¿Cuántas permutaciones de las letras de PROGRAMACIÓN no tienen dos letras consecutivas iguales?

23. *a*) ¿En cuántas ordenaciones de la palabra PERIÓDICO aparecen la E y la D juntas?

    b) ¿En cuántas están todas las vocales juntas?

    c) ¿En cuántas no hay dos letras consecutivas iguales?

24. ¿De cuántas formas se pueden disponer los números $1, 2, \ldots, 10$ para que ninguno ocupe su posición natural?

25. Un investigador tiene 5 ayudantes y participa en un proyecto que exige la síntesis de 9 compuestos.

    ¿De cuántas maneras puede el investigador asignar estas síntesis a los 5 ayudantes para que cada uno trabaje al menos en una?

26. Una empresa contrata a once nuevos empleados para sus cuatro oficinas. ¿De cuántas maneras es posible destinarlos? ¿Y si cada oficina debe recibir al menos un nuevo empleado?

27. Un alumno de primer curso se va a examinar de cuatro asignaturas: MD, CC, FP y FF. Dispone de los siete días de una semana durante los cuales repasará todas las asignaturas dedicando cada día al estudio de una única asignatura (sin descansar ningún día).

    a) ¿De cuántas maneras distintas puede organizar su estudio, si se consideran los días indistinguibles?

    b) ¿Y si consideramos que los días son distintos?

28. Se consideran siete pelotas de distintos colores y cuatro recipientes numerados I, II, III, IV.

    a) ¿De cuántas maneras se pueden distribuir las pelotas sin dejar ningún recipiente vacío?

b) Si una de las pelotas es blanca, ¿de cuántas formas podemos hacer la distribución para que no quede ningún recipiente vacío y la pelota blanca esté en el recipiente III?

c) Si se elimina la numeración de los recipientes de modo que no sea posible diferenciarlos, ¿de cuántas formas se pueden distribuir, con la posibilidad de recipiente(s) vacío(s)?

29. Los Reyes Magos traen n juguetes diferentes a n niños. En el camino deciden dejar sin juguete exactamente a un niño y repartir todos los juguetes entre los restantes niños. ¿De cuántas formas pueden hacerlo?

30. Una empresa quiere repartir 100 lápices de memoria entre sus cuatro oficinas de manera que cada una reciba al menos 5, pero no más de 40. Sabiendo que se entregan en paquetes de cinco, ¿de cuántas maneras se puede hacer el reparto?

31. Una empresa de telecomunicaciones desea instalar en Málaga 210 antenas de telefonía móvil y 600 antenas parabólicas de televisión. La ciudad de Málaga se divide en 40 sectores. Para que no queden zonas sin cobertura es necesario que en cada sector haya un mínimo de 10 antenas de televisión y 4 de telefonía. Por otra parte, las ordenanzas municipales impiden colocar más de 7 antenas de telefonía en cada sector. Determina el número de formas distintas de colocar las antenas cumpliendo las restricciones anteriores, sabiendo que las antenas de televisión son indistinguibles entre sí y las de telefonía son también indistinguibles entre sí, pero obviamente se distinguen unas antenas de un tipo de las de otro.

32. Halla una fórmula explícita para las sucesiones de los siguientes apartados:

a) $u_0 = 1$, $u_1 = 2$ y $u_n = -2u_{n-1} + 3u_{n-2}$ para todo $n \geq 2$.

b) $u_0 = 0$, $u_1 = 1$ y $u_n - 5u_{n-1} + 6u_{n-2} = 0$ para todo $n \geq 2$.

c) $u_0 = 1$, $u_1 = 3$ y $u_n - 4u_{n-1} + 4u_{n-2} = 0$ para todo $n \geq 2$.

33. Se sabe que las raíces de la ecuación característica de una recurrencia lineal y homogénea son: -2 triple, -1 doble y 3 simple. ¿De qué orden es la recurrencia? ¿Qué forma tiene la solución general? Justifica las respuestas.

34. Halla una recurrencia lineal homogénea cuyo término general sea

a) $u_n = 3^{n+2} + n3^{n-2}$

b) $u_n = 2^{n+1} + n2^{n-1}$

35. Halla una fórmula explícita para las sucesiones de los siguientes apartados:

a) $u_0 = 0$, $u_n - 2u_{n-1} = 1$ para todo $n \geq 1$,

b) $u_0 = 0$, $u_1 = 2$ y $u_n + 4u_{n-1} + 4u_{n-2} = n^2$ para todo $n \geq 2$.

c) $u_0 = 2$ y $u_n - u_{n-1} = 3n^2$ para todo $n \geq 1$.

d) $u_0 = 1$ y $u_n - 2u_{n-1} = 2^{n-1}$ para todo $n \geq 1$.

e) $u_0 = 0$, $u_1 = 1$ y $u_n + 3u_{n-1} + 2u_{n-2} = 3^{n-2}$ para todo $n \in \mathbb{N}$.

f) $u_0 = 2$ y $u_n - 3u_{n-1} = 5 \cdot 7^n$ para todo $n \geq 1$.

g) $u_0 = 3$, $u_1 = -2$ y $u_n - 4u_{n-1} + 4u_{n-2} = (n-2)2^{n-2}$ para todo $n \geq 0$.

h) $u_0 = 1$, $u_1 = 0$ y $u_n + 6u_{n-1} + 9u_{n-2} = (n-1)3^n$ para todo $n \geq 2$

36. Plantea y resuelve una ecuación de recurrencia para hallar el término general de la sucesión

$$u_n = \sum_{k=1}^{n} k \cdot 2^k = 1 \cdot 2^1 + 2 \cdot 2^2 + 3 \cdot 2^3 + \cdots + n \cdot 2^n$$

37. Se considera la sucesión $\{q_n\}$ del número de cadenas de longitud n que se pueden formar con símbolos del alfabeto $\Sigma = \{0, 1\}$ con la propiedad de que no tienen dos ceros consecutivos. Plantea y resuelve una recurrencia lineal para q_n.

38. Se quiere cubrir un tablero rectangular de tamaño $2 \times n$ usando piezas de tamaño 1×2 y 2×2. Utiliza una sucesión definida recursivamente para para determinar de cuantas maneras se puede cubrir el tablero.

39. Se estacionan motocicletas y turismos en una fila de n espacios. Usa una relación de recurrencia para determinar el número de formas de estacionar dichos vehículos, sabiendo que todos los espacios deben quedar ocupados y que cada motocicleta ocupa un espacio y cada turismo dos. (Se supone que todas las motocicletas son iguales y todos los turismos también).

40. Un muchacho dispone de n monedas para comprar chucherías. Le gustan las palomitas, que cuestan 1 moneda cada bolsa y dos tipos de pasteles que cuestan 2 monedas cada uno. ¿De cuantas maneras se puede gastar las n monedas? (Indicación: distinguir entre n par e impar)

41. Un muchacho se dispone a comprar chucherías en una máquina expendedora. Le gustan las palomitas, que cuestan 1 moneda cada bolsa y dos tipos de pasteles que cuestan 2 monedas cada uno. ¿De cuantas maneras puede sacar las chuches que elija gastando n monedas?

TEMA 3

Relaciones y grafos

3.1. Relaciones binarias

Las *relaciones* en matemáticas formalizan las conexiones entre elementos de varios conjuntos. Son importantes en matemáticas y en computación, tanto por sus aplicaciones teóricas (relaciones de orden, equivalencias,...), como por su aplicaciones más prácticas (bases de datos, redes sociales,...). Constantemente encontramos y manejamos relaciones: ordenación de números, congruencias, divisibilidad, relaciones de parentesco, guías telefónicas, directorios de personal,...

En el tema anterior definimos las relaciones binarias como subconjuntos del producto cartesiano de dos conjuntos. Pero es posible establecer relaciones entre elementos de varios conjuntos, es decir, subconjuntos del producto cartesiano de varios conjuntos:

DEFINICIÓN 3.1.1 (PRODUCTO CARTESIANO DE VARIOS CONJUNTOS) *El* producto cartesiano *de los conjuntos* $A_1, A_2, \ldots, A_n$*, denotado* $A_1 \times A_2 \times \cdots A_n$*, es el conjunto de todas las* n-tuplas ordenadas*:*

$$A_1 \times A_2 \times \cdots \times A_n = \big\{(x_1, x_2, \ldots, x_n); \quad x_j \in A_j, \quad j \in \{1, 2, \ldots, n\}\big\}$$

La denominación n-tupla es genérica, para números concretos, utilizaremos la denominación adecuada: terna, cuádrupla, quíntupla,...

Por otra parte, si los conjuntos del producto son iguales, se puede utilizar una notación abreviada:

$$A \times A = A^2, \qquad A \overset{(n}{\times} \cdots \times A = A^n$$

DEFINICIÓN 3.1.2 (RELACIÓN n-ARIA) *Una* relación n-aria *en los conjuntos* A_1, A_2,..., A_n *es cualquier subconjunto* $\mathcal{R}$ *del producto cartesiano* $A_1 \times A_2 \times \cdots A_n$

$$\mathcal{R} \subseteq A_1 \times A_2 \times \cdots A_n$$

El número n *se denomina igualmente* grado *o* aridad *de la relación.*

La denominación n-aria es genérica, en cada caso particular utilizaremos la denominación específica, como *binaria* para grado dos o *ternaria* para grado tres, aunque para más de tres elementos, es preferible utilizar la palabra grado: grado cuatro, grado cinco,...

En este curso, nos vamos a centrar en relaciones binarias.

Ejemplo 3.1.3

- Si E es el conjunto de los españoles, podemos considerar la relación $\mathcal{M}$ que contiene a todos los matrimonios: $\mathcal{M}$ es una relación binaria.
- Para $A = \mathbb{R}^+$ y $B = \mathbb{R}$, podemos considerar la relación

$$\mathcal{R} = \{(x, y) \in \mathbb{R}^+ \times \mathbb{R} \mid x = y^2\}$$

- Para $A = \mathbb{R}$ y $B = \mathbb{R}$, podemos considerar la relación

$$\mathcal{E} = \left\{(x, y) \in \mathbb{R} \times \mathbb{R} \middle| \frac{x^2}{9} + \frac{y^2}{4} = 1\right\}$$ □

Para las relaciones binarias, lo más habitual es utilizar la notación *infija* para presentar los pares relacionados: si $\mathcal{R} \subseteq A \times B$ y $(x, y) \in \mathcal{R}$, escribiremos $x\mathcal{R}y$, que se lee x está relacionado (mediante $\mathcal{R}$) con y.

Ejemplo 3.1.4 Si $A = \{1, 2, 3\}$, $B = \{a, b, c\}$, podemos definir la relación $\mathcal{R}$ escribiendo:

$$1\mathcal{R}c, \quad 2\mathcal{R}a, \quad 3\mathcal{R}b.$$

Es decir, $\mathcal{R} = \{(1, c), (2, a), (3, b)\}$. □

Ya conocemos y hemos trabajo con distintas relaciones representadas por símbolos conocidos. Por ejemplo, $x \leq y$ indica que el par (x, y) pertenece a la siguiente relación en cualquier conjunto de números:

$$\{(x, y) \mid x \text{ es menor o igual que } y\}$$

En el primer tema, definimos las relaciones de congruencia, que se denotan con el símbolo $\equiv$. También definimos la relación de divisibilidad entre números enteros, que se denota por el símbolo $|$.

Definición 3.1.5 (Dominio y Rango) *Sea $\mathcal{R}$ una relación binaria de A en B. Llamamos* dominio *de $\mathcal{R}$ al conjunto*

$$\mathrm{Dom}(\mathcal{R}) = \{x \in A \mid \textit{existe } y \in B \textit{ tal que } (x, y) \in \mathcal{R}\}$$

Naturalmente, $\mathrm{Dom}(\mathcal{R}) \subseteq A$

Llamamos rango *de* $\mathcal{R}$ *al conjunto*

$$\mathrm{Ran}(\mathcal{R}) = \{x \in B \mid \textit{existe } y \in A \textit{ tal que } (y, x) \in \mathcal{R}\}$$

Naturalmente, $\mathrm{Ran}(\mathcal{R}) \subseteq B$*. El rango se puede denominar igualmente* Imagen*, y denotarse* $\mathrm{Im}(\mathcal{R})$*.*

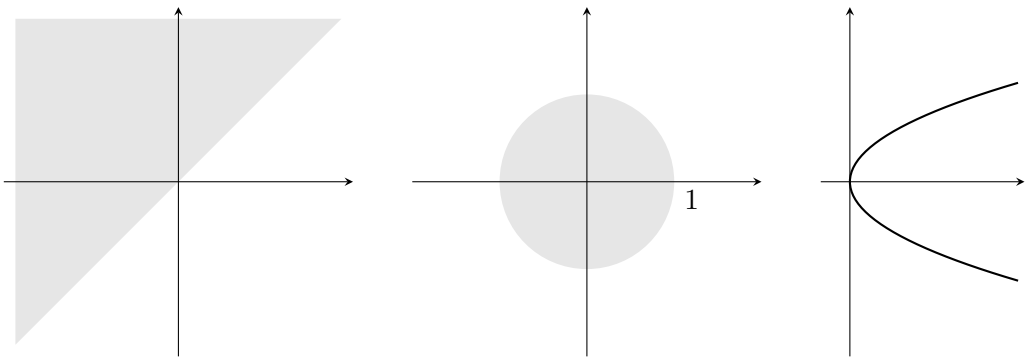
Ejemplo 3.1.6 Para $\mathcal{R} = \{(1,2),(2,3),(3,2),(4,2),(4,5),(5,1)\} \subset \mathbb{N} \times \mathbb{N}$:

$$\mathrm{Dom}(\mathcal{R}) = \{1,2,3,4,5\}, \qquad \mathrm{Ran}(\mathcal{R}) = \{1,2,3,5\}$$

Como vemos, tanto el dominio como el rango pueden estar estrictamente contenidos en los conjuntos en los que se define la relación. □

Representación de relaciones binarias La representación que podamos hacer de una relación depende de los conjuntos entre los que se defina. Por ejemplo, las relaciones entre números pueden representarse gráficamente con regiones del plano.

Ejemplo 3.1.7 Las regiones sombreadas en los siguientes gráficos representan las relaciones de $\mathbb{R}$ en $\mathbb{R}$ que se indican. En el tercer ejemplo, los pares de la relación son los puntos de la parábola.



$x \leq y$ $\qquad$ $x\mathcal{R}y \Leftrightarrow x^2 + y^2 \leq 1$ $\qquad$ $x\mathcal{S}y \Leftrightarrow x = y^2$

En el tema anterior hemos visto que las relaciones y funciones entre conjuntos finitos se pueden representar mediante las *matrices de adyacencia*. Si $\mathcal{R}$ es una relación binaria entre dos conjuntos finitos A y B y $[x_1, x_2, \ldots, x_n]$, $[y_1, y_2, \ldots, y_m]$ son ordenaciones de los elementos de A y B respectivamente, entonces la matriz (de adyacencia) asociada a $\mathcal{R}$ es la matriz $\mathcal{M}_{\mathcal{R}} = (m_{ij})$, de tamaño $n \times m$ (es decir, con n filas y m columnas) dada por:

$$m_{ij} = \begin{cases} 1 & \text{si } (x_i, y_j) \in \mathcal{R} \\ 0 & \text{si } (x_i, y_j) \notin \mathcal{R} \end{cases}$$

Ejemplo 3.1.8 Consideremos la relación $\mathcal{R} \subseteq \{1,2,3,4\} \times \{a,b,c,d\}$:

$$\mathcal{R} = \{(1,d),(3,d),(3,b),(2,a),(4,c)\}$$

Esta relación se puede representar con la tabla

	a	b	c	d
1				×
2	×			
3		×		×
4			×	

La matriz de adyacencia construida utilizando el orden en el que aparecen escritos los elementos en la tabla de arriba es:

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

□

Ejemplo con Maxima 3.1.9 Dado que una relación es un conjunto, podemos utilizar los operadores de `Maxima`, que ya conocemos, para definir nuevas relaciones. Por ejemplo, el operador `is` permite definir relaciones usando predicados binarios:

```
(%i1)  R(x,y) := is(remainder(x-y,3)=0)$
```

$R(x,y)$ es verdadero o "x está relacionado con y por R" si 3 divide a $x-y$.

```
(%i2)  R(-1,3);
```

```
false
```

```
(%i3)  R(2,5);
```

```
true
```

También podemos definir las relaciones como un conjunto de pares.

```
(%i4)  S: {[1,2],[2,4],[3,2],[4,1]}$
```

El operador `elementp` permite convertir la definición anterior en un predicado:

```
(%i5)  Sp(x,y):= elementp([x,y],S)$

(%i6)  Sp(2,4);
```

```
true
```

```
(%i7)  Sp(1,4);
```

```
false
```

E incluso construir la matriz de adyacencia:

```
(%i8)  M_S: genmatrix(lambda([i,j],
            if elementp([i,j],S) then 1 else 0),
            4,4);
```

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

□

3.1.1. Operaciones entre relaciones binarias

Dado que las relaciones son conjuntos, podemos combinar dos o más relaciones con las distintas operaciones entre conjuntos que ya conocemos: unión, intersección, diferencia, complementario y diferencia simétrica.

Ejemplo 3.1.10 Supongamos que A es el conjunto de alumnos de un centro universitario y B el conjunto de libros de la biblioteca. Consideremos la relación $\mathcal{R}$ definida por:

$x\mathcal{R}y$ si y solo si "al estudiante x se le recomienda el libro y"

Y la relación $\mathcal{S}$ definida por:

$x\mathcal{S}s$ si y solo si "el estudiante x utiliza el libro y"

A partir de ellas, construimos las siguientes relaciones.

- $x(R \cup S)y$: "el estudiante x tiene recomendado o utiliza el libro y".
- $x(R \cap S)y$. "el estudiante x tiene recomendado y utiliza el libro y".
- $x(R - S)y$: "el estudiante x tiene recomendo el libro y pero no lo consulta"
- $x(S - R)y$: "el estudiante x utiliza el libro y aunque no lo tiene por recomendado".
- $x\overline{R}y$: "el estudiante x no tiene recomendado el libro y".
- $x\overline{S}y$: "el estudiante x no utiliza el libro y". □

Teorema 3.1.11 *Si $\mathcal{R}$ y $\mathcal{S}$ son relaciones entre los conjuntos finitos A y B con matrices $\mathcal{M}_{\mathcal{R}} = (m_{ij})$ y $\mathcal{M}_{\mathcal{S}} = (m_{ij})$, entonces*

$$\mathcal{M}_{\mathcal{R}\cap\mathcal{S}} = \mathcal{M}_{\mathcal{R}} \wedge \mathcal{M}_{\mathcal{S}} = (m_{ij} \wedge m_{ij})$$
$$\mathcal{M}_{\mathcal{R}\cup\mathcal{S}} = \mathcal{M}_{\mathcal{R}} \vee \mathcal{M}_{\mathcal{S}} = (m_{ij} \vee m_{ij})$$

Es decir, para calcular la matriz de la intersección y unión de relaciones, multiplicamos y sumamos las matrices respectivamente, considerando el producto y suma booleanos elemento a elemento.

Aparte de estas operaciones conjuntistas podemos definir otras operaciones específicas sobre relaciones.

Definición 3.1.12 (Relación inversa) *Sea $\mathcal{R}$ una relación de A en B. Se llama* relación inversa *de $\mathcal{R}$ a la relación de B en A definida por:*

$$\mathcal{R}^{-1} = \{(x, y) \in B \times A \mid (y, x) \in \mathcal{R}\}$$

Obsérvese que $\mathrm{Dom}(\mathcal{R}^{-1}) = \mathrm{Ran}(\mathcal{R})$ y $\mathrm{Ran}(\mathcal{R}^{-1}) = \mathrm{Dom}(\mathcal{R})$. Además, si $\mathcal{R}$ es una relación entre conjuntos finitos, podemos calcular fácilmente la matriz de adyacencia de su inversa según establece el siguiente resultado.

Teorema 3.1.13 *Si $\mathcal{R}$ es una relación entre los conjuntos finitos A y B con matriz de adyacencia $\mathcal{M}_{\mathcal{R}} = (m_{ij})$, entonces la matriz de la relación inversa es la transpuesta de $\mathcal{M}_{\mathcal{R}}$:*

$$\mathcal{M}_{\mathcal{R}^{-1}} = \mathcal{M}_{\mathcal{R}}^t$$

Ejemplo 3.1.14 La relación inversa de

$$\mathcal{R} = \{(a, 1), (a, 2), (c, 4), (d, 3)\} \subset \{a, b, c, d\} \times \{1, 2, 3, 4\}$$

es

$$\mathcal{R}^{-1} = \{(1, a), (2, a), (4, c), (3, d)\} \subset \{1, 2, 3, 4\} \times \{a, b, c, d\}$$

Observamos además que

$$\mathcal{M}_{\mathcal{R}} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}; \qquad \mathcal{M}_{\mathcal{R}^{-1}} = \mathcal{M}_{\mathcal{R}}^t = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \qquad \square$$

Definición 3.1.15 (Composición de relaciones) *Sean los conjuntos A, B y C y las relaciones binarias*

$$\mathcal{R} \subseteq A \times B \quad y \quad \mathcal{S} \subseteq B \times C$$

La composición *de las relaciones $\mathcal{R}$ y $\mathcal{S}$ es la relación de A en C definida por:*

$$\mathcal{R} \circ \mathcal{S} = \{(x, y) \in A \times C \mid \textit{existe } z \in B \textit{ tal que } (x, z) \in \mathcal{R}, (z, y) \in \mathcal{S}\}$$

Es decir,

$$x(\mathcal{R}\circ\mathcal{S})y \iff \text{Existe } z\in B \text{ tal que } x\mathcal{R}z,\quad z\mathcal{S}y$$

TEOREMA 3.1.16 *Si A, B y C son conjuntos finitos, $\mathcal{R}$ es una relación de A en B y $\mathcal{S}$ una relación de B en C, entonces*

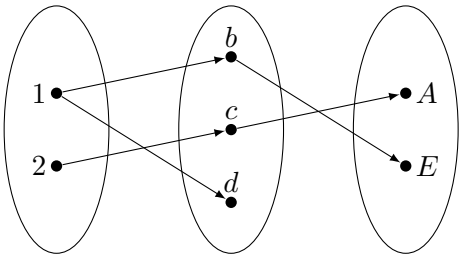
$$\mathcal{M}_{\mathcal{R}\circ\mathcal{S}} = \mathcal{M}_{\mathcal{R}} \odot \mathcal{M}_{\mathcal{S}}$$

en donde $(m_{ij})\odot(n_{ij}) = \Big(\bigvee_k (m_{ik}\wedge n_{kj})\Big)$.

EJEMPLO 3.1.17 Consideremos las relaciones

$$\mathcal{R} = \{(1,b),(1,d),(2,c)\} \subseteq \{1,2\}\times\{b,c,d\}$$
$$\mathcal{S} = \{(b,E),(c,A)\} \subseteq \{b,c,d\}\times\{A,E\}$$

La representación de estas relaciones con diagramas de Venn es la siguiente:



Las conexiones entre el primer y el tercer conjunto determinan la relación de composición, es decir, 1 está relacionado con E y 2 está relacionado con A. Vamos a calcular esta composición utilizando sus matrices

$$\mathcal{M}_{\mathcal{R}\circ\mathcal{S}} = \mathcal{M}_{\mathcal{R}} \odot \mathcal{M}_{\mathcal{S}} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \odot \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Por lo tanto,

$$\mathcal{R}\circ\mathcal{S} = \{(1,E),(2,A)\}$$ □

EJEMPLO CON MAXIMA 3.1.18 En `Maxima`, podemos trabajar con las constantes booleanas `true` y `false`, y el sistema dispone de varios operadores sobre estos valores. Sin embargo, es más simple y práctico trabajar con los números 0 y 1 y definir las operaciones booleanas sobre ellos. Solo necesitamos definir la "suma booleana", ya que el "producto booleano" coincide con el producto numérico. Vamos a utilizar la secuencia "+b" escrita de forma infija para representar la suma booleana:

```
(%i1) infix("+b")$
(%i2) m +b n := m+n-m*n$
```

De esta forma, podremos usar el operador "+b" también para operar elemento a elemento matrices booleanas .

```
(%i3) 1 +b 1;
```

$$1$$

```
(%i4) matrix([1,0],[0,1]) +b matrix([1,0],[1,0]);
```

$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

También podemos redefinir el producto estándar de matrices numéricas para que opere con la suma y producto booleano tal y como hemos visto anteriormente. Para ello, `Maxima` dispone de dos constantes con las que podemos establecer cual debe ser la suma y cual el producto en el producto de matrices: con `matrix_element_add` podemos establecer la suma y con `matrix_element_mult` el producto. En este caso, solo necesitamos cambiar la suma, puesto que sí estamos usando el producto numérico.

```
(%i5) matrix_element_add:
        lambda ([[x]], lreduce ("+b", x))$
```

A partir de aquí, el operador producto matricial (representado por un punto bajo) se comportará como el producto de matrices booleanas. Obsérvese que hemos tenido que recurrir al operador `lreduce`, puesto que "+b" se ha definido para dos argumentos y en el producto matricial se aplicará a una cantidad arbitraria de sumandos.

```
(%i6) matrix([1,0,1],[0,1,0]).
matrix([0,1],[1,0],[0,0]);
```

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$ □

3.1.2. Relaciones binarias en un conjunto

En el caso particular de relaciones definidas de un conjunto en sí mismo, es importante estudiar las propiedades básicas que tales relaciones pueden verificar.

DEFINICIÓN 3.1.19 *Sea $\mathcal{R}$ una relación binaria en A (es decir, de A en A).*

- $\mathcal{R}$ *se dice* reflexiva *si* $x\mathcal{R}x$ *para todo* $x \in A$.
- $\mathcal{R}$ *se dice* simétrica *si para todo* $x, y \in A$*: si* $x\mathcal{R}y$*, entonces* $y\mathcal{R}x$
- $\mathcal{R}$ *se dice* transitiva *si para todo* $x, y, z \in A$*: si* $x\mathcal{R}y$*,* $y\mathcal{R}z$*, entonces* $x\mathcal{R}z$.

- $\mathcal{R}$ *se dice* antisimétrica *si para todo* $x, y \in A$*: si* $x\mathcal{R}y$*,* $y\mathcal{R}x$*, entonces* $x = y$*. Equivalentemente,* $\mathcal{R}$ *es* antisimétrica *si para todo* $x, y \in A$*: si* $x \neq y$*, entonces o bien* $x\not\mathcal{R}y$*, o bien* $y\not\mathcal{R}x$
- $\mathcal{R}$ *se dice* conexa *si para todo* $x, y \in A$*, o bien* $x\mathcal{R}y$*, o bien* $y\mathcal{R}x$

Estas definiciones pueden enunciarse mediante operaciones conjuntistas.

TEOREMA 3.1.20 *Sea* $\mathcal{R}$ *una relación binaria en* A *(es decir, de* A *en* A*).*

1. $\mathcal{R}$ *es reflexiva si y solo* $\mathcal{I}_A \subseteq \mathcal{R}$*, en donde* $\mathcal{I}_A$ *es la* relación identidad*:*
$$\mathcal{I}_A = \{(x, x); x \in A\}$$
2. $\mathcal{R}$ *es simétrica si y solo si* $\mathcal{R}^{-1} = \mathcal{R}$
3. $\mathcal{R}$ *es transitiva si y solo si* $\mathcal{R}^2 \subseteq \mathcal{R}$*.*
4. $\mathcal{R}$ *es antisimétrica si y solo si* $\mathcal{R} \cap \mathcal{R}^{-1} \subseteq \mathcal{I}_A$
5. $\mathcal{R}$ *es conexa si y solo si* $\mathcal{R} \cup \mathcal{R}^{-1} = A \times A$*.*

Para relaciones entre conjuntos finitos, podemos estudiar estas propiedades utilizando sus matrices de adyacencia. En el siguiente resultado, utilizamos la notación $\mathcal{I}_n$ para representar la matriz $n \times n$ en la cual los elementos de la diagonal principal son iguales a 1 y el resto son iguales a 0. También vamos a utilizar la siguiente relación entre matrices: $\mathcal{M} = (m_{ij})$ es menor o igual que $\mathcal{N} = (n_{ij})$ si $m_{ij} \leq n_{ij}$ para cada posición ij de las matrices, y lo denotamos por $\mathcal{M} \leq \mathcal{N}$.

TEOREMA 3.1.21 *Sea* $\mathcal{R}$ *una relación en un conjunto finito* A *y* $\mathcal{M}_\mathcal{R} = (m_{ij})_{n\times n}$ *su matriz de adyacencia. Entonces:*

1. $\mathcal{R}$ *es reflexiva si y solo si* $m_{ii} = 1$ *para todo* i*, es decir, si todos los elementos de la diagonal principal son iguales a 1.*
2. $\mathcal{R}$ *es simétrica si y solo si* $m_{ij} = m_{ji}$ *para todo* i, j*, es decir, si la matriz es simétrica:* $\mathcal{M}_\mathcal{R} = \mathcal{M}_\mathcal{R}^{-1}$*.*
3. *Si* $\mathcal{M}_{\mathcal{R}^2} = (n_{ij})$*, entonces* $\mathcal{R}$ *es transitiva si y solo si* $n_{ij} \leq m_{ij}$ *para todo* i, j*; es decir, si* $\mathcal{M}_\mathcal{R}^2 \leq \mathcal{M}_\mathcal{R}$*.*
4. $\mathcal{R}$ *es antisimétrica si y solo si* $m_{ij} \wedge m_{ji} = 0$ *para todo* i, j *tales que* $i \neq j$*, es decir, si en cada pareja de elementos simétricos, hay al menos un cero. Matricialmente: es antisimétrica si y solo si* $\mathcal{M}_R \wedge \mathcal{M}_R^t \leq \mathcal{I}_n$
5. $\mathcal{R}$ *es conexa si y solo si* $m_{ij} \vee m_{ji} = 1$ *para todo* i, j*. Matricialmente: es conexa si y solo si* $\mathcal{M}_R \vee \mathcal{M}_R^t = (1)_{n\times n}$

La siguiente tabla resume las definiciones de las propiedades y sus caracterizaciones en forma matricial.

	Definición	Matricialmente (Solo finitas)
Reflexiva	$x\mathcal{R}x$	$\mathcal{I} \leq \mathcal{M}_R$
Simétrica	$x\mathcal{R}y \implies y\mathcal{R}x$	$\mathcal{M}_R^t = \mathcal{M}_R$
Transitiva	$(x\mathcal{R}y, y\mathcal{R}z) \implies x\mathcal{R}z$	$\mathcal{M}_R^2 \leq \mathcal{M}_R$
Antisimétrica	$(x\mathcal{R}y, y\mathcal{R}x) \implies x = y$	$\mathcal{M}_R \wedge \mathcal{M}_R^t \leq \mathcal{I}$
Conexa	$x\not\mathcal{R}y \implies y\mathcal{R}x$	$\mathcal{M}_R \vee \mathcal{M}_R^t = (1)_{n\times n}$

Ejemplo con Maxima 3.1.22 Vamos a estudiar las propiedades de la siguiente relación.

```
(%i1)  S: {[1,2],[2,4],[3,2],[4,1]}$
(%i2)  MS: genmatrix(lambda([i,j],
                  if elementp([i,j],S) then 1 else 0),
                  4,4);
```

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Por ejemplo, la relación no es simétrica, puesto que no lo es la matriz:

```
(%i3)  is(MS=transpose(MS));
```

```
false
```

Como hemos visto anteriormente, para estudiar algunas propiedades, necesitamos algunas matrices auxiliares. Por ejemplo, la matriz identidad, cuyos elementos en la diagonal principal son iguales a 1 y el resto son iguales a 0:

```
(%i4)  ident(4);
```

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

También necesitamos las matrices cuadradas con todos lo elementos iguales a 1, que podemos definir fácilmente:

```
(%i5)  unos(n) := genmatrix(lambda([i,j],1),n,n)$
(%i6)  unos(4);
```

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

La relación S es antisimétrica, ya que el producto booleano elemento a elemento de $\mathcal{M}_S$ por $\mathcal{M}_S^t$, es menor que la matriz diagonal.

```
(%i7) MS*transpose(MS);
```

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

```
(%i8) is(%=%*ident(4));
```

```
true
```

Para comprobar la propiedad transitiva, tenemos que usar el producto matricial considerando la suma y el producto booleano.

```
(%i9) infix("+b")$
(%i10) m +b n := m+n-m*n$
(%i11) matrix_element_add:
         lambda ([[x]], lreduce ("+b", x))$
(%i12) MS.MS;
```

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Por lo tanto, la relación S no es transitiva.

```
(%i13) is(%=%*MS);
```

```
false
```

En estas comprobaciones hemos tenido en cuenta que el producto booleano coincide con la función mínimo y que $\min(a, b) = a$ si y solamente si $a \leq b$.

Finalmente, también podemos comprobar que la relación no es conexa.

```
(%i14) is (MS +b transpose(MS)=unos(4));
```

```
false
```

□

Definición 3.1.23 (Relación de orden parcial) *Una relación binaria $\mathcal{R}$ en un conjunto no vacío A se dice que es una relación de* orden parcial *si es reflexiva, antisimétrica y transitiva. Habitualmente, lo escribimos diciendo que el par $(A, \mathcal{R})$ es un* conjunto parcialmente ordenado.

Si una relación solo tiene las propiedades reflexiva y transitiva, se dice que es un *preorden*. Si una relación de orden es además conexa, se dice que la relación es de *orden total*. Las relaciones de orden sirven comparar magnitudes y establecer preferencias.

La relación de orden habitual entre números es efectivamente una relación de orden y además es total. En el tema anterior hemos estudiado la relación de divisibilidad, que es también una relación de orden parcial entre números naturales. La relación de inclusión entre conjuntos, es otro ejemplo de relación de orden parcial.

Habitualmente, las relaciones de orden se representan por símbolos que recuerdan a la relación de orden entre números, como por ejemplo $\preceq$ o $\sqsubseteq$.

3.1.3. Relaciones de equivalencia

Definición 3.1.24 *Una relación $\mathcal{R}$ en A se dice que es una* relación de equivalencia *si es reflexiva, simétrica y transitiva.*

Las relaciones de equivalencia sirven para introducir nociones de igualdad basada en determinadas características. Por ejemplo, las relaciones de congruencia que hemos estudiado en el tema anterior son relaciones de equivalencia. Otros ejemplo de relación de equivalencia es la semejanza de triángulos.

Habitualmente, las relaciones de equivalencia se representan por símbolos que recuerdan el símbolo de igualdad, como por ejemplo $\equiv$, $\sim$, $\approx$.

La noción de clase de equivalencia se ha introducido en el tema anterior para el ejemplo de las relaciones de congruencia y la extendemos ahora a cualquier relación de equivalencia.

Definición 3.1.25 (Clases de equivalencia) *Sea $\sim$ una relación de equivalencia definida en un conjunto A y sea $x \in A$.*

- *El subconjunto de A formado por los elementos equivalentes a x se llama* clase de equivalencia *de x y se denota $[x]_\sim$.*

$$[x]_\sim = \{y \in A \mid y \sim x\}$$

- *El conjunto formado por las clases de equivalencia de $\sim$ se denomina* conjunto cociente *y se denota $A_{/\sim}$:*

$$A_{/\sim} = \{[x]_\sim; x \in A\}$$

Cada elemento $y \in [x]_\sim$ puede ser considerado como *representante* de esta clase de equivalencia. Por ejemplo, para la relación de congruencia módulo 3 en $\mathbb{Z}$

$$x \equiv_3 y \iff x - y \text{ es multiplo de 3,}$$

el conjunto cociente está formado por las tres clases de congruencia

$$\mathbb{Z}_3 = \mathbb{Z}_{/\equiv_3} = \{[0]_3, [1]_3, [2]_3\}$$

Según vimos en el tema anterior, las clases de congruencia determinan una *partición* del conjunto de números enteros; esto ocurre en cualquier conjunto y para cualquier relación de equivalencia.

Definición 3.1.26 *Una* partición *de un conjunto S es una familia de subconjuntos, $\mathcal{P} = \{S_1, ..., S_k\}$ tales que:*

- *$S_i \cap S_j = \varnothing$ para todo i, j (es decir, son* disjuntos dos a dos*).*
- *$S_1 \cup S_2 \cup \cdots \cup S_k = S$*

Ejemplo 3.1.27

- $\mathcal{P} = \{\{1,3\}, \{2,5\}, \{4\}\}$ es una partición del conjunto $S = \{1,2,3,4,5\}$.
- $\mathcal{P} = \{\{x \in \mathbb{N} \mid x \text{ es par }\},\ \{x \in \mathbb{N} \mid x \text{ es impar }\}\}$ es una partición de $\mathbb{N}$. □

Teorema 3.1.28 *Si $\sim$ es una relación de equivalencia en A, entonces el conjunto cociente, $A_{/\sim}$ es una partición de A.*

Teorema 3.1.29 *Si $\mathcal{P}$ es un partición de A, entonces existe una relación de equivalencia en A tal que $A_{/\sim} = \mathcal{P}$.*

La demostración de estos resultados es bastante simple. Por ejemplo, para el segundo de ellos, basta probar que la relación

$$x \sim_{\mathcal{P}} y \iff \text{Existe } S \in \mathcal{P} \text{ tal que } x, y \in S$$

tiene como conjunto cociente la partición $\mathcal{P}$. La relación $\sim_{\mathcal{P}}$ se denomina *relación inducida* por $\mathcal{P}$.

Ejemplo 3.1.30 La relación en $A = \{1,2,3,4,5,6\}$ inducida por la partición

$$A_1 = \{1,3,5\},\ A_2 = \{4,6\},\ A_3 = \{2\}$$

es la siguiente

$$\begin{aligned}\mathcal{R}_{\mathcal{P}} = \{&(1,1),(1,3),(1,5),(3,1),(3,3),(3,5),(5,1),(5,3),(5,5),\\ &(4,4),(4,6),(6,4),(6,6),\\ &(2,2)\}\end{aligned}$$ □

Ejemplo con Maxima 3.1.31 El operador `equiv_classes`(A,R) determina las clases de equivalencia en el conjunto A dadas por la relación R definida con un predicado.

```
(%i1) conj: setify(makelist(i,i,-10,10));
```

$$\{-10,-9,-8,-7,-6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6,7,8,9,10\}$$

```
(%i2) R(x,y) := is(remainder(x-y,3)=0)$
(%i3) equiv_classes(conj,R);
```

$$\{\{-10,-7,-4,-1,2,5,8\},\{-9,-6,-3,0,3,6,9\},\{-8,-5,-2,1,4,7,10\}\}$$ □

3.1.4. Cierres de relaciones

Dada una relación $\mathcal{R}$ buscamos la mínima relación que la contiene y que verifica una o varias propiedades básicas.

Definición 3.1.32 *Sea $\mathcal{R}$ una relación binaria en A. El* cierre reflexivo *de $\mathcal{R}$ denotado por $\mathrm{r}(\mathcal{R})$ es la menor relación reflexiva que contiene a $\mathcal{R}$. Es decir, es la única relación reflexiva que verifica que $\mathcal{R}\subseteq \mathrm{r}(\mathcal{R})$ y $\mathrm{r}(\mathcal{R})\subseteq \mathcal{S}$ para toda relación reflexiva $\mathcal{S}\supseteq\mathcal{R}$.*

Obviamente, una relación es reflexiva si y solo si coincide con su cierre reflexivo. Por otra parte, es muy sencillo determinar el cierre reflexivo de una relación.

Teorema 3.1.33 *Si $\mathcal{R}$ una relación binaria en A, entonces $\mathrm{r}(\mathcal{R})=\mathcal{R}\cup\mathcal{I}_A$, en donde $\mathcal{I}_A=\{(x,x);x\in A\}$.*

Definición 3.1.34 *Sea $\mathcal{R}$ una relación binaria en A. El* cierre simétrico *de $\mathcal{R}$, denotado por $\mathrm{s}(\mathcal{R})$, es la menor relación simétrica que contiene a $\mathcal{R}$. Es decir, $\mathrm{s}(\mathcal{R})$ es la única relación simétrica tal que $\mathcal{R}\subseteq \mathrm{s}(\mathcal{R})$ y $\mathrm{s}(\mathcal{R})\subseteq\mathcal{S}$ para toda relación simétrica $\mathcal{S}\supseteq\mathcal{R}$.*

Obviamente, una relación es simétrica si y solo si coincide con su cierre simétrico. Por otra parte, es muy sencillo determinar el cierre simétrico de una relación.

Teorema 3.1.35 *Si $\mathcal{R}$ una relación binaria en A, entonces $\mathrm{s}(\mathcal{R})=\mathcal{R}\cup\mathcal{R}^{-1}$.*

Ejemplo 3.1.36 Sea el conjunto $A=\{1,2,3\}$ y $\mathcal{R}=\{(1,2),(2,3)\}$ Entonces

- $\mathrm{r}(\mathcal{R})=\{(1,2),(2,3),(1,1),(2,2),(3,3)\}$. Ya que solo necesitamos añadir los pares $(1,1)$, $(2,2)$ y $(3,3)$ para obtener una relación reflexiva.

- $\mathrm{s}(\mathcal{R}) = \{(1,2),(2,3),(2,1),(3,2)\}$. Ya que solo necesitamos añadir los pares opuestos de los que forman $\mathcal{R}$. □

DEFINICIÓN 3.1.37 *Sea $\mathcal{R}$ una relación binaria en A. El* cierre transitivo *de $\mathcal{R}$, denotado por $\mathrm{t}(\mathcal{R})$, es la menor relación transitiva que contiene a $\mathcal{R}$. Es decir, $\mathrm{t}(\mathcal{R})$ es la única relación transitiva tal que $\mathcal{R} \subseteq \mathrm{t}(\mathcal{R})$ y $\mathrm{t}(\mathcal{R}) \subseteq \mathcal{S}$ para toda relación transitiva $\mathcal{S} \supseteq \mathcal{R}$.*

Obviamente, una relación es transitiva si y solo si coincide con su cierre transitivo. El siguiente resultado establece un método para determinar el cierre transitivo de una relación.

TEOREMA 3.1.38 *Sea $\mathcal{R}$ una relación binaria en A. Entonces*

$$\mathrm{t}(\mathcal{R}) = \mathcal{R} \cup \mathcal{R}^2 \cup \mathcal{R}^3 \cup \cdots = \bigcup_{i=1}^{\infty} \mathcal{R}^i$$

COROLARIO 3.1.39 *Si $\mathcal{R}$ es una relación binaria en un conjunto finito A, entonces*

$$\mathrm{t}(\mathcal{R}) = \bigcup_{i=1}^{n} \mathcal{R}^i$$

siendo n el número de elementos de A.

EJEMPLO 3.1.40 Consideremos el conjunto $A = \{a,b,c,d\}$ y la relación

$$\mathcal{R} = \{(a,a),(a,d),(b,d),(c,a),(d,b)\}$$

Vamos a calcular el cierre transitivo de $\mathcal{R}$.

Utilizando la ordenación $[a,b,c,d]$ de los elementos de A, la matriz de adyacencia de $\mathcal{R}$ es

$$\mathcal{M}_{\mathcal{R}} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

De esta forma, podemos calcular fácilmente las matrices de $\mathcal{R}^2$, $\mathcal{R}^3$ y $\mathcal{R}^4$.

$$\mathcal{M}_{\mathcal{R}^2} = \mathcal{M}_{\mathcal{R}}^2 = \begin{pmatrix} 1&0&0&1\\0&0&0&1\\1&0&0&0\\0&1&0&0 \end{pmatrix} \begin{pmatrix} 1&0&0&1\\0&0&0&1\\1&0&0&0\\0&1&0&0 \end{pmatrix} = \begin{pmatrix} 1&1&0&1\\0&1&0&0\\1&0&0&1\\0&0&0&1 \end{pmatrix}$$

$$\mathcal{M}_{\mathcal{R}^3} = \mathcal{M}_{\mathcal{R}}^3 = \begin{pmatrix} 1&1&0&1\\0&1&0&0\\1&0&0&1\\0&0&0&1 \end{pmatrix} \begin{pmatrix} 1&0&0&1\\0&0&0&1\\1&0&0&0\\0&1&0&0 \end{pmatrix} = \begin{pmatrix} 1&1&0&1\\0&0&0&1\\1&1&0&1\\0&1&0&0 \end{pmatrix}$$

$$\mathcal{M}_{\mathcal{R}^4} = \mathcal{M}_{\mathcal{R}}^4 = \begin{pmatrix} 1&1&0&1\\0&0&0&1\\1&1&0&1\\0&1&0&0 \end{pmatrix} \begin{pmatrix} 1&0&0&1\\0&0&0&1\\1&0&0&0\\0&1&0&0 \end{pmatrix} = \begin{pmatrix} 1&1&0&1\\0&1&0&0\\1&1&0&1\\0&0&0&1 \end{pmatrix}$$

Por lo tanto,

$$\mathcal{M}_{\mathrm{t}(\mathcal{R})} = \mathcal{M}_{\mathcal{R}} \vee \mathcal{M}_{\mathcal{R}^2} \vee \mathcal{M}_{\mathcal{R}^3} \vee \mathcal{M}_{\mathcal{R}^4} =$$

$$= \begin{pmatrix} 1&0&0&1\\0&0&0&1\\1&0&0&0\\0&1&0&0 \end{pmatrix} \vee \begin{pmatrix} 1&1&0&1\\0&1&0&0\\1&0&0&1\\0&0&0&1 \end{pmatrix} \vee \begin{pmatrix} 1&1&0&1\\0&0&0&1\\1&1&0&1\\0&1&0&0 \end{pmatrix} \vee \begin{pmatrix} 1&1&0&1\\0&1&0&0\\1&1&0&1\\0&0&0&1 \end{pmatrix} =$$

$$= \begin{pmatrix} 1&1&0&1\\0&1&0&1\\1&1&0&1\\0&1&0&1 \end{pmatrix}$$

Es decir, $\mathrm{t}(\mathcal{R}) = \{(a,a), (a,b), (a,d), (b,b), (b,d), (c,a), (c,b), (c,d), (d,b), (d,d)\}$. □

Algoritmo de Warshall. Para calcular cada elemento de las matrices $\mathcal{M}_{\mathcal{R}^i}$ en el método del teorema anterior, tenemos que realizar $2n-1$ operaciones booleanas. De esta forma, para calcular cada elemento de $\mathcal{M}_{\mathrm{t}(\mathcal{R})}$ tenemos que realizar $2n(n-1)$ operaciones. Por lo tanto, el cálculo de $\mathcal{M}_{\mathrm{t}(\mathcal{R})}$ requiere $2n^3(n-1)$ operaciones.

El teorema siguiente describe el *Algoritmo de Warshall* para calcular el cierre transitivo de una relación. Este algoritmo reduce el número de operaciones necesarias.

Teorema 3.1.41 (Algoritmo de Warshall) *Dado un conjunto A con n elementos y una relación $\mathcal{R}$ en A, determinamos el cierre transitivo de $\mathcal{R}$ calculando una secuencia de matrices*

$$\mathcal{W}_0, \mathcal{W}_1, \cdots, \mathcal{W}_k, \cdots, \mathcal{W}_n$$

de forma que $\mathcal{W}_0 = \mathcal{M}_{\mathcal{R}}$, $\mathcal{W}_n = \mathcal{M}_{\mathrm{t}(\mathcal{R})}$ y si $\mathcal{W}_k = (w_{ij}^{(k)})$ entonces

$$w_{ij}^{(k)} = w_{ij}^{(k-1)} \vee (w_{ik}^{(k-1)} \wedge w_{kj}^{(k-1)})$$

Para calcular $w_{ij}^{(k)}$ a partir de matriz $\mathcal{W}_{k-1}$ necesitamos 2 operaciones booleanas y por lo tanto, para hallar $\mathcal{W}_k$ a partir de $\mathcal{W}_{k-1}$ necesitamos $2n^2$ operaciones. Por lo tanto, dado que calculamos n matrices, necesitamos $2n^3$ operaciones.

A la hora de calcular la secuencia de matrices en el algoritmo de Warshall, es conveniente tener en cuenta las siguientes observaciones:

- $w_{ij}^{(k-1)} \leq w_{ij}^{(k)}$ y por lo tanto, los posiciones iguales a 1 en $\mathcal{W}_{k-1}$ se mantienen en $\mathcal{W}_k$.
- Teniendo en cuenta las siguientes igualdades

$$w_{kj}^{(k)} = w_{kj}^{(k-1)} \vee (w_{kk}^{(k-1)} \wedge w_{kj}^{(k-1)}) = w_{kj}^{(k-1)}$$
$$w_{ik}^{(k)} = w_{ik}^{(k-1)} \vee (w_{ik}^{(k-1)} \wedge w_{kk}^{(k-1)}) = w_{ik}^{(k-1)}$$

  deducimos que la fila k-ésima y la columna k-ésima de $\mathcal{W}_k$ son iguales a las de $\mathcal{W}_{k-1}$.

Ejemplo 3.1.42 Consideremos el conjunto $A = \{a, b, c, d\}$ y la relación

$$\mathcal{R} = \{(a, a), (a, d), (b, d), (c, a), (d, b)\}$$

Vamos a calcular el cierre transitivo de $\mathcal{R}$ utilizando el algoritmo de Warshall.

$$\mathcal{W}_0 = \mathcal{M}_{\mathcal{R}} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\mathcal{W}_1 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & \boxed{1} \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

En la matriz $\mathcal{W}_1$ copiamos la primera fila y la primera columna de $\mathcal{W}_0$. Recorremos sus elementos y vemos que $w_{31}^{(1)} = 1$ y $w_{14}^{(1)} = 1$, por lo que $w_{34}^{(1)} = 1$. El resto de elementos de la matriz se copian de la matriz anterior.

$$\mathcal{W}_2 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & \boxed{1} \end{pmatrix}$$

En la matriz $\mathcal{W}_2$ copiamos la segunda fila y la segunda columna de $\mathcal{W}_1$. Recorremos sus elementos y vemos que $w_{42}^{(2)} = 1$ y $w_{24}^{(2)} = 1$, por lo que $w_{44}^{(2)} = 1$. El resto de elementos de la matriz se copian de la matriz anterior.

$$\mathcal{W}_3 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

En la matriz $\mathcal{W}_3$ copiamos la tercera fila y la tercera columna de $\mathcal{W}_2$. Dado que la columna tercera esta compuesta solamente por ceros, en este paso no añadimos ningún uno más, por lo que el resto de elementos de la matriz se copian de la matriz anterior.

$$\mathcal{W}_4 = \mathcal{M}_{\mathrm{t}(\mathcal{R})} = \begin{pmatrix} 1 & \boxed{1} & 0 & 1 \\ 0 & \boxed{1} & 0 & 1 \\ 1 & \boxed{1} & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

En la matriz $\mathcal{W}_4$ copiamos la cuarta fila y la cuarta columna de $\mathcal{W}_3$. Recorremos sus elementos y vemos que $w_{14}^{(4)} = 1$ y $w_{42}^{(4)} = 1$, por lo que $w_{12}^{(2)} = 1$; $w_{24}^{(4)} = 1$ y $w_{42}^{(4)} = 1$, por lo que $w_{22}^{(2)} = 1$; $w_{34}^{(4)} = 1$ y $w_{42}^{(4)} = 1$, por lo que $w_{32}^{(2)} = 1$. El resto de elementos de la matriz se copian de la matriz anterior. □

Ejemplo con Maxima 3.1.43 Aunque `Maxima` no incluye ningún operador o paquete específico para el algoritmo de Warshall, no es difícil escribir una función recursiva a partir de la descripción del algoritmo.

```
(%i1) warshall(m):= warshall_aux(m,length(m),1)$
(%i2) warshall_aux(m,n,k):= if k=n+1 then m else
          warshall_aux (genmatrix(lambda([i,j],
                        max(m[i][j],min(m[i][k],m[k][j]))),
                        n,n),n,k+1)$
```

Para utilizar recursión de cola, recurrimos a un operador auxiliar con dos argumentos adicionales, el tamaño de la matriz y un contador para las iteraciones del algoritmo. Hemos utilizado el operador `genmatrix`$(a[i,j],k,m)$, que construye una matriz de tamaño $k \times m$ con el elemento $a[i,j]$ en la posición (i,j). Para describir los elementos de la matriz, recurrimos al operador `lambda`$([i,j],<$ expresión en i, j $>)$, que permite dar la expresión que corresponde a cada posición sin necesidad de asignarle un nombre. De esta forma, es fácil observar que si $m = \mathcal{M} = \mathcal{W}_0$, entonces `warshall_aux`$(m,,k) = \mathcal{W}_k$.

Vamos a verificar el ejemplo que hemos hecho más arriba, "trazando" el operador auxiliar para verificar las etapas intermedias.

```
(%i3) M0: matrix([1,0,0,1],
                 [0,0,0,1],
                 [1,0,0,0],
                 [0,1,0,0])$
(%i4) trace(warshall_aux)$
(%i15)  warshall(M0);


1 Introducir warshall_aux [matrix(
[1,0,0,1],
```

```
[0,0,0,1],
[1,0,0,0],
[0,1,0,0]
),4,1]
 .2 Introducir warshall_aux [matrix(
[1,0,0,1],
[0,0,0,1],
[1,0,0,1],
[0,1,0,0]
),4,2]
 ..3 Introducir warshall_aux [matrix(
[1,0,0,1],
[0,0,0,1],
[1,0,0,1],
[0,1,0,1]
),4,3]
 ...4 Introducir warshall_aux [matrix(
[1,0,0,1],
[0,0,0,1],
[1,0,0,1],
[0,1,0,1]
),4,4]

 ....5 Introducir warshall_aux [matrix(
[1,1,0,1],
[0,1,0,1],
[1,1,0,1],
[0,1,0,1]
),4,5]
 ....5 Salir  warshall_aux matrix(
[1,1,0,1],
[0,1,0,1],
[1,1,0,1],
[0,1,0,1]
..............
)
```

$$\begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

□

Para calcular el cierre de equivalencia basta realizar los tres cierres correspondientes a las propiedades que lo definen, pero hay que tener en cuenta que el cierre transitivo siempre debe ser el último.

TEOREMA 3.1.44 *Sea $\mathcal{R}$ una relación binaria definida en un conjunto A. Entonces, la mínima relación de equivalencia que contiene a $\mathcal{R}$ es* $\mathrm{t}(\mathrm{s}(\mathrm{r}(\mathcal{R})))$. *Es decir,*

$$\mathrm{t}(\mathcal{R} \cup \mathcal{R}^{-1} \cup \mathcal{I}_A)$$

3.2. Grafos

Los **grafos** son modelos matemáticos utilizados para representar relaciones entre objetos de un conjunto, y que permiten generalizar el modelo de las relaciones binarias que estudiamos en la lección anterior. Utilizamos grafos para estudiar conexiones entre objetos, datos o fuentes de información. Un ejemplo de grafo lo encontramos en las redes de carreteras, en donde las poblaciones se enlazan o conectan con una o varias carreteras. Otros ejemplos son las redes de comunicaciones, redes eléctricas, redes de carreteras, pero también redes sociales y árboles genealógicos.

Algunos problemas que habitualmente se estudian y resuelven desde la teoría de grafos son: calcular el número de combinaciones diferentes de vuelos entre dos ciudades de una red aérea; determinar las posibles rutas entre dos localizaciones de una ciudad o región; encontrar el camino más corto entre dos ciudades en una red de transporte; programar actividades en un calendario; asignar canales a las emisoras de televisión; determinar si se puede crear un circuito en una placa de una sola capa; construir modelos para redes informáticas, determinar si dos ordenadores están conectados entre sí.

3.2.1. Grafos simples

DEFINICIÓN 3.2.1 *Un **grafo simple** es un par $G = \langle V, E\rangle$ formado por un conjunto V, cuyos elementos denominaremos **vértices** de G, y*

$$E \subseteq \Big\{\{u, v\}; u, v \in V, u \neq v\Big\};$$

*los elementos de E se denominan **aristas** de G.*

Los vértices de un grafo pueden ser objetos (ordenadores, ciudades, personas,...) o datos (fechas, direcciones, módulos de un programa, identificadores,...) y las aristas pueden ser conexiones físicas (carreteras, conexiones de red,...) o conexiones virtuales (dependencias, herencias, conflictos,...).

EJEMPLO 3.2.2 Consideremos el grafo $G = \langle V, E\rangle$ determinado por:

$$V = \{a, b, c, d\}, \qquad E = \Big\{\{a, b\}, \{a, d\}, \{b, c\}, \{b, d\}, \{c, d\}\Big\}.$$

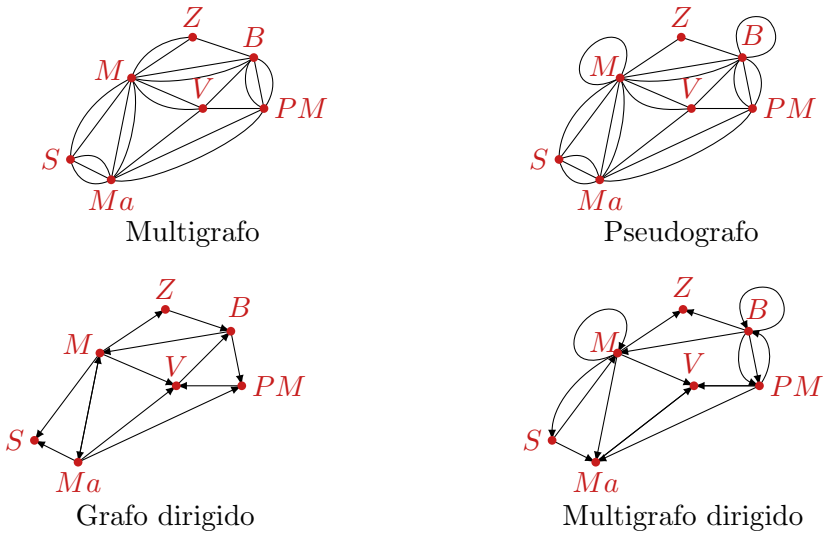
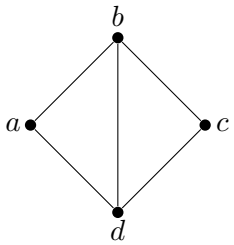


Figura 3.1: Tipos de grafos que no son simples.

Más adelante, aprenderemos a trabajar con diferentes representaciones formales de los grafos, pero en grafos con pocos vértices es conveniente utilizar igualmente su representación gráfica.



Los vértices se representan como puntos en el plano y las aristas se representan por líneas uniendo vértices. □

Los grafos simples son el tipo más sencillo de grafo y es un modelo formal que coincide con las *relaciones binarias simétricas y antirreflexivas* (es decir, ningún elemento está relacionado consigo mismo) en un conjunto. Existen varias posibles generalizaciones de la noción de grafo simple (ver figura 3.1). En los **multigrafos**, se admiten múltiples aristas conectando dos vértices; en los **pseudografos**, se permite conectar un vértice consigo mismo; en los **grafos dirigidos**, los pares que determinan las aristas son ordenados; en los **multigrafos dirigidos**, las aristas son pares ordenados y puede haber multiples aristas entre vértices; en los **grafos ponderados**, las conexiones determinadas por las aristas tiene asignadas pesos.

3.2.2. Conceptos y resultados básicos

DEFINICIÓN 3.2.3 *Si $G = (V, E)$ es un grafo simple, $u, v \in V$, $e = \{u, v\} \in E$, decimos que*

- *los vértices u y v son **adyacentes**,*
- *que la arista e es **incidente** con los vértices u y v,*
- *que los vértices u y v son **extremos** de la arista e,*
- *que la arista e **conecta** a los vértices u y v.*

DEFINICIÓN 3.2.4 *Sea $G = \langle V, E\rangle$ un grafo simple. Definimos la función **grado**, $\delta\colon V \to \mathbb{N}$, como: $\delta(v)$ es el número de aristas incidentes en v. Si $\delta(v) = 0$, decimos que v es un vértice **aislado**; si $\delta(v) = 1$, decimos que v es una **hoja**. Decimos que G es **regular** si todos sus vértices tienen el mismo grado.*

Enunciamos a continuación el teorema de Euler, en el que utilizamos las siguiente notación: si A es un conjunto finito, $|A|$ denota el número de elementos de A.

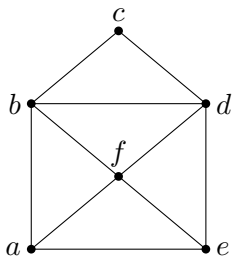
TEOREMA 3.2.5 (EULER) *Si $G = \langle V, E\rangle$ es un grafo simple, entonces*

$$\sum_{v \in V} \delta(v) = 2|E|.$$

En particular, la suma de los grados de todos los vértices de un grafo simple es un número par, de donde se deduce el siguiente resultado.

COROLARIO 3.2.6 *Todo grafo simple tiene un número par de vértices de grado impar.*

EJEMPLO 3.2.7 Consideremos el grafo



Entonces:

$$\delta(a) = 3, \quad \delta(b) = 4, \quad \delta(c) = 2, \quad \delta(d) = 4, \quad \delta(e) = 3, \quad \delta(f) = 4$$

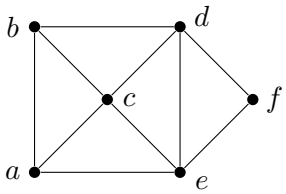
Por lo que efectivamente se verifica el teorema de Euler en este grafo:

$$\sum_{v\in V}\delta(v) = \delta(a)+\delta(b)+\delta(c)+\delta(d)+\delta(e)+\delta(f) =$$
$$= 3+4+2+4+3+4 = 20 = 2\cdot 10 = 2|E|$$

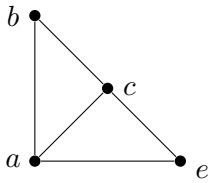
Además, hay exactamente dos vértices, a y e, que tienen grados impares. □

Definición 3.2.8 (Subgrafos) *Se dice que el grafo $H = \langle V_H, E_H\rangle$ es un **subgrafo** del grafo $G = \langle V_G, E_G\rangle$ si $V_H \subseteq V_G$ y $E_H \subseteq E_G$. Decimos que es **subgrafo propio** si, o bien $V_H \neq V_G$, o bien $E_H \neq E_G$. Si $V_H = V_G$, se dice que H es un subgrafo **generador** (en inglés, se denomina **spanning subgraph**). del grafo G.*

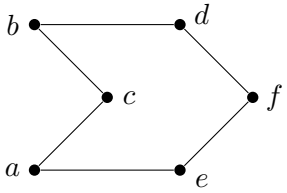
Ejemplo 3.2.9 Los grafos que aparecen abajo verifican que tanto H_1 como H_2 son subgrafos propios de G. En H_1 hay menos vértices, y por lo tanto, menos aristas. Los vértices de H_2 son los mismo que los de G, pero en H_2 hay menos aristas, es decir, H_2 es un subgrafo generador de G.



G



H_1



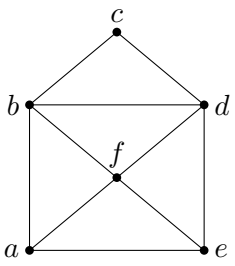
H_2

□

3.2.3. Representación de grafos simples

La representación gráfica que hemos usado en ejemplos anteriores ayuda a entender el concepto de grafo y a visualizar propiedades y operaciones sobre ellos. Si el número de vértices o el número de aristas es muy grande, esta representación carecerá de utilidad práctica. Por eso son necesarias otras formas de representación que permitan la descripción fácil de un grafo, con independencia de su tamaño, y sobre las cuales podamos realizar transformaciones y estudiar propiedades de forma efectiva y algorítmica.

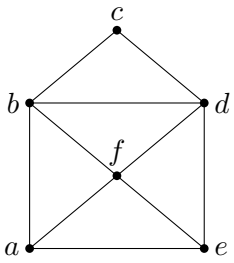
Dado que un grafo simple es una relación binaria, podemos representarlo por las matrices de adyacencia que estudiamos en el tema anterior y como vemos en el siguiente ejemplo.



$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Recordemos que es necesario fijar previamente un orden en el conjunto de vértices, por lo que no será necesario especificar la correspondencia de filas y columnas con los vértices. Por otra parte, dado que el grafo simple es una relación simétrica, estas matrices siempre serán simétricas. De la misma forma, dado que el grafo simple es una relación antirreflexiva, la diagonal principal solo contiene ceros.

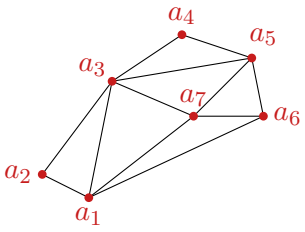
Otra forma de describir un grafo es mediante las **listas de adyacencia**. Estas listas son en realidad una tabla en cuya primera fila se disponen todos los vértices del grafo y, por debajo de cada vértice formando una columna, los vértices adyacentes a él.



a	b	c	d	e	f
b	a	b	b	a	a
e	c	d	c	d	b
f	d		e	f	d
	f		f		e

Aunque no es necesario, es conveniente establecer previamente un orden dentro del conjunto de vértices (igual que hacemos en las matrices de adyacencia) y utilizarlo al disponer los vértices en la primera fila y en cada columna.

Ejemplo 3.2.10 Mostramos a continuación un grafo dado por su representación gráfica, por su lista de adyacencia y por su matriz de adyacencia, en la cual, se ha elegido el orden de los vértices dado por sus subíndices.



a_1	a_2	a_3	a_4	a_5	a_6	a_7
a_2	a_1	a_1	a_3	a_3	a_1	a_1
a_3	a_3	a_2	a_5	a_4	a_5	a_3
a_6		a_4		a_6	a_7	a_5
a_7		a_5		a_7		a_6
		a_7				

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

□

Ejemplo con Maxima 3.2.11 `Maxima` dispone de un paquete con operadores específicos para trabajar con grafos. Podemos trabajar con grafos simples, dirigidos o

no dirigidos, y también con grafos ponderados. Internamente los grafos son representados por sus listas de adyacencia, aunque también podemos trabajar con la matriz de adyacencia. Los vértices son identificados con números naturales y las aristas son representadas por listas de longitud dos. Se pueden asignar etiquetas a vértices y se pueden asignar pesos a las aristas para definir grafos ponderados. El operador `create_graph` sirve para definir una grafo a partir de su lista de vértices y su lista de aristas. Este operador admite varias sintaxis; en la más simple, escribimos un primer argumento con un número positivo n, que será el número de vértices y los vértices serán identificados con los números 0, 1, 2,..., $n-1$. El segundo argumento es una lista de listas de longitud 2 que definen las aristas del grafo.

```
(%i1) load(graphs)$
(%i2) g1: create_graph(5,[[1,2],[1,3],[2,3],[0,4]]);
(%o2) GRAPH(5 vertices, 4 edges)
```

Como vemos, la salida no muestra nada, solo nos da la información del grafo que hemos definido, concretamente el número de vértices y el número de aristas. Para ver su representación como lista de adyacencia usamos `print_graph`:

```
(%i3) print_graph(g1);
Graph on 5 vertices with 4 edges.
Adjacencies:
  4 :  0
  3 :  2  1
  2 :  3  1
  1 :  3  2
  0 :  4
(%o3)   done
```

Obsérvese que la muestra como columna, no como fila. También podemos obtener la representación mediante la matriz de adyacencia del grafo:

```
(%i4) mg1: adjacency_matrix(g1);
```

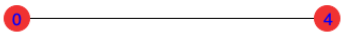
$$\begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

A lo largo del tema iremos viendo distintos operadores para estudiar los grafos y obtener información sobre ellos. Por ejemplo, en la sección anterior, hemos definido la noción de grado de un vértice y la lista de adyacencia no permite visualizar ese grado, pero un elemento importante del grafo es lo que se llama la *secuencia gráfica*, la lista de los grados de todos los vértices ordenada de forma creciente:

```
(%i5) degree_sequence(g1);
(%o15)   [1,1,2,2,2]
```

Finalmente, el operador `draw_graph` implementa un algoritmo para construir una representación gráfica.

```
(%i6) draw_graph(g1, vertex_size=4,show_id=true);
```



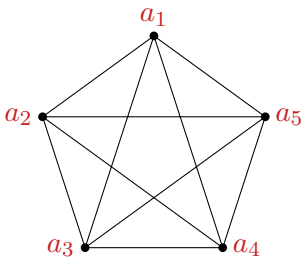
El operador `draw_graph` dispone de varias opciones para configurar y mejorar la representación gráfica. En este caso, hemos utilizado "show_id=true" para que se muestre la etiqueta de cada vértice y "vertex_size= 4" para aumentar el tamaño del círculo que los encierra.

También podemos crear un grafo a partir de su matriz de adyacencia.

```
(%i7) mat: matrix([0,0,1,1,1],[0,0,1,0,1],[1,1,0,0,0],
                  [1,0,0,0,0],[1,1,0,0,0])$
(%i7) g2: from_adjacency_matrix(mat);
(%o7) GRAPH(5 vertices, 5 edges)
(%i8) print_graph(g2);
Graph on 5 vertices with 5 edges.
Adjacencies:
  4 :  1  0
  3 :  0
  2 :  1  0
  1 :  4  2
  0 :  4  3  2
(%o8) done
```

Como vemos, los vértices se etiquetan con los números consecutivos desde el 0, siguiendo el orden dado por las filas y columnas. □

EJEMPLO 3.2.12 El grafo **completo** K_n es el grafo de n vértices en el que cualquier par de vértices está conectado por una arista. Por ejemplo, el grafo K_5 es el que mostramos a continuación:



$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Obsérvese que todos los elementos de la matriz son iguales a 1 excepto los de la diagonal principal.

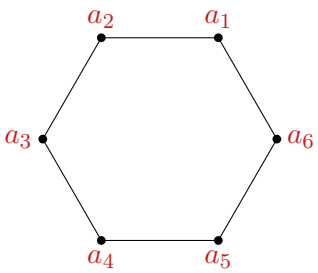
Por otra parte, para hacer el dibujo del grafo, hemos necesitado que las aristas "se corten" en puntos que no corresponden a vértices. Como veremos más adelante, esto es inevitable en algunos grafos y por eso es conveniente remarcar claramente los puntos del dibujo que corresponden con vértices.

En `Maxima`, estos grafos están predefinidos:

```
(%i1) load(graphs)$
(%i2) k5: complete_graph(5)$
(%i3) draw_graph(k5);
```

Nos muestra el grafo completo de 5 vértices. □

EJEMPLO 3.2.13 El **ciclo** C_n es un grafo de n vértices que están conectados formando una cadena cerrada. Vemos a continuación el ciclo C_6:



$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

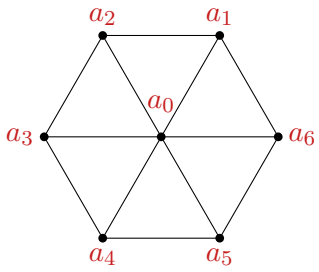
Obsérvese que hemos obtenido esa matriz de adyacencia porque los vértices se han ordenado siguiendo el recorrido circular sobre el ciclo.

En `Maxima`, estos grafos están predefinidos:

```
(%i1) load(graphs)$
(%i2) c6: cycle_graph(6)$
(%i3) draw_graph(c6);
```

Nos muestra el ciclo de 6 vértices. □

EJEMPLO 3.2.14 La **rueda** W_n es un grafo de $n+1$ que se obtiene añadiendo un vértice al ciclo C_n que está conectado con todos sus vértices. Mostramos a continuación el ejemplo W_6



$$\left(\begin{array}{c|cccccc} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{array}\right)$$

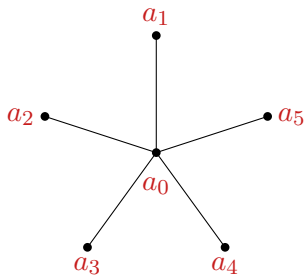
Hemos añadido una primera fila y una primera columna a la matriz de adyacencia de C_6 que corresponden al vértice central de la rueda.

En `Maxima`, estos grafos están predefinidos:

```
(%i1) load(graphs)$
(%i2) w6: wheel_graph(6)$
(%i3) draw_graph(w6);
```

Nos muestra la rueda de 7 vértices. □

EJEMPLO 3.2.15 La **estrella** S_n es un grafo de $n+1$ vértices que se obtiene eliminando del grafo W_n las aristas correspondientes al ciclo C_n. Vemos a continuación el ejemplo S_5:

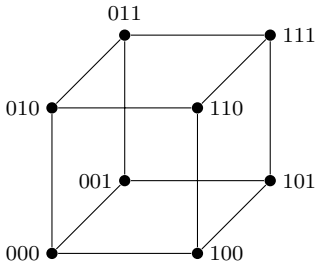


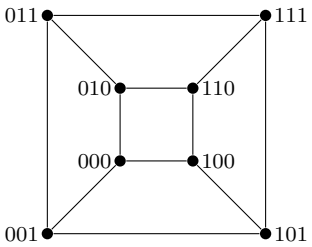
$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

□

EJEMPLO 3.2.16 El n**-cubo**, es el grafo simple $Q_n = (V_n, E_n)$ definido como sigue. Los vértices son las cadenas binarias de longitud n, es decir, $V = \{0,1\}^n$, que está formado por 2^n vértices. Las aristas son pares de cadenas que se diferencian en un solo elemento; por ejemplo, $\{010, 110\} \in E_3$, pero $\{010, 111\} \notin E_3$.

Representamos a continuación el grafo Q_3 con dos dibujos. Uno con la forma de cubo que justifica el nombre de este tipo de grafos y otro en el que las aristas se trazan sin cortarse.





En `Maxima`, estos grafos están predefinidos:

```
(%i1) load(graphs)$
(%i2) cube4: cube_graph(4)$
(%i3) draw_graph(cube4);
```

Nos muestra un cubo de dimensión 4, que se forma a partir de listas de longitud cuatro, tiene 16 vértices y 32 aristas. □

DEFINICIÓN 3.2.17 (GRAFOS BIPARTITOS) *Un grafo $G = \langle V, E\rangle$ se dice que es **bipartito** si $V = V_1 \cup V_2$, $V_1 \cap V_2 = \varnothing$, ningún par de vertices en V_1 está conectado y ningún par de vertices en V_2 está conectado.*

EJEMPLO CON MAXIMA 3.2.18 En `Maxima` disponemos de un predicado que analiza si un grafo es bipartito, `is_bipartite` y un operador que determina las dos partes de un grafo si es bipartito, `bipartition`.

- El grafo C_6 (ciclo de longitud 6) es bipartito:

$$V_1 = \{a_1, a_3, a_5\}, \qquad V_2 = \{a_2, a_4, a_6\}$$

```
(%i1) load(graphs)$
(%i2) is_bipartite(cycle_graph(6));
(%o2) true
(%i3) bipartition(cycle_graph(6));
(%o3) [[4,2,0],[5,3,1]]
```

- El grafo S_5 (estrella de 5 puntas) también es bipartito:

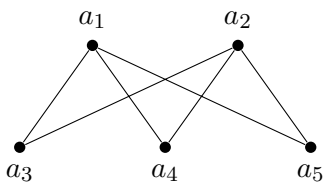
$$V_1 = \{a_0\}, \qquad V_2 = \{a_1, a_2, a_3, a_4, a_5\}$$

- Sin embargo, el grafo W_6 (rueda de 7 vértices) no es bipartito.

```
(%i5) is_bipartite(wheel_graph(6));
(%o5) false
(%i6) bipartition(wheel_graph(6));
(%o6) []
```

□

Ejemplo con Maxima 3.2.19 El grafo **bipartito completo** $K_{m,n}$ es el grafo con $m+n$ vértices en donde el conjunto de vértices está partido en dos conjuntos V_1 con m vértices y V_2 con n vértices, de tal forma que cada vértice de V_1 está conectado con cada vértice de V_2. Vemos abajo el grafo bipartito completo $K_{3,2}$:



$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Para escribir la matriz de adyacencia, hemos ordenado los vértices según su subíndice, los dos primeros corresponden a una parte de los vértices y los tres últimos a la otra parte. De esta forma, la matriz ha quedado formada por cuatro cajas que contienen o bien unos o bien ceros.

El operador `complete_bipartite_graph` define en `Maxima` operadores bipartitos completos del tamaño que deseemos. La siguiente línea siguiente mostrará el grafo del ejemplo anterior.

```
(%i1) load(graphs)$
(%i2) draw_graph(complete_bipartite_graph(3,2),
          vertex_size=3,show_id=true);
```

□

3.2.4. Conexión en grafos

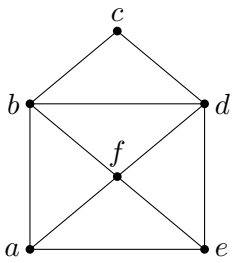
En esta sección vamos a introducir distintos conceptos y propiedades sobre la conexión de vértices en un grafo que se establece recorriendo una o varias aristas.

Definición 3.2.20 *Un **camino** (en inglés, se dice **walk**) entre los vértices v_0 y v_k de un grafo $G = (V, E)$, es una secuencia finita de vértices, no necesariamente distintos,*

$$C = v_0 v_1 v_2 \dots v_k$$

tal que $e_i = \{v_{i-1}, v_i\} \in E$ para todo $i = 1 \dots k$. En este caso, decimos que el camino C ***recorre*** *las aristas e_1, e_2,...,e_k y pasa por los vértices v_0, v_1,... v_k. El vértice v_0 se denomina* ***vértice inicial*** *de C y v_k se denomina* ***vértice final*** *de C. El número natural k es la longitud de C, es decir, el número de aristas que recorre.*

Ejemplo 3.2.21 La secuencia $C = abfdefbc$ es un camino del grafo

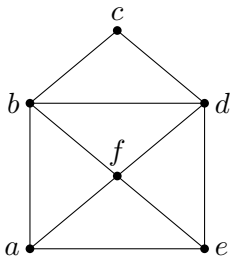


□

Definición 3.2.22 *Sea $C = v_0v_1v_2 \dots v_k$ un camino en un grafo $G = (V, E)$ y sea $e_i = \{v_{i-1}, v_i\}$ para cada $i = 1 \dots k$.*

- *C se dice que es* ***simple*** *(en inglés, se denomina* ***trail****) si $e_i \neq e_j$ para cada cada $i \neq j$; es decir, cada arista se recorre una única vez.*
- *C se dice que es* ***elemental*** *(en inglés, un camino elemental se denomina* ***path****), si $v_i \neq v_j$, para cada $i \neq j$; es decir por cada vértice se pasa a lo sumo una vez.*
- *C es un camino* ***cerrado*** *si $v_0 = v_k$, es decir, si empieza y termina en el mismo vértice.*
- *El camino C se dice que es un* ***circuito*** *si es un camino cerrado que no repite aristas.*
- *El camino C se dice que es un* ***ciclo*** *si es un camino cerrado en el que todos los vértices son distintos.*

Ejemplo 3.2.23 Consideramos el siguiente grafo simple



- El camino $C_1 = abdcbfe$ es simple.

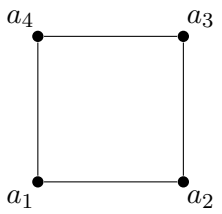
- El camino $C_2 = abfde$ es elemental
- El camino $C_3 = abdcba$ es cerrado.
- El camino $C_4 = abfdbcdea$ es un circuito.
- El camino $C_5 = abfdea$ es un ciclo. □

DEFINICIÓN 3.2.24 *Un grafo $G = (V, E)$ se dice* ***conexo*** *si hay un camino entre cada par de vértices distintos del grafo.*

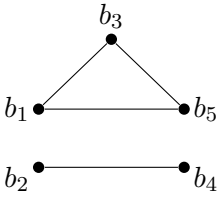
EJEMPLO 3.2.25 Consideremos los grafos G y H dados por las siguientes matrices de adyacencia:

$$M_G = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \qquad M_H = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Mientras que G es un grafo conexo, H no lo es. Evidentemente, no es fácil deducir estas afirmaciones a partir de las matrices, pero puede serlo si observamos sus representaciones gráficas.



G



H

En ambos casos, la ordenación de los vértices para la construcción de las matrices está dado por por los subíndices. □

En el ejemplo anterior, observamos que el grafo H, que no es conexo, es unión de dos subgrafos conexos y tales que no tienen aristas entre los vértices de uno y los vértices del otro. Estos subgrafos conexos disjuntos se denominan **componentes conexas** del grafo. En particular, un grafo es conexo si y solo si tiene exactamente una componente conexa.

EJEMPLO CON MAXIMA 3.2.26 En `Maxima`, el operador `is_conneceted` analiza si un grafo es o no conexo y `connected_components` nos devuelve las componentes conexas.

```
(%i1) load(graphs)$
(%i2) g1: create_graph(5,[[1,2],[1,3],[2,3],[0,4]])$
(%i3) is_connected(g1);
(%o3) false
(%i4) connected_components(g1);
(%o4) [[2,1,3],[0,4]]
(%i5) g2: create_graph([1,2,3,4,5,6,7,8], [[1,2], [1,6],
        [1,8],[1,3], [2,4],[2,5],[2,7],[3,4],[3,5],[3,7],
        [4,6],[4,8],[5,6],[5,8],[6,7],[7,8]])$
(%i6) is_connected(g2);
(%o6) true
(%i7) connected_components(g2);
(%o7) [[3,8,6,7,5,4,2,1]]
```

□

El siguiente resultado nos dice como calcular el número de caminos de una determinada longitud entre dos vértices, y nos dará un método para estudiar si un grafo es conexo a partir de su matriz de adyacencia.

TEOREMA 3.2.27 *Sea G un grafo con matriz de adyacencia M_G y sea $M_G^k = (m_{ij})$ la potencia k-ésima, calculada utilizando la suma y el producto en $\mathbb{N}$, de la matriz M_G. Entonces, m_{ij} es el número de caminos de longitud k entre los vértices v_i y v_j.*

EJEMPLO 3.2.28 Volvamos a considerar los grafos del ejemplo 3.2.25:

$$M_G = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \qquad M_H = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Empezamos estudiando G y calculamos M_G^2:

$$M_G^2 = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 \\ 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 \end{pmatrix}$$

El número 2 de la posición $(1,3)$ en la matriz M_G^2, significa que hay dos caminos de longitud 2 que conectan los vértices a_1 y a_3; estos caminos son $C_1 = a_1a_2a_3$ y $C_2 = a_1a_4a_3$. Dado que cada una de las posiciones en una matriz 4×4 es distinta de cero en la matriz M_G^2 o en la matriz M_G, podemos deducir que el grafo es conexo, ya que cada par de vértices (determinado por cada posición en la matriz) están conectados por al menos un camino de longitud menor o igual que 2.

Calculemos ahora las potencias sucesivas de M_H.

$$M_H = \begin{pmatrix} 0&0&1&0&1\\ 0&0&0&1&0\\ 1&0&0&0&1\\ 0&1&0&0&0\\ 1&0&1&0&0 \end{pmatrix}; \qquad M_H^2 = \begin{pmatrix} 2&0&1&0&1\\ 0&1&0&0&0\\ 1&0&2&0&1\\ 0&0&0&1&0\\ 1&0&1&0&2 \end{pmatrix};$$

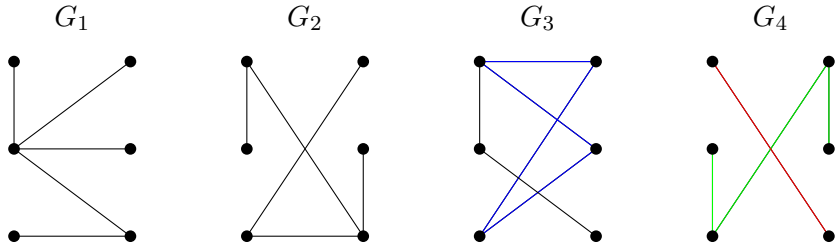
$$M_H^3 = \begin{pmatrix} 2&0&3&0&3\\ 0&0&0&1&0\\ 3&0&2&0&3\\ 0&1&0&0&0\\ 3&0&3&0&2 \end{pmatrix}; \qquad M_H^4 = \begin{pmatrix} 6&0&5&0&5\\ 0&1&0&0&0\\ 5&0&6&0&5\\ 0&0&0&1&0\\ 5&0&5&0&6 \end{pmatrix}$$

Observamos entonces que hay doce posiciones que son nulas en las cuatro matrices, las que determinan el número de caminos entre los vértices b_1 y b_2, entre los vértices b_1 y b_4, entre los vértices b_2 y b_5, entre los vértices b_4 y b_5 y entre los vértices b_3 y b_4. Podemos deducir que H no es conexo, ya que cuatro es el número de aristas del grafo y si un grafo es conexo, se debería poder conectar cada par de vértices con un número menor o igual que el total de las aristas. □

3.2.5. Árboles

DEFINICIÓN 3.2.29 *Un **árbol** es un grafo conexo sin ciclos.*

Por ejemplo, los grafos G_1 y G_2 son árboles, pero G_3 y G_4 no:



El grafo G_3 no es un árbol porque contiene un ciclo, mientras que G_4 no lo es porque no es conexo.

El matemático Arthur Cayley fue quien usó árboles por primera vez en 1857 para representar ciertos tipos de componentes químicos. En ciencias de la computación los árboles nos sirven para: almacenar y recuperar información; construir algoritmos eficientes para hacer búsquedas; construir códigos eficientes para almacenar y transmitir datos; modelar procedimientos que se realizan usando una secuencia de decisiones.

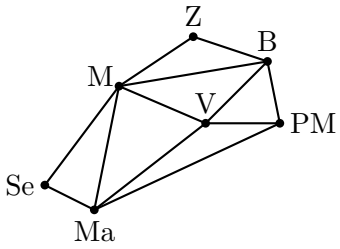
TEOREMA 3.2.30 *Sea $G = (V, E)$ un grafo simple. Los siguientes enunciados son equivalentes:*

1. *G es un árbol.*
2. *G es conexo y $|E| = |V| - 1$.*
3. *G no tiene ciclos y $|E| = |V| - 1$.*
4. *En G hay exactamente un camino entre cada par de vértices.*
5. *G es conexo, pero si eliminamos una arista cualquiera, el grafo resultante no es conexo.*

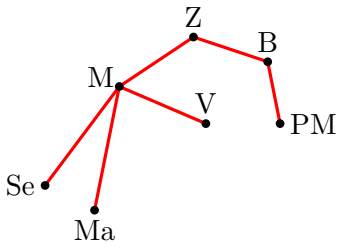
Los árboles tienen propiedades de minimalidad, en el sentido de que son los grafos conexos con el menor número posible de aristas. Por esa razón, nos sirven para modelizar redes de comunicación con un mínimo número de conexiones entre nodos.

DEFINICIÓN 3.2.31 *Un **árbol generador** (en inglés, se dice **spanning tree**) de un grafo simple G es un subgrafo generador de G que además es un árbol; es decir, es un subgrafo que es árbol y contiene todos los vértices de G.*

EJEMPLO 3.2.32 El grafo de la derecha en un árbol generador del grafo G de la izquierda:

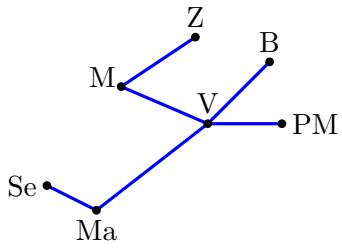


G



Árbol generador de G

Un grafo puede tener más de un árbol generador; por ejemplo, abajo aparece otro árbol generador del mismo grafo G.



□

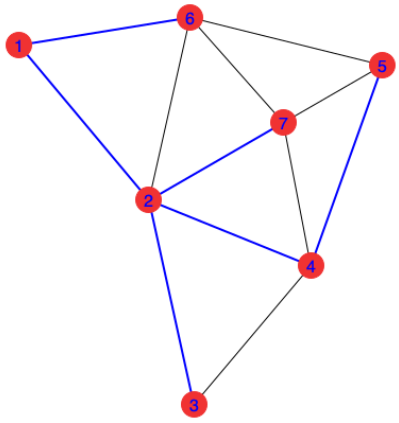
Los árboles generadores se pueden usar para estudiar propiedades de los grafos y definir sobre ellos algoritmos y funciones.

Ejemplo con Maxima 3.2.33 En `Maxima`, el operador `is_tree` analiza si un grafo es o no árbol.

```
(%i1) load(graphs)$
(%i2) is_tree(cube_graph(3));
(%o2) false
(%i3) arb: create_graph(8,[[0,5],[0,6],[1,3],[2,4],[2,6],[2,7],[3,6]])$
(%i4) is_tree(arb);
(%o4) true
```

Podemos determinar árboles generadores de un grafo, concretamente aquellos que contienen el menor número de aristas; para ello usamos el operador `minimun_spanning_tree`.

```
(%i5) gt: create_graph([1,2,3,4,5,6,7],
        [[1,2],[1,6],[2,3],    [2,4],  [2,6],  [2,7],  [3,4],
        [4,5],  [4,7],  [5,6],  [5,7],  [6,7]])$
(%i6) sgt: minimum_spanning_tree(gt)$
(%i7) draw_graph(gt, vertex_size=4,
        show_id=true, show_edges=edges(sgt));
(%o8) done
```



En el código anterior hemos usado la opción `show_edges=edges(sgt)` para que se resalten las aristas del subgrafo `sgt`, es decir, del subárbol generador minimal. □

Teorema 3.2.34 *Un grafo simple G es conexo si y sólo si tiene un árbol generador T.*

Demostración: Por definición, un árbol es conexo y si es subárbol de otro grafo, necesariamente este también será conexo. Para demostrar que un grafo conexo contiene un árbol generador, vamos a probar que podemos construirlo siguiendo el siguiente proceso. Si G no es un árbol, entonces contiene un ciclo; si eliminamos una

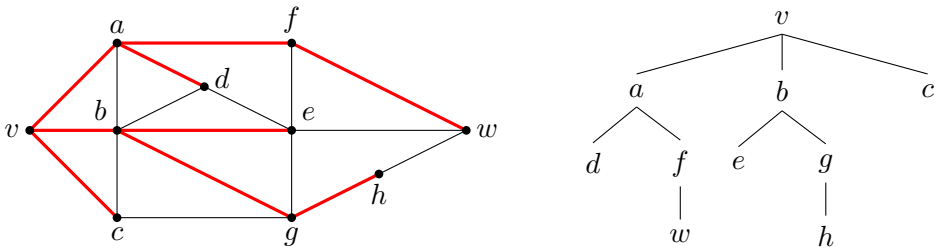
arista de este ciclo, estamos construyendo un subgrafo generador. En este subgrafo repetimos el análisis; si no es un árbol, entonces contiene un ciclo y eliminando una de sus aristas construimos un nuevo subgrafo. Para construir el árbol generador, solo tendremos que repetir el proceso eliminando tantas aristas como sea necesario. □

El algoritmo que hemos utilizado en la demostración anterior nos permite determinar fácilmente un árbol generador. Vemos a continuación otros métodos de generación de estos árboles mediante lo que se denominan árboles de búsqueda.

Búsqueda en anchura. El siguiente algoritmo construye un árbol generador siguiendo un recorrido en el grafo mediante **búsqueda en anchura**. La búsqueda en anchura es adecuada para resolver problemas de optimización, en los que se deba elegir la mejor solución entre varias posibles. También los usamos, por ejemplo, para colorear grafos bipartitos o para encontrar el camino más corto en un grafo entre dos vértices.

1. Elegimos un vértice arbitrario v_0, que llamaremos **raíz**.
2. Añadimos todas las aristas incidentes en v_0.
3. Para cada uno de los vértices conectados con los vértices añadido en el paso anterior, añadimos todas las aristas que inciden en ellos si el otro vértice que conectan no se había añadido ya al árbol.
4. Con los nuevos vértices, procedemos de la misma forma hasta añadir todos los vértices de árbol.

Ejemplo 3.2.35 Empezando en el vértice v, vamos a construir un árbol generador del siguiente grafo usando búsqueda en anchura.



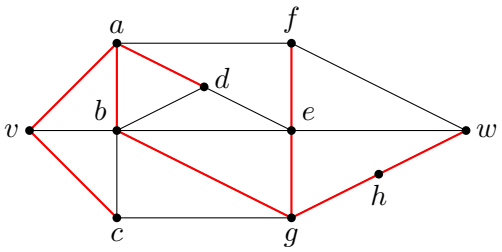
A la derecha mostramos el árbol generador dibujado con la forma habitual de un árbol. □

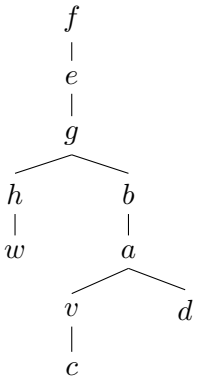
Búsqueda en profundidad. El siguiente algoritmo construye un árbol generador siguiendo un recorrido en el grafo mediante **búsqueda en profundidad**. Esta

búsqueda y los árboles así construidos se usan, por ejemplo, para encontrar las componentes conexas o para comprobar si un grafo es acíclico (es decir, no contiene ciclos).

- Elegimos un vértice arbitrario v_0, que llamamo raíz.
- Construimos un camino que comenzando en v_0 y añadiendo sucesivamente aristas y vértices mientras sea posible sin utilizar vértices ya añadidos al camino.
- Si el camino así construido pasa por todos los vértices del grafo, entonces el árbol generador es dicho camino. En caso contrario, retrocedemos al penúltimo vértice del camino y, si es posible, formamos un nuevo camino que empiece en este vértice y que pase por vértices no visitados.
- Si esto no se puede hacer, lo intentamos retrocediendo al vértice anterior.
- Repetimos el proceso hasta añadir todos los vértices.

Ejemplo 3.2.36 Empezando en el vértice f, hemos construido el árbol generador del siguiente grafo mediante búsqueda en profundidad.



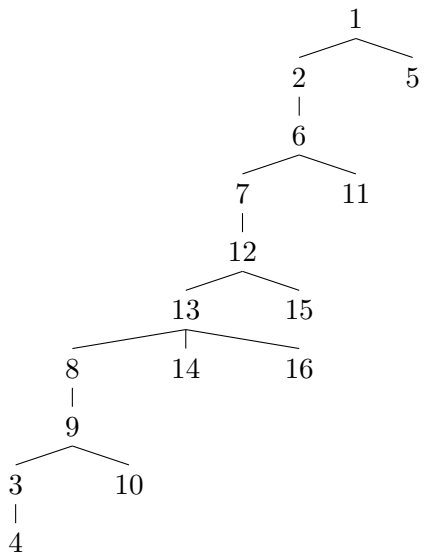


A la derecha, hemos dibujado el árbol con su forma habitual. □

Ejemplo 3.2.37 Considera el grafo dado por la siguiente lista de adyacencia

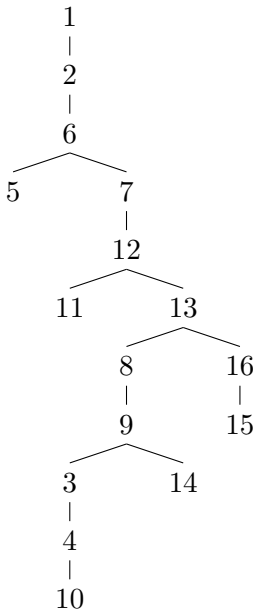
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	1	4	3	1	2	6	9	3	4	6	7	8	9	12	13
5	6	9	10	6	5	12	13	8	9	12	11	12	13	16	15
					7			10			13	14			
					11			14			15	16			

Un árbol generador construido mediante búsqueda en anchura es el siguiente



Aunque este dibujo no representa a todo el grafo, ya que no incluye todas las aristas, sí nos puede ayudar a estudiar sus propiedades; por ejemplo, podemos deducir que el grafo es conexo, ya que incluye todos los vértices.

A continuación, mostramos otro árbol generador construido mediante búsqueda en profundidad. Para construir los caminos, elegimos los vértices según el orden numérico.



Igual que en el árbol anterior, debemos tener en cuenta que este dibujo no representa a todo el grafo, ya que no incluye todas las aristas. □

3.2.6. Isomorfismo de grafos

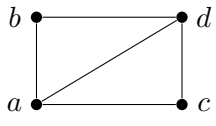
DEFINICIÓN 3.2.38 *Se dice que dos grafos simples $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$ son **isomorfos** si existe una función biyectiva $\phi\colon V_1 \to V_2$ tal que, para todo $u, v \in V_1$,*

$$\{u, v\} \in E_1 \quad \textit{si y solo si} \quad \{\phi(u), \phi(v)\} \in E_2$$

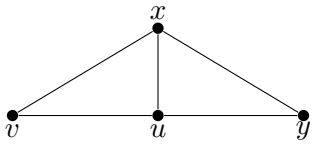
*Decimos también que la función ϕ es un **isomorfismo de grafos**.*

Es decir, dos grafos son isomorfos si entre sus vértices existe una función biyectiva que conserva las adyacencias en los dos sentidos. Matricialmente, esto significa que existe una reordenación de los vértices de G_1 y G_2 de manera que las matrices de adyacencia son exactamente iguales.

EJEMPLO 3.2.39 Los siguientes grafos son isomorfos



$G_1 = (V_1, E_1)$



$G_2 = (V_2, E_2)$

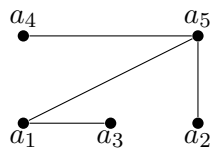
Y el isomorfismo está dado por la siguiente biyección:

$$\begin{array}{rcl} \phi : V_1 & \to & V_2 \\ a & \mapsto & u \\ b & \mapsto & v \\ c & \mapsto & y \\ d & \mapsto & x \end{array} \qquad \left\{ \begin{array}{l} \{a,b\} \in E_1 \iff \phi(\{a,b\}) = \{u,v\} \in E_2 \\ \{a,c\} \in E_1 \iff \phi(\{a,c\}) = \{u,y\} \in E_2 \\ \{a,d\} \in E_1 \iff \phi(\{a,d\}) = \{x,u\} \in E_2 \\ \{b,d\} \in E_1 \iff \phi(\{b,d\}) = \{v,x\} \in E_2 \\ \{c,d\} \in E_1 \iff \phi(\{c,d\}) = \{x,y\} \in E_2 \end{array} \right.$$

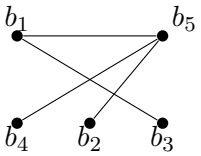
□

La relación de isomorfía entre grafos es una relación de equivalencia. Cada clase de equivalencia definida por esta relación es un conjunto de grafos isomorfos que se denomina **grafo no etiquetado**.

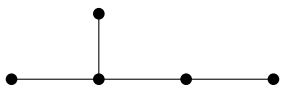
EJEMPLO 3.2.40 Los grafos G_1 y G_2 de la siguiente figura son isomorfos, basta tomar la función dada por $\phi(a_i) = b_i$.



G_1



G_2



G_3

Ambos, como grafos isomorfos, pertenecen a la clase representada por un grafo no etiquetado que podemos dibujar, por ejemplo, como G_3. □

Por lo general, verificar si dos grafos son o no isomorfos es un problema bastante complejo cuando trabajamos con grafos con gran número de vértices. Para abordar el problema, en primer lugar se analizan características necesarias para que dos grafos sean o no isomorfos y cuyo análisis tenga poco coste computacional. Como último recurso, se procedería a la búsqueda sistemática de un isomorfismo.

Por ejemplo, en primer lugar podríamos obtener y analizar lo que se llama **invariantes** de un grafo, es decir, magnitudes que se mantienen bajo isomorfismo. Si dos grafos difieren en alguna de estas invariantes, no pueden ser isomorfos. Algunas invariantes son:

1. Número de vértices.
2. Número de aristas.
3. *Sucesión gráfica*, es decir la lista de grados de los vértices, ordenados en orden decreciente.
4. Número de componentes conexas.
5. Número de ciclos o circuitos de determinada longitud incluidos en el grafo.

Y otros características que iremos estudiando a lo largo del tema, como coloración, planaridad,...

Ejemplo 3.2.41 Los grafos G_1 y G_2 no son isomorfos, aunque tiene el mismo número de aristas y vértices y la secuencia gráfica es la misma en ambos grafos, $(3, 3, 2, 2, 2)$.

G_1 G_2

Para demostrar que no son isomorfos, basta observar el grafo G_2 contiene un ciclo de longitud 3, mientras que G_1 no contiene ningún ciclo de longitud 3. □

Ejemplo 3.2.42 Los grafos dados por las siguientes listas de adyacencia no son isomorfos.

G_1

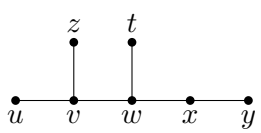
1	2	3	4	5	6	7	8	9
2	1	2	3	4	1	5	6	3
6	3	4	5	7	7	6		
		9			8			

G_2

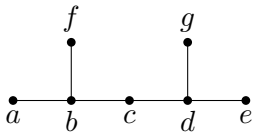
1	2	3	4	5	6	7	8	9
2	1	4	3	2	1	3	9	8
6	5	7	7	6	2	4		
	6				5			

El número de vértices y arístas es el mismo, y la sucesión gráfica también es la misma en ambos grafos: $(3, 3, 2, 2, 2, 2, 1, 1)$. Sin embargo, el grafo G_1 es conexo y solo tiene una componente conexa, mientras que el grafo G_2 tiene tres, las formadas por los siguientes conjuntos de vértices: $\{1, 2, 5, 6\}, \{3, 4, 7\}, \{8, 9\}$. □

Ejemplo 3.2.43 Los siguientes grafo no son isomorfos.



G_1



G_2

Vemos que el número de vértices y aristas es el mismo, y también lo es la sucesión gráfica. Además, ninguno de los dos contiene ciclos ni circuitos. En este caso, nos podemos fijar en los dos vértices de grado 3 de los: en G_1, estos dos vértices están conectados, mientras que en G_2, no lo están. □

Ejemplo con Maxima 3.2.44 En `Maxima`, disponemos del operador `isomorphism` que analiza si dos grafos son isomorfos y, en tal, caso determina un isomorfismo. Los tres ejemplos que mostramos a continuación tienen la misma secuencia gráfica, pero solo dos de ellos son isomorfos.

```
(%i1) load(graphs)$
(%i2) g4: create_graph([1,2,3,4,5],[[1,4],[1,5],[2,3],
        [2,4],[2,5],[3,4]])$
      degree_sequence(g4);
(%o2) [2,2,2,3,3]
(%i3) g5: create_graph([1,2,3,4,5],[[1,2],[1,3],[1,5],
        [2,3],[3,4],[4,5]])$
      degree_sequence(g5);
(%o3) [2,2,2,3,3]
(%i4) g6: create_graph([1,2,3,4,5],[[1,4],[1,5],[2,4],
        [2,5],[3,4],[3,5]])$
```

```
          degree_sequence(g6);
(%o4)  [2,2,2,3,3]
(%i5)  isomorphism(g4,g5);
(%o5)  [4->3,5->5,3->2,1->4,2->1]
 (%i6)  isomorphism(g4,g6);
(%o6)  []
```

□

3.2.7. Grafos Eulerianos

Los grafos **eulerianos** deben su nombre a Leonhard Paul Euler, que es considerado el padre de la teoría de grafos. Esto se debe a que en 1736, resolvió matemáticamente un entretenimiento habitual en la ciudad de Königsberg (nombre de la ciudad rusa actualmente conocida como Kaliningrado y que entonces pertenecía a Prusia Oriental). Los habitantes de la ciudad se preguntaban si era posible recorrer los siete puentes que en aquel momento permitían cruzar el río Pregel, pero pasando solamente una vez por cada puente (ver figura 3.2). Euler demostró que no era posible hacerlo, ya que el número de puentes que se podían tomar era impar en más de dos zonas o islas.

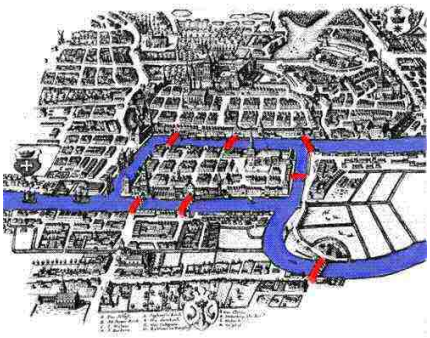
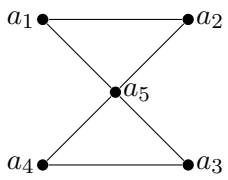


Figura 3.2: Mapa de la ciudad de Köningsber en el siglo XVIII.[1]

DEFINICIÓN 3.2.45 *Un **camino de Euler** o **euleriano** entre dos vértices distinos u y v de un grafo, es un camino que recorre cada arista del grafo exactamente una vez, es decir, un camino simple que recorre todas las aristas. Un **circuito de Euler** o **euleriano** es un camino cerrado que recorre cada arista del grafo exactamente una vez; si existe un circuito de Euler, decimos que el grafo es **euleriano**.*

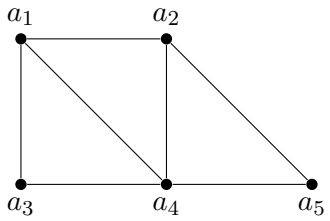
[1] Imagen de Bogdan Giuşcă, CC BY-SA 3.0, tomada de Wikipedia. `http://commons.wikimedia.org/wiki/File:Konigsberg_bridges.png`

Ejemplo 3.2.46 El grafo de la figura es euleriano, ya que todas sus aristas se pueden recorrer con un circuito de euler.



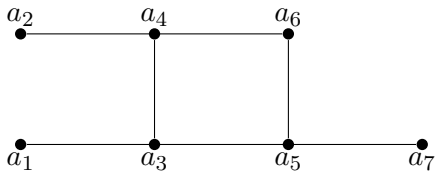
Un posible circuito de Euler es $C = a_1\ a_5\ a_3\ a_4\ a_5\ a_2\ a_1$. □

Ejemplo 3.2.47 Para el grafo de la figura siguiente, no es posible encontrar un circuito de Euler, pero sí es posible recorrer todas sus aristas con un camino de Euler.



Los caminos de Euler de este grafo deben conectar los vértices a_1 y a_2 y uno de ellos es: $C = a_1\ a_3\ a_4\ a_5\ a_2\ a_1\ a_4\ a_2$. □

Ejemplo 3.2.48 El siguiente grafo no contiene ni caminos ni circuitos de Euler



Los resultados siguientes fueron demostrados por Euler para justificar la imposibilidad de realizar el recorrido por todos los puentes de Köningsberg y justifican las afirmaciones hechas en el estudio de los ejemplos anteriores.

Teorema 3.2.49 *Un grafo simple tiene un circuito de Euler si y solo si es conexo y todos los vértices son de grado par.*

Corolario 3.2.50 *Un grafo simple tiene un camino de Euler entre los vértices u, v si y solo si es conexo y esos vértices son los únicos con grado impar.*

Los dos resultados anteriores son válidos también sobre multigrafos. De hecho, el grafo que representa el pasatiempo de lo puentes de Köningsber es un multigrafo.

Hallar caminos y circuitos de Euler es bastante simple y disponemos de varios algoritmos para ello. Los más conocidos son el **algoritmo de Fleury** y el **algoritmo de Hierholzer**; este segundo es el más eficiente y lo describimos a continuación.

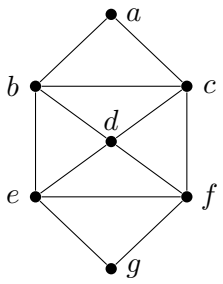
Algoritmo de Hierholzer. Se lo aplicamos a un grafo simple conexo cuyos vértices tienen grado par o que contiene exactamente dos vértices de grado impar.

1. Si el grafo tiene dos vértices de grado impar, elegimos uno de ellos; si todos los vértices tienen grado par podemos elegir cualquier vértice.
2. A partir de ese vértice empezamos a recorrer aristas, sin repetir ninguna hasta que no podamos continuar más.

   Si hemos empezado en un vértice de grado impar, esto ocurrirá necesariamente construyendo un camino que termina en el otro vértice de grado impar. Si el grafo solo tiene vértices de grado par, esto ocurrirá necesariamente construyendo un circuito que termina en el mismo vértice en el que lo hemos iniciado.
3. Si el camino o circuito contiene a todas las aristas, ya hemos terminado. En caso contrario, elegimos un vértice del camino o circuito que hemos construido, y en el que incidan aristas fuera del camino o circuito parcial construido.
4. A partir de ese vértice, construimos otro circuito recorriendo nuevas aristas hasta que no podamos continuar más. Este circuito terminará necesariamente en el mismo vértice en el que hemos empezado. Este segundo circuito se puede insertar en el camino o circuito que ya habíamos construido.
5. Si el camino o circuito contiene a todas las aristas, ya hemos terminado. En caso contrario, repetimos el proceso descrito en el punto anterior hasta conseguir incluir todas las aristas.

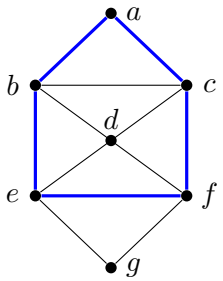
El algoritmo anterior es *no determinista*, ya que la elección de los vértices y la construcción de los circuitos parciales se hace de manera arbitraria.

Ejemplo 3.2.51 Vamos a determinar una circuito de Euler en el siguiente grafo



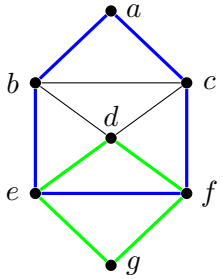
Empezamos eligiendo un vértice y construyendo un circuito que empiece y termine en ese vértice. Por ejemplo, desde el vértice a podemos construir el circuito.

$$a \blacktriangleright b \blacktriangleright e \blacktriangleright f \blacktriangleright c \blacktriangleright a$$



Excepto el vértice a, los demás vértices tienen aristas incidentes que no hemos incluido en el circuito. Elegimos uno cualquiera, por ejemplo e, y construimos un circuito que empiece y termine en e y utilizando aristas que todavía no hemos recorrido:

$$e \blacktriangleright d \blacktriangleright f \blacktriangleright g \blacktriangleright e$$



Insertamos entonces el circuito "verde" en el circuito "azul",

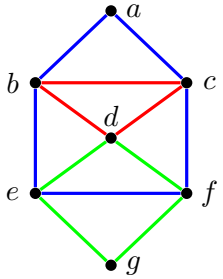
$$\overbrace{a \blacktriangleright b \blacktriangleright e \blacktriangleright f \blacktriangleright c \blacktriangleright a}^{e \blacktriangleright d \blacktriangleright f \blacktriangleright g \blacktriangleright e}$$

y obtenemos un circuito que recorre más aristas del grafo inicial

$$a \blacktriangleright b \blacktriangleright e \blacktriangleright d \blacktriangleright f \blacktriangleright g \blacktriangleright e \blacktriangleright f \blacktriangleright c \blacktriangleright a$$

Las aristas $\{b, c\}$, $\{c, d\}$, y $\{d, b\}$ no están en el circuito parcial, elegimos una de ellas para construir el siguiente. Por ejemplo, elegimos el vértice d y la arista $\{d, b\}$ para construir el siguiente circuito parcial:

$$d \blacktriangleright b \blacktriangleright c \blacktriangleright d$$



Insertamos este último circuito en el anterior

$$\overbrace{a \blacktriangleright b \blacktriangleright e \blacktriangleright d \blacktriangleright f \blacktriangleright g \blacktriangleright e \blacktriangleright f \blacktriangleright c \blacktriangleright a}^{d \blacktriangleright b \blacktriangleright c \blacktriangleright d}$$

y obtenemos el circuito de Euler del grafo inicial

$$a \blacktriangleright b \blacktriangleright e \blacktriangleright d \blacktriangleright b \blacktriangleright c \blacktriangleright d \blacktriangleright f \blacktriangleright g \blacktriangleright e \blacktriangleright f \blacktriangleright c \blacktriangleright a$$ □

Ejemplo 3.2.52 Consideremos el grafo dado por la siguiente lista de adyacencia:

a	b	c	d	e	f	g	h
c	e	a	a	b	c	b	e
d	g	d	c	c	d	d	g
		e	f	g		e	
		f	g	h		h	

Todos los vértices tienen grado par, por lo que es euleriano. Vamos a utilizar la búsqueda primero en profundidad para construir los circuitos parciales en el algoritmo de Hierholzer. Empezamos por el vértice a y vamos describiendo un circuito eligiendo el primer vértice en la columna del cada vértice que no hayamos incluido previamente. Indicaremos con un superíndice el orden en el que vamos eligiendo y recorriendo los vértices. Empezando en a vamos a c; en la columna de c, descartamos a y vamos al siguiente vértice, que es d; en la columna d, descartamos c y vamos al siguiente vértice que es a, por lo que hemos construido el primer circuito.

a^1	b	c^2	d^3	e	f	g	h
c^2	e	a^1	a^1	b	c	b	e
d^3	g	d^3	c^2	c	d	d	g
		e	f	g		e	
		f	g	h		h	

$$a - c - d - a$$

Borramos las aristas recorridas para ver mejor las que nos quedan por recorrer:

a	b	c	d	e	f	g	h
	e			b	c	b	e
	g			c	d	d	g
		e	f	g		e	
		f	g	h		h	

El primer vértice de este circuito cuya columna tiene vértices sin visitar es c, así que a partir de él construimos el siguiente circuito parcial.

a	b^3	c^1	d^5	e^2	f^6	g^4	h
	e^2			b^3	c^1	b^3	e
	g^4			c^1	d^5	d^5	g
		e^2	f^6	g		e	
		f	g^4	h		h	

$$\overbrace{a - \boxed{c} - d - a}^{c - e - b - g - d - f - c}$$

Insertamos este segundo circuito en el anterior y obtenemos

$$a - c - e - b - g - d - f - c - d - a$$

Borramos las aristas que hemos incluido en el circuito para visualizar mejor las aristas pendientes

a	b	c	d	e	f	g	h
							e
							g
				g		e	
				h		h	

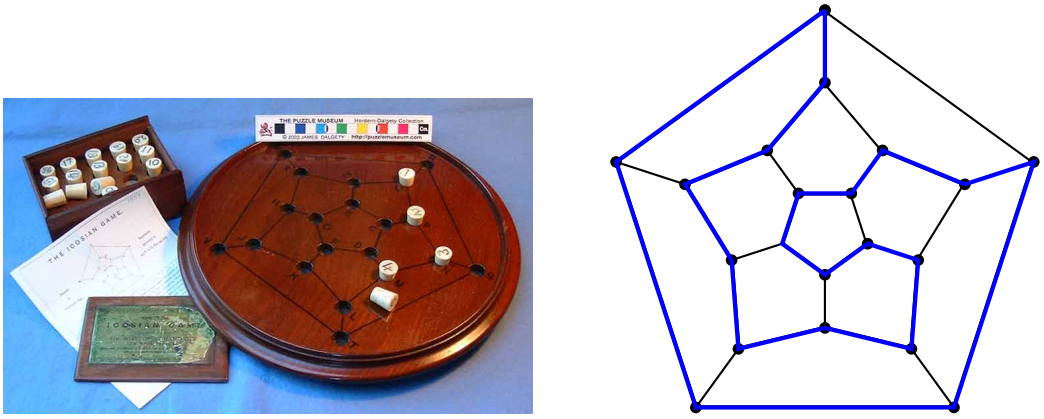


Figura 3.3: Ciclo hamiltoniano en el juego icosiano.[2]

El primer vértice con una arista incidente fuera del circuito es e; a partir de él, construimos el último circuito que insertaremos:

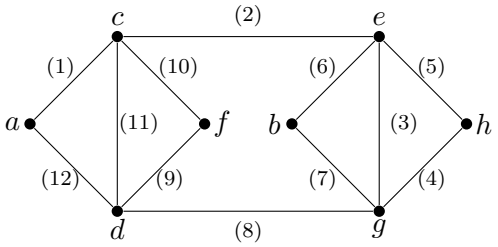
$$\begin{array}{cccccccc} a & b & c & d & e^1 & f & g^2 & h^3 \\ \hline & & & & & & & e^1 \\ & & & & & & & g^2 \\ & & & & g^2 & & e^1 & \\ & & & & h^3 & & h^3 & \end{array}$$

$$a - c - \boxed{e} - \underbrace{b - g - d - f - c - d - a}_{e - g - h - e}$$

Por lo tanto, el resultado es:

$$a - c - e - g - h - e - b - g - d - f - c - d - a$$

Vemos a continuación el dibujo del grafo con las aristas numeradas siguiendo el orden de recorrido en el circuito de Euler.



□

3.2.8. Grafos hamiltonianos

Los grafos hamiltonianos toman su nombre del matemático irlandés William Rowan Hamilton, que en 1857 presentó ante la Asociación Británica un juego llamado *Juego Icosiano*. Este juego contenía un tablero en la que aparecía dibujado

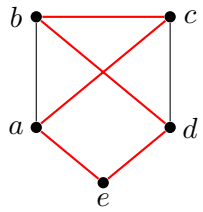
[2] La imagen de la izquierda está tomada de la web de *The Puzzle Museum*, (`http://puzzlemuseum.com/month/picm02/200201hamilton.jpg`).

un dodecaedro cuyos vértices estaban etiquetados con el nombre de veinte ciudades importantes y el objetivo era encontrar recorridos cerrados a través de las veinte ciudades que pasaran exactamente una vez por cada una de ellas. El juego no tuvo mucho éxito comercial y, aunque su origen estaba en unos estudios sobre estructuras algebraicas que generalizaban a los números complejos, suponía formular por primera vez el problema de encontrar en un grafo lo que en adelante se llamarían **ciclos de Hamilton**.

DEFINICIÓN 3.2.53 *Un **camino de Hamilton** o **hamiltoniano** entre dos vértices distintos u y v, es un camino que pasa por cada vértice del grafo exactamente una vez, es decir, un camino elemental que recorre todos los vértices. Un **ciclo de Hamilton** o **hamiltoniano** es un camino cerrado que recorre todas los vértices del grafo exactamente una vez. Un grafo se dice hamiltoniano si contiene un ciclo de Hamilton.*

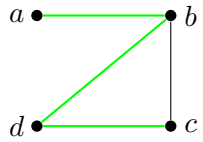
El grafo que aparece a la derecha en la figura 3.3 contiene un ciclo de Hamilton, que aparece resaltado en azul.

EJEMPLO 3.2.54 El siguiente grafo contiene ciclos de Hamilton, por ejemplo, el que aparece resaltado en rojo: $C = a\ c\ b\ d\ e\ a$;



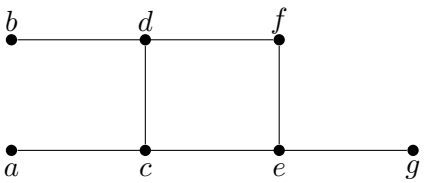
□

EJEMPLO 3.2.55 El siguiente grafo no contiene ciclos de Hamilton, pero sí contiene caminos de Hamilton: el que aparece resaltado en verde.



□

EJEMPLO 3.2.56 El siguiente grafo no contiene ni ciclos ni caminos de Hamilton.



□

El estudio de la existencia de caminos o ciclos de Hamilton es mucho más complejo que el correspondiente para los caminos y circuitos de Euler. No disponemos de resultados que caractericen los grafos que contienen estos caminos y que estén basados en los grados de los vértices. Tampoco hay algoritmos eficientes para determinar, si es posible, este tipo de caminos. Los siguientes resultados establecen condiciones suficientes, pero no necesarias, para afirmar la existencia de ciclos de Hamilton.

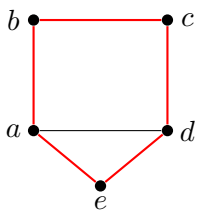
TEOREMA 3.2.57 (DIRAC) *Si $n \geq 3$ es el número de vértices de un grafo simple y todos ellos tienen grado mayor o igual a $\frac{n}{2}$, entonces el grafo contiene un ciclo hamiltoniano.*

TEOREMA 3.2.58 (ORE) *Si $n \geq 3$ es el número de vértices de un grafo simple y para cada par de vértices NO adyacentes u, v, se verifica que $\delta(u) + \delta(v) \geq n$, entonces el grafo contiene un ciclo hamiltoniano.*

El ejemplo 3.2.56 no verifica ni la condición del teorema de Dirac ni la condición del teorema de Ore. Sin embargo, eso no permite concluir que el grafo no contenga ciclos de Hamilton, debemos realizar la búsqueda sistemática de un ciclo y comprobar que no es posible hacerlo.

El ejemplo 3.2.54 no verifica la condición del teorema de Dirac, ya que el grado del vértice e es $2 < \frac{5}{2}$. Sin embargo, sí verifica la condición del teorema de Ore.

EJEMPLO 3.2.59 El siguiente grafo no verifica ni la condición del teorema de Dirac, ni la condición del teorema de Ore, sin embargo, sí contiene un ciclo de Hamilton: $C = a\ b\ c\ d\ e\ a$



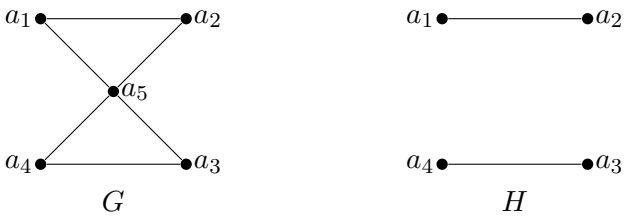
□

Para demostrar que un grafo no es hamiltoniano, podemos usar el siguiente resultado, que establece una condición necesaria, pero no suficiente.

TEOREMA 3.2.60 *Si G un grafo hamiltoniano y H un subgrafo de G obtenido eliminando n vértices (y las aristas incidentes en ellos), entonces el número de componentes conexas de H es menor o igual que n.*

Es decir, si al quitar n vértices obtenemos más de n componentes conexas, podemos afirmar que el grafo no es hamiltoniano.

EJEMPLO 3.2.61 El grafo G no es hamiltoniano, ya que su subgrafo H tiene dos componentes conexas y se ha obtenido eliminado solamente un vértice, el a_5:



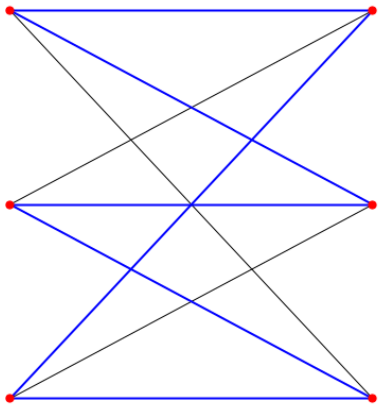
□

EJEMPLO CON MAXIMA 3.2.62 El operador `hamilton_cycle` determina si un grafo es o no Hamiltoniano y en tal caso calcula el ciclo de Hamilton.

```
(%i1) load(graphs)$
(%i2) k33: complete_bipartite_graph(3,3)$
(%i3) hk33: hamilton_cycle(k33);
(%o3)     [0,5,2,4,1,3,0]
```

Utilizando la opción `show_edges` del operador `draw_graph` podemos ver el dibujo de un grafo en el que se resalte un determinado camino, por ejemplo el ciclo de Hamilton de un grafo hamiltoniano.

```
(%i4) draw_graph(k33,show_edges=vertices_to_cycle(hk33));
(%o4)     done
```

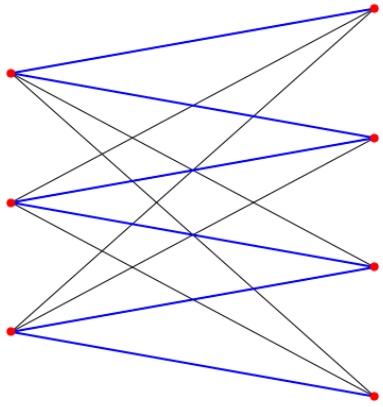


También podemos determinar el camino de Hamilton contenido en un grafo semihamiltoniano. Por ejemplo, sabemos que el grafo $K_{3,4}$ no es hamiltoniano, pero sí es semihamiltoniano.

```
(%i5) k34:complete_bipartite_graph(3,4)$
      hamilton_cycle(k34);
(%o5) []
(%i6) hk34: hamilton_path(k34);
(%o6) [6,2,5,1,4,0,3]
```

En este caso, el operador `vertices_to_path` en la opción `show_edges` también nos permite visualizar el camino de Hamilton.

```
(%i7) draw_graph(k34,show_edges=vertices_to_path(hk34));
(%o7) done
```

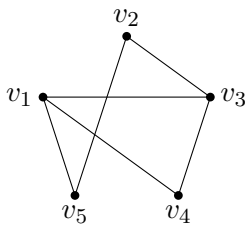


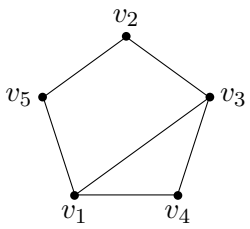
□

3.2.9. Planaridad

DEFINICIÓN 3.2.63 *Se dice que un grafo es* ***plano*** *si puede dibujarse en el plano sin que se corten ningún par de aristas.*

Por ejemplo, las figuras





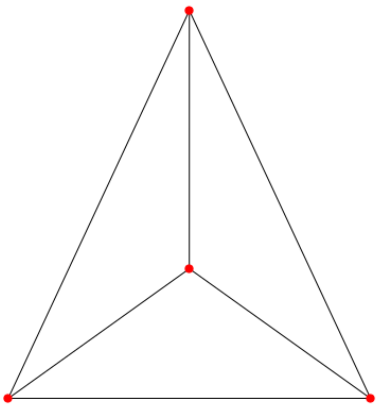
son dos dibujos distintos del mismo grafo; el de la izquierda presenta intersecciones de aristas mientras que el de la derecha no. Por lo tanto, el grafo es plano.

Ejemplo con Maxima 3.2.64 En `Maxima`, podemos analizar la planaridad de un grafo utilizanado el operador `is_planar`.

```
(%i1) load(graphs)$
(%i2) g9: create_graph(7, [[0,1],[0,5],[1,2],[1,4],
        [2,3], [0,3], [2,5],[3,6],[4,5],[4,6],[5,6]])$
      is_planar(g9);
(%o2)    false
```

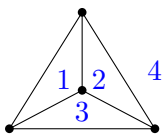
El algoritmo básico que utiliza `Maxima` para generar la representación gráfica de un grafo, no produce siempre una representación plana. Sin embargo, también permite elegir entre varios programas para determinar la posición final de los vértices en la representación gráfica; por ejemplo, `planar_embedding` fuerza la representación plana en la "mayoría" de las situaciones.

```
(%i3) draw_graph(complete_graph(4),redraw=true,
        program=planar_embedding);
(%o3)    done
```



□

Una representación plana de un grafo divide al plano en **regiones**, las partes del plano que quedan delimitadas por las aristas y los vértices, en donde la parte externa, no acotada, también se considera región. Por ejemplo, el grafo completo K_4, determina las cuatro regiones numeradas en la siguiente figura.



En un grafo plano y conexo, el número de vértices, el número de aristas y el número de regiones están relacionados por la fórmula de Euler.

TEOREMA 3.2.65 (FÓRMULA DE EULER) *Sea $G = (V, E)$ un grafo plano conexo con $|V| = v$, $|E| = e$, y sea r el número de regiones de una representación plana de G. Entonces, $v - e + r = 2$*

COROLARIO 3.2.66 *El grafo $K_{3,3}$ es un grafo no plano.*

Demostración: Si $K_{3,3}$ fuese plano, por la fórmula de Euler, su representación plana dividiría el plano en $9 - 6 + 2 = 5$ regiones; vamos a ver que esto es no posible. Por ser bipartito, el grafo $K_{3,3}$ no puede contener ciclos de longitud 3, ya que C_3 no es bipartito. Por lo tanto, los ciclos contenidos $K_{3,3}$ deben tener al menos 4 aristas. En consecuencia, cada región debe estar acotada por un ciclo de al menos 4 aristas. Dado que además cada arista es común a dos regiones, se debe verificar que $4r \leq 2e = 18$, es decir, $r \leq 4.5$. □

Observamos que en esta demostración solo hemos usado que el grafo $K_{3,3}$ no contiene ciclos de longitud tres, y que como mínimo los ciclos que contengan tendrán longitud 4. De ahí hemos deducido que todo grafo plano sin ciclos de lóngitud 3 debe verificar que $4r \leq 2e$ y, en consecuencia, por la fórmula de Euler:

$$2e \geq 4r = 4(e - v + 2) = 4e - 4v + 8 \qquad \Longrightarrow \qquad e \leq 2v - 4$$

Esta condición necesaria de planaridad se puede usar para deducir que un grafo no es plano, tal y como establece el siguiente resultado.

COROLARIO 3.2.67 *Si G es un grafo simple conexo con $v \geq 3$ vértices, e aristas, no contiene ciclos de longitud 3 y verifica que $e > 2v - 4$, entonces G no es plano.*

En general, si un grafo plano contiene ciclos de longitud 3, solo podríamos deducir que $2e \geq 3r$ y en consecuencia,

$$2e \geq 3r = 3(e - v + 2) = 3e - 3v + 6 \qquad \Longrightarrow \qquad e \leq 3v - 6$$

Obtenemos entonces una condición necesaria de planaridad más general. Esta condición, igual que la anterior, es necesaria, pero no suficiente. Por ejemplo, hemos visto que $K_{3,3}$ no es plano y, sin embargo, sí verifica la desigualdad anterior: $e = 9 \leq 3 \cdot v - 6 = 3 \cdot 6 - 6 = 12$. Por esta razón, esta condición se utiliza igualmente para deducir que un grafo no es plano.

COROLARIO 3.2.68 *Si G un grafo conexo con $v \geq 3$ vértices y e aristas y verifica que $e > 3v - 6$, entonces G no es plano.*

COROLARIO 3.2.69 *El grafo K_5 no es plano.*

Demostración: Para el grafo K_5, $3 \cdot v - 6 = 3 \cdot 5 - 6 = 9 < 10 = e$ y por lo tanto, no es plano. □

Finalmente, podemos obtener otra condición necesaria de planaridad a partir del teorema de Euler sobre los grados de los vertices en un grafo simple:

COROLARIO 3.2.70 *Todo grafo plano conexo tiene un vértice de grado menor o igual a 5.*

Demostración: Si todos los vértices tuvieran grado mayor que 5, entonces

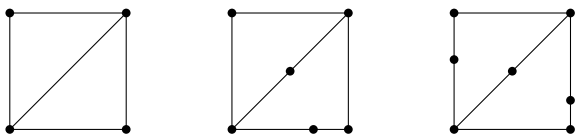
$$e = \frac{1}{2}\sum_{x\in V}\delta(x) \geq \frac{5}{2}\cdot v \geq 3v > 3v - 6,$$

y en consecuencia el grafo no sería plano.

Hemos destacado entre los resultados anteriores el hecho de que los grafos K_5 y $K_{3,3}$ no son planos. Esto se debe a que, estos grafos nos van a permitir caracterizar los grafos planos con el teorema de Kuratowski. Antes de enunciar este teorema, necesitamos introducir el concepto de *homeomorfismo* de grafos.

DEFINICIÓN 3.2.71 *Decimos que el grafo G_1 es **homeomorfo** a G_2 si uno de ellos se puede obtener a partir de el otro insertando vértices en alguna de sus aristas.*

Por ejemplo, los tres grafos de la figura son homeomorfos:

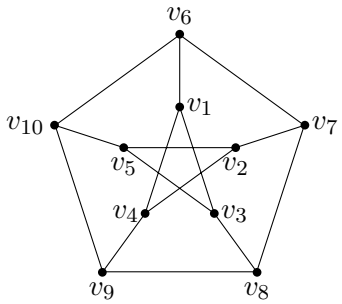


Es obvio que si añadimos vértices en medio de una arista, no cambiamos la condición de planaridad del grafo. El matemático polaco Kazimierz Kuratowski estableció en 1930 el siguiente teorema que caracteriza los grafos planos utilizando el concepto de homeomorfismo de grafos.

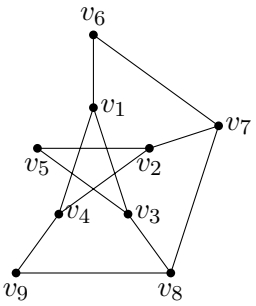
TEOREMA 3.2.72 (KURATOWSKI) *Un grafo es no plano si, y sólo si, contiene un subgrafo que es homeomorfo al grafo K_5 ó al grafo $K_{3,3}$.*

Es decir, podemos deducir que un grafo no es plano si podemos obtener el grafo K_5 o el grafo $K_{3,3}$ eliminando vértices (y las aristas que inciden en él), eliminando aristas o eliminando vértices de grado dos uniendo las aristas que inciden en él en una única arista.

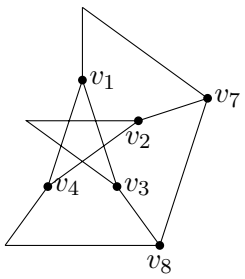
EJEMPLO 3.2.73 El siguiente grafo se conoce como **grafo de Petersen** y no es plano.



Vamos a construir un subgrafo homeomorfo a $K_{3,3}$ para demostrar que efectivamente no es plano. En primer lugar construimos un subgrafo eliminando el vértice v_{10} y todas las aristas que inciden en él.



Este grafo es homeomorfo a $K_{3,3}$, lo que se puede observar fácilmente si eliminamos los vértices v_5, v_6 y v_9, pero manteniendo las aristas $\{v_2, v_3\}$, $\{v_1, v_7\}$ y $\{v_4, v_8\}$:



□

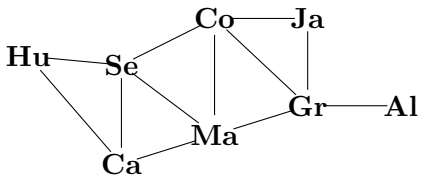
3.2.10. Coloración de Grafos

Colorear los vértices de un grafo consiste en asignar un color a cada vértice del grafo de manera que dos vértices adyacentes no tengan el mismo color.

DEFINICIÓN 3.2.74 *Sea $G = (V, E)$ un grafo simple y C un conjunto de m elementos (colores). Una* ***coloración*** *con m colores de los vértices del grafo G es una función $c\colon V \to C$ tal que si $u, v \in V$ y $\{u, v\} \in E$, entonces $c(u) \neq c(v)$.*

La denominación coloración viene del problema más representativo asociado a este concepto, la coloración de un mapa de tal forma que países o provincias con frontera común no estén coloreadas con el mismo color.

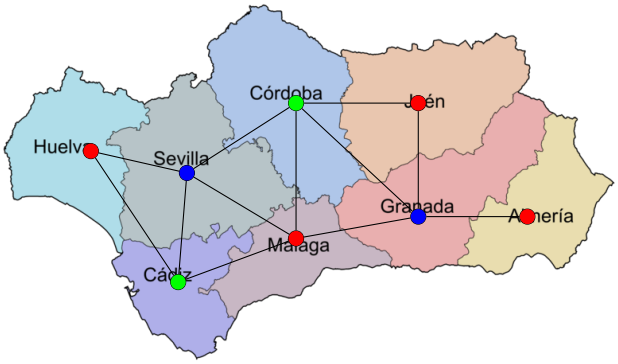
El grafo asociado a este mapa tiene por vértices a las provincias y las aristas unen las provincias con frontera común.



En el mapa de Andalucía hemos utilizado ocho colores diferentes, lo que naturalmente asegura una coloración, pero ¿cuál es el menor número de colores necesario? En 1976 se demostró el denominado **teorema de los 4 colores**, que se había conjeturado en 1852, y que establece que no necesitamos más de 4 colores.

Teorema 3.2.75 (Guthrie/ Appel/ Haken) *Todo grafo plano se puede colorear con cuatro colores o menos.*

Por ejemplo, para colorear el mapa de Andalucía son suficientes tres colores:



El menor número de colores necesario para colorear un grafo se denomina **número cromático** del grafo, y se denota $\chi(G)$. Por lo tanto, para determinar que el

número cromático de un grafo es m, es necesario en primer lugar encontrar una coloración con m colores y, en segundo lugar, demostrar que no es posible colorear con menos colores.

Ejemplo con Maxima 3.2.76 En `Maxima`, disponemos de operadores que nos determinan el número cromático de un grafo y calculan una coloración óptima.

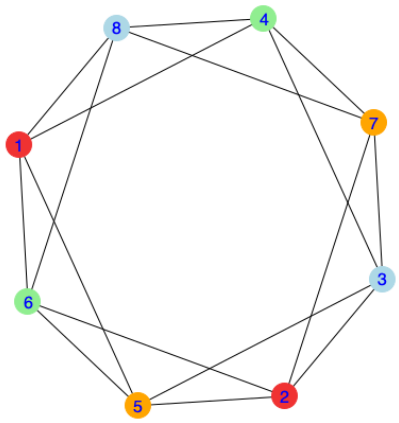
```
(%i1) load(graphs)$
(%i2) gcol:create_graph([1,2,3,4,5,6,7,8],[[3, 4],[4, 8],
          [2,5],[1,8],[5,6],[7,8],[4,7],[2,6],[1,4],[3,7],
          [2,7],[6,8],[2,3],[3,5],[1,6],[1,5]])$
(%i3) chromatic_number(gcol);
(%o4)  4
```

Por otra parte, `vertex_coloring`, también devuelve el número cromático pero incluyendo una coloración con ese número de colores. Los colores están representados por números naturales

```
(%i5) vertex_coloring(gcol);
(%o5)  [4,[[8,2],[7,4],[6,1],[5,4],[4,1],[3,2],[2,3],[1,3]]]
```

Es decir, los vértices 4 y6 están coloreados con el color 1, los vértices 3 y 8 con el color 2, los vértices 1 y 2 con el color 3 y los vértices 5 y7 están con el color 4. Podemos visualizar el grafo con diferentes colores usando la opción `vertex_partition`.

```
(%i6) draw_graph(gcol, vertex_size=4,show_id=true,
              vertex_partition=[[1,2],[3,8],[4,6],[5,7]]);
(%o72)  done
```



□

No existe ningún algoritmo que de forma eficiente determine el número cromático de un grafo y la coloración con el menor número de colores. El siguiente algoritmo

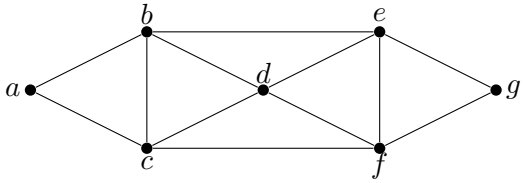
permite obtener fácilmente una coloración de un grafo, pero no garantiza que se use el menor número de colores.

- Empezamos ordenando los vértices según el orden decreciente de sus grados,

$$\delta(v_1) \geq \delta(v_2) \geq \ldots \delta(v_{n-1}) \geq \delta(v_n)$$

- Asignamos el primer color al vértice v_1, es decir: $c(v_1) = 1$ y, siguiendo la secuencia, a los vértices que no sean adyacentes a él y a los que coloreemos con el mismo color.
- Asignamos el segundo color al primer vértice de la secuencia al que no se le haya asignado el primer color, y siguiendo la secuencia a los vértices que no sean adyacentes a él y a los que coloreemos con el mismo color.
- Siguiendo la lista ordenada de vértices, repetimos el proceso hasta colorear todos los vértices.

Ejemplo 3.2.77 Consideremos el grafo



Una ordenación de los vértices según su grado sería

Grados	4	4	4	4	4	2	2
Vértices	b	c	d	e	f	a	g

Asignamos el color 1 al vértice b, después al vértice f. El resto de vértices son adyacentes a b o a f.

Grados	4	4	4	4	4	2	2
Vértices	b	c	d	e	f	a	g
Color	1				1		

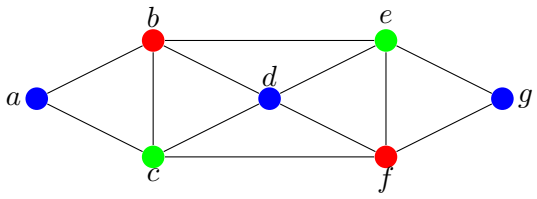
Asignamos el color 2 al vértice c y después al vértice e. El resto de vértices sin colorear son adyacentes a c o a e.

Grados	4	4	4	4	4	2	2
Vértices	b	c	d	e	f	a	g
Color	1	2		2	1		

Finalmente, asignamos el color 3 a los vértices d, a y g.

Grados	4	4	4	4	4	2	2
Vértices	b	c	d	e	f	a	g
Color	1	2	3	2	1	3	3

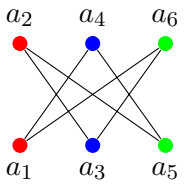
Mostramos el grafo identificando el color 1 con el rojo, el color 2 con el verde y el color 3 con el azul.



Por lo tanto, hemos coloreado el grafo con tres colores, pero todavía no podemos concluir que su número cromático sea 3. Necesitamos probar que no es posible colorearlo con solamente dos colores. Sin embargo esto es bastante inmediato, ya que el grafo contiene ciclos de longitud 3, que evidentemente no se pueden colorear con menos de tres colores. □

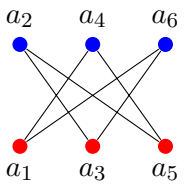
Debemos insistir en que el algoritmo mostrado en el ejemplo anterior no produce necesariamente una coloración óptima, con el menor número de colores posible. Dependiendo de la ordenación inicial de los vértices, el número de colores necesarios puede ser diferente.

Ejemplo 3.2.78 Los vértices del grafo siguiente tienen grado 2, por lo que podemos aplicar el algoritmo de coloración a partir de cualquier ordenación de sus vértices. Si utilizamos la ordenación dada por los subíndices utilizados en las etiquetas, necesitamos tres colores:



a_1	a_2	a_3	a_4	a_5	a_6
2	2	2	2	2	2
R	R	A	A	V	V

Sin embargo, la ordenación $[a_1, a_3, a_5, a_2, a_4, a_6]$ sí permite construir una coloración con dos colores, que es evidentemente su número cromático.



a_1	a_3	a_5	a_2	a_4	a_6
2	2	2	2	2	2
R	R	R	A	A	A

□

El algoritmo, por tanto, nos dará una cota superior del número cromático, pero necesitaremos analizar si es o no posible realizar la coloración con menos colores. Para este trabajo, utilizaremos el número cromático conocido de otros grafos, como los grafos bipartitos, grafos completos o ciclos.

Ejemplo 3.2.79 El número cromático del grafo K_n es n, ya que no es posible colorear dos vértices con un mismo color, puesto que todos los pares de vértices están conectados al ser completo. □

Proposición 3.2.80 *Si un grafo contiene un subgrafo isomorfo a K_n, entonces su número cromático es mayor o igual que n.*

Ejemplo 3.2.81 El número cromático de un grafo bipartito es dos. Es más, un grafo es bipartito si y solo si su número cromático es 2. Estas afirmaciones son evidentes, si el grafo es bipartito, asignamos un color a los vértices de una de las partes y el segundo color a los vértices de la otra parte.

De la misma forma, si un grafo se puede colorear con solo dos colores, los conjuntos de vértices coloreados con el mismo color determinan las dos partes del grafo bipartito. □

Ejemplo 3.2.82 Si n es par, el grafo C_n es bipartito y, en consecuencia, su número cromático es 2. Sin embargo, si n es impar, necesitamos un tercer color, es decir, su número cromático es 3. □

Aplicar y utilizar conceptos y algoritmos de la teoría de grafos requiere modelizar un problema de manera adecuada usando los grafos. Esa parte de la resolución de un problema puede ser compleja y requiere entender que la conexión que definen las aristas pueden corresponder con aspectos muy diversos.

Un problema típico en el que se usa la coloración de grafos es la creación de un calendario de exámenes. En este problema, se busca que en el mismo día no coincidan dos exámenes si un mismo alumno tiene que realizar esos dos exámenes. Para plantear este problema con teoría de grafos, consideramos que las asignaturas son los vértices de un grafo y que las aristas conectan dos asignaturas que comparten estudiantes. De esta forma, una coloración óptima del grafo nos diría el menor número de días o franjas necesarias para programar todos los exámenes de forma que todos los alumnos puedan presentarse a sus asignaturas.

Ejemplo 3.2.83 En un laboratorio hay una serie de compuestos químicos, a, b, c, d, e, f, g, h que transportar a otro laboratorio. Por cuestiones de seguridad, no se pueden mover juntos dos compuestos que puedan reaccionar si hay un accidente. Las reacciones peligrosas vienen descritas por las adyacencias definidas por la siguiente

tabla.

a	b	c	d	e	f	g	h
b	a	a	b	b	c	d	e
c	d	e	e	c	h	e	f
	e	f	g	d		h	g
				g			
				h			

Una coloración del grafo descrito por esta tabla, los grupos de productos que podemos trasladar en un mismo viaje y la coloración optima nos daría el menor número de viajes necesario. Vamos a construir una primera coloración. El primer color (en este caso primer traslado) se lo asignaríamos a los compuestos e y a.

Grados	5	3	3	3	3	3	2	2
Vértices	e	b	c	d	g	h	a	f
Color	1						1	1

El segundo color se lo podríamos asignar a b, c y g.

Grados	5	3	3	3	3	3	2	2
Vértices	e	b	c	d	g	h	a	f
Color	1	2	2		2		1	1

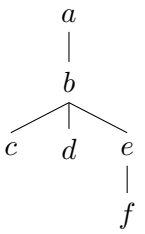
Finalmente, necesitaríamos un tercer color para d y h.

Grados	5	3	3	3	3	3	2	2
Vértices	e	b	c	d	g	h	a	f
Color	1	2	2	3	2	3	1	1

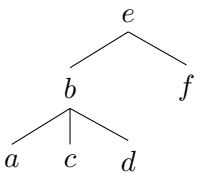
También podemos concluir que no es posible hacer menos viajes para trasladar todos los compuestos, ya que el camino $C = ebde$ es un subgrafo isomorfo a K_3, cuyo número cromático es 3, según hemos visto en el ejemplo 3.2.79. □

3.2.11. Árboles con raíz ordenados

En la construcción de los árboles de búsqueda de las secciones anteriores hemos partido de un vértice, que en principio se puede elegir arbitrariamente, pero que por lo general tendrá un significado destacado en la aplicación. Este nodo destacado recibe el nombre de **raíz**. Habitualmente, los árboles se dibujan tal y como hemos hecho en los ejemplos de las secciones anteriores, orientando hacía abajo los caminos que conectan la raíz con el resto de vértices.



Árbol con raíz a



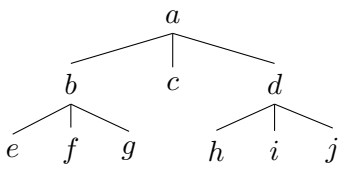
Árbol con raíz e

De esta forma, la disposición de los vértices determina un *orden* entre los vértices, y por ello, los árboles con raíz pueden ser considerados como grafos dirigidos. Además, ese orden establecido por la posición relativa de los vértices introduce los siguientes conceptos.

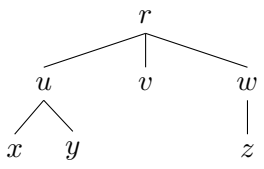
Si T es un árbol con raíz y v un vértice distinto de su raíz: el **padre** de v es el único vértice u tal que hay una arista desde u hasta v en el camino que une v con la raíz; decimos igualmente que v es **hijo** de u. Los vértices con el mismo padre se llaman **hermanos**. Los **antecesores** de un vértice v son todos los vértices del único camino desde la raíz hasta v. Los **descendientes** de un vértice v son aquellos vértices para los que v es un antecesor. Un vértice se dice que es una **hoja** si no tiene hijos. Los **vértices internos** son los vértices que tienen hijos; la raíz se considera vértice interno, a menos que sea el único vértice del grafo, en cuyo caso, se considera hoja. Si v es un vértice de T, el **subárbol** con raíz en x es el subgrafo que contiene al v, a todos sus descendientes y a todas las aristas incidentes en dichos descendientes.

El **nivel** de un vértice en un árbol con raíz es la longitud del único camino desde la raíz hasta dicho vértice; se considera que el nivel de la raíz es cero. La **altura** de un árbol con raíz es el máximo de los niveles de sus vértices. Un árbol con raíz m-ario de altura h se dice que está **equilibrado** si todas sus hojas están en los niveles h o $h-1$.

Un árbol con raíz se llama **árbol m-ario** si todos los vértices internos tienen, a lo sumo, m hijos; en particular, se dirá **binario** si cada vértice interno tiene a los sumo 2 hijos, **ternario** si cada vértice interno tiene a los sumo tres hijos. El árbol se llama **m-ario completo** si cada vértice interno tiene exactamente m hijos. Los árboles m-arios completos se usan habitualmente en problemas de búsqueda, ordenación y codificación.



Árbol ternario **completo**



Árbol ternario **no completo**

TEOREMA 3.2.84 *Un árbol de n vértices tiene $n-1$ aristas.*

TEOREMA 3.2.85 *Un árbol m-ario completo con i vértices internos tiene $n = i \cdot m + 1$ vértices.*

TEOREMA 3.2.86 *Un árbol m-ario de altura h tiene, a lo sumo, m^h hojas.*

Estos resultados permiten abordar problemas en los que necesitemos contar vértices, aristas, hojas,... en estructuras físicas o virtuales dispuestas en forma de árbol.

EJEMPLO 3.2.87 Si sabemos que un árbol ternario es completo y tiene 34 vértices internos, entonces tiene $34 \cdot 3 + 1 = 103$ vértices en total; y por lo tanto, tiene $103 - 1 = 102$ aristas. El número de hojas será el número total de vértices menos los vértices internos, es decir, $103 - 34 = 69$. □

EJEMPLO 3.2.88 Si sabemos que un árbol completo de aridad 5 tiene 817 hojas podemos saber cuántos vértices internos tiene. Si n es el número total de vértices y i es el número de vértices internos, entonces sabemos que $n = i + 817$ y que $n = 5i + 1$; eliminando la variable n, podemos despejar el valor de i:

$$i + 817 = 5i + 1 \qquad \Longrightarrow \qquad 816 = 4i \qquad \Longrightarrow \qquad i = \frac{816}{4} = 204$$ □

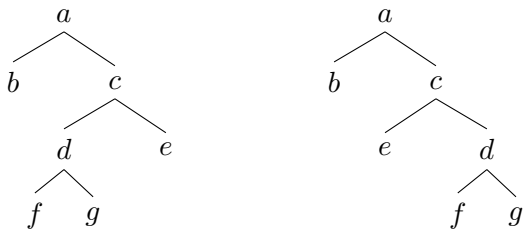
EJEMPLO 3.2.89 En un aula necesitamos conectar 25 ordenadores a un único enchufe de pared. Para ello, disponemos de cables de extensión con con cuatro salidas cada uno. ¿Cuál es el número mínimo de estos cables que necesitamos?

La disposición de los cables, sea cual sea la configuración, tendrá la forma de un árbol con raíz (el aula) que será de aridad 4 y será completo. Dado que tenemos que conectar 25 ordenadores, necesitaremos que el árbol formado tenga 25 hojas, ya que los vértices internos se utilizan para conectar los propios cables. Además, en este caso, el número de cables coincidirá con el número de vértices internos de la configuración. Repetimos el desarrollo que hemos hecho en el ejemplo anterior para calcular este número de vértices internos:

$$i + 25 = 4i + 1 \qquad \Longrightarrow \qquad 24 = 3i \qquad \Longrightarrow \qquad i = \frac{24}{3} = 8$$ □

Árboles con raíz ordenados Un **árbol con raíz ordenado** es un árbol con raíz en el que los hijos de cada vértice interno están ordenados. Este orden se refleja en su representación, disponiéndolos de izquierda a derecha según este orden.

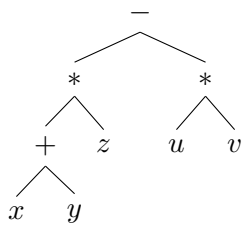
Por ejemplo, los siguientes árboles son iguales (isomorfos) si los consideramos como grafos, pero no son iguales si los consideramos con árboles con raíz ordenados, ya que la hoja e y el subárbol con raíz d están dispuestos en distinto orden.



Los árboles con raíz ordenados se usan para representar expresiones algebraicas, enunciados, expresiones gramáticales,... En estos casos, las hojas del árbol se etiquetan con variables o constantes de algún dominio numérico o semántico, y los vértices interiores se etiquetan con operadores definidos en ese dominio. Es fundamental considerar el orden en un árbol cuando consideramos operaciones que no son conmutativas (como la diferencia o la división entre números), pero también si queremos forzar un orden en la evaluación.

En un árbol ordenado, los hijos de los vértices binarios se denominan **hijo izquierdo**, el primero, e **hijo derecho**, el segundo; además, los correspondiente subárboles se denominan **subárbol izquierdo** y **subárbol derecho** respectivamente.

Ejemplo 3.2.90 La expresión algebraica $\Big(\big((x+y)*z\big)-\big(u*v\big)\Big)$ se representa con el siguiente árbol ordenado:



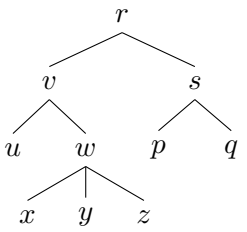
□

Una operación importante cuando trabajamos con árboles ordenados es el recorrido de sus nodos, ya sea para la evaluación en el caso de representar expresiones algebraicas o para la búsqueda de información representada con este tipo de árboles.

Recorrido en orden previo. Este recorrido corresponde a la búsqueda en anchura, que ya hemos aprendido anteriormente, empezando desde la raíz.

- Si T solo consta de la raíz, entonces r es el *recorrido en orden previo* de T.
- En otro caso, si $v_1, v_2, ..., v_k$ son los hijos de r (leídos de izquierda a derecha) en T y $T_1, T_2, ..., T_k$ los subárboles correspondientes, el recorrido en orden previo empieza por visitar la raíz r continúa recorriendo T_1 en orden previo, después T_2 en orden previo y así sucesivamente hasta que T_k.

Ejemplo 3.2.91 El recorrido en orden previo dará la secuencia de vértices en el orden dado por ese recorrido. Vamos a obtener esta secuencia para el siguiente árbol.



Representamos por T_α el subárbol con raíz en α que está pendiente de recorrido.

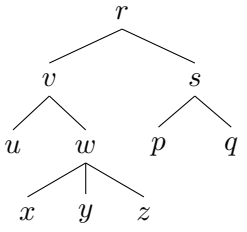
$$r - T_v - T_s$$
$$r - v - u - T_w - T_s$$
$$r - v - u - w - x - y - z - T_s$$
$$r - v - u - w - x - y - z - s - p - q$$

Aunque mostramos tres secuencias previas a la que definitivamente da el recorrido, esta última secuencia debe obtenerse directamente. □

Recorridos en orden posterior. Este recorrido corresponde a la búsqueda en profundidad, pero recorriendo el árbol desde las hojas a la raíz.

- Si T solo consiste en la raíz, entonces r es el *recorrido en orden posterior* de T.
- En otro caso, si $v_1, v_2, ..., v_k$ son los hijos de r (leídos de izquierda a derecha) en T y $T_1, T_2, ..., T_k$ los subárboles correspondientes, entonces el recorrido en orden posterior empieza por recorrer T_1 en orden posterior, después T_2 en orden posterior, así sucesivamente hasta que T_k, y termina visitando la raíz r.

Ejemplo 3.2.92 El recorrido en orden posterior dará la secuencia de vértices en el orden dado por ese recorrido. Vamos a obtener esta secuencia para el siguiente árbol.



Representamos por T_α el subárbol con raíz en α que está pendiente de recorrido.

$$T_v - T_s - r$$
$$u - T_w - v - T_s - r$$
$$u - x - y - z - w - v - T_s - r$$
$$u - x - y - z - w - v - p - q - s - r$$

Aunque mostramos tres secuencias previas a la que definitivamente da el recorrido, esta última secuencia debe obtenerse directamente. □

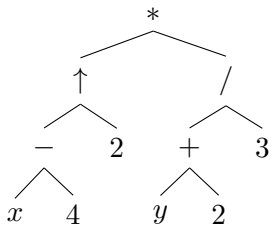
Se pueden realizar otros recorridos en un árbol dependiendo de la aplicación con la que estemos trabajando. Por ejemplo, en los árboles binarios se puede usar el recorrido en *orden simétrico* y en general también podemos utilizar las búsquedas en profundidad para determinar los recorridos, ya sea de la raíz a las hojas o de las hojas a la raíz.

Una de las aplicaciones de los recorridos que acabamos de introducir es obtener la representación de expresiones algebraicas, y expresiones sintácticas en general, sin necesidad de usar paréntesis y situando los operadores ya sea de forma prefija o de forma postfija o sufija.

Ejemplo 3.2.93 La expresión algebraica $(x-4)^2\Big(\dfrac{y+2}{3}\Big)$ se escribe de la siguiente forma usando operadores binarios y paréntesis

$$\big((x-4)\uparrow 2\big) * \big((y+2)/3\big)$$

y se representa por el siguiente árbol con raíz ordenado



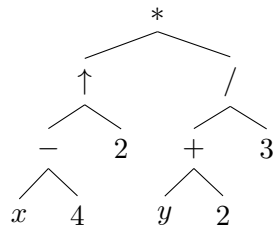
El recorrido en orden previo está dado por la secuencia

$$* \uparrow - x\, 4\, 2\, / + \, y\, 2\, 3$$

que determina la fórmula sin ambigüedades si consideramos que los operadores están escritos de forma prefija. □

Ejemplo 3.2.94 La **Notación polaca** recibe su nombre de la escuela polaca de lógicos, que escribían los operadores lógicos de forma postfija, es decir, primero los argumentos y luego el operador. En términos de recorridos de árbol, esto corresponde a escribir el recorrido del árbol sintáctico en orden posterior.

Volvamos a considerar la expresión algebraica $\big((x-4)\uparrow 2\big) * \big((y+2)/3\big)$ representada por el siguiente árbol con raíz ordenado



El recorrido en orden posterior está dado por la secuencia

$$x\,4 - 2 \uparrow y2 + 3/*$$

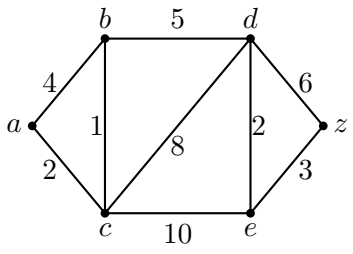
que determina la fórmula sin ambigüedades si consideramos que los operadores están escritos de forma postfija o sufija. □

3.2.12. Grafos Ponderados

DEFINICIÓN 3.2.95 *Un **grafo (simple) ponderado** es una terna $G = (V, E, w)$ tal que $G = (V, E)$ un grafo simple y $w\colon E \to \mathbb{R}^+$ es una aplicación, que se denomina función **peso**.*

Los grafos ponderados permiten representar muchos problemas (redes de transporte en donde el peso representa la distancia, redes de comunicación en donde el peso representa el coste,...) Se pueden considerar grafos ponderados más generales, por ejemplo considerando pesos sobre un grafo dirigido o sobre un multigrafo o también considerando pesos negativos. En este curso, solo vamos a trabajar con grafos simples ponderados.

Cuando representamos gráficamente un grafo ponderado, etiquetamos cada arista con el peso asignado por la aplicación w.



(3.1)

También podemos describir una grafo ponderado utilizando su **matriz de pesos**: a partir de una ordenación de los vértices, el elemento (i, j) de la matriz es el peso de la arista que une el i-ésimo vértices con el j-ésimo vértice, y es un 'guión', $-$,

si los vértices no están conectados. El grafo de arriba se describe por la siguiente matriz de pesos, tomando el orden lexicográfico como el orden entre los vértices:

$$W_G = \begin{pmatrix} - & 4 & 2 & - & - & - \\ 4 & - & 1 & 5 & - & - \\ 2 & 1 & - & 8 & 10 & - \\ - & 5 & 8 & - & 2 & 6 \\ - & - & 10 & 2 & - & 3 \\ - & - & - & 6 & 3 & - \end{pmatrix} \tag{3.2}$$

La **longitud** de un camino en un grafo ponderado es la suma de los pesos de las aristas. Por ejemplo, la longitud del camino $a - b - d - c - e - z$ en el grafo (3.1) es 30.

Un **árbol generador minimal** de un grafo conexo ponderado es un árbol generador tal que la suma de los pesos de sus aristas es mínima, respecto de todos los árboles generadores. Los algoritmos más usados para determinar el árbol generador minimal de un grafo ponderado son el **Algoritmo de Kruskal** y el **Algoritmo de Prim**; en este curso, vamos a ver solamente este último.

Algoritmo de Prim. Sea $G = (V, E, w)$ un grafo conexo ponderado con n vértices. En el algoritmo, vamos a hallar una secuencia de árboles $A_i = (V_i, E_i, w)$, cada uno de ellos subgrafo de G y de tal forma que A_n es el árbol generador minimal.

- Para $i = 1$, tomamos $V_1 = \{v\}$, siendo v un vértice cualquiera, y $E_1 = \varnothing$.
- Supongamos que hemos construido V_{i-1} y E_{i-1}. Sea

  $$w_i = \text{mín}\{w(\{x, y\}); x \in V_{i-1}, y \in V - V_{i-1}\}$$

  y $e_i = \{x_i, y_i\}$ una arista en la que se alcanza ese mínimo, es decir, $x_i \in V_{i-1}$, $y_i \in V - V_{i-1}$, $w(\{x_i, y_i\}) = w_i$.

  Tomamos $V_i = V_{i-1} \cup \{y_i\}$, $E_i = E_{i-1} \cup \{e_i\}$.

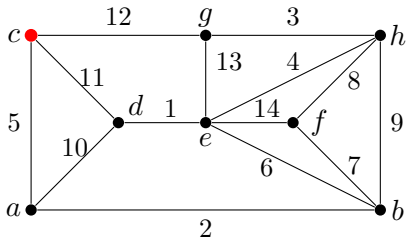
Ejemplo 3.2.96 El estudio de localización de terminales de ordenadores que van a ser instalados en una empresa viene dado por la siguiente tabla, donde los números representan el coste de instalar las conexiones entre los distintos terminales.

	a	b	c	d	e	f	g	h
a	–	2	5	10	–	–	–	–
b	2	–	–	–	6	7	–	9
c	5	–	–	11	–	–	12	–
d	10	–	11	–	1	–	–	–
e	–	6	–	1	–	14	13	4
f	–	7	–	–	14	–	–	8
g	–	–	12	–	13	–	–	3
h	–	9	–	–	4	8	3	–

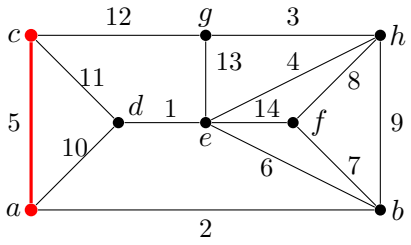
El terminal c corresponde al ordenador principal y el resto de los terminales deben estar conectados a él mediante líneas telefónicas. ¿Cuáles son las conexiones que debemos hacer para que todos los terminales estén conectados a través de c y la inversión realizada sea mínima?

La que necesitamos es obtener el árbol generador minimal, con raíz c, del grafo ponderado determinado por las conexiones entre terminales con su coste.

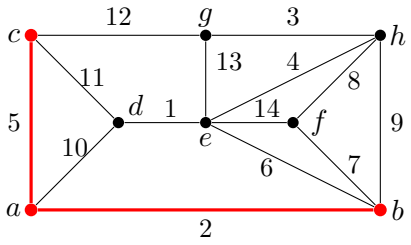
$V_1 = \{c\}$
$E_1 = \varnothing$



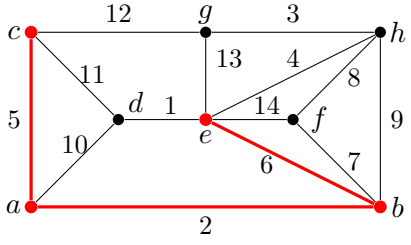
$w_2 = \text{mín}\{5, 11, 12\} = 5 = w(\{c, a\})$
$V_2 = \{c, a\}$
$E_2 = \{\{c, a\}\}$



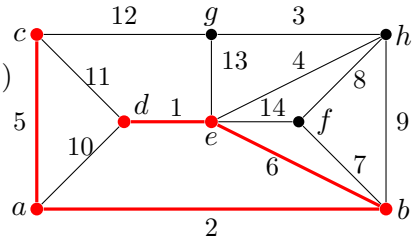
$w_3 = \text{mín}\{2, 10, 11, 12\} = 2 = w(\{a, b\})$
$V_3 = \{c, a, b\}$
$E_3 = E_2 \cup \{\{a, b\}\}$



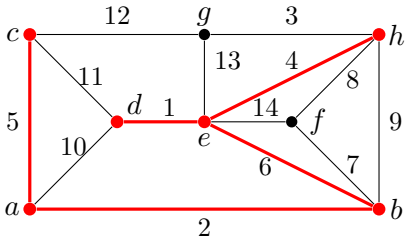
$w_4 = \text{mín}\{6, 7, 9, 10, 11, 12\} = 6 = w(\{b, e\})$
$V_4 = \{c, a, b, e\}$
$E_4 = E_3 \cup \{\{b, e\}\}$



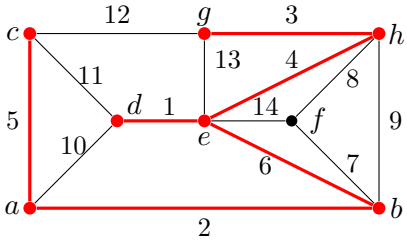
$w_5 = \text{mín}\{1, 4, 7, 9, 10, 11, 12, 13, 14\} = 1 = w(\{e, d\})$
$V_5 = \{c, a, b, e, d\}$
$E_5 = E_4 \cup \{\{e, d\}\}$



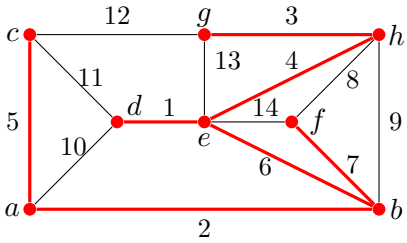
$w_6 = \text{mín}\{4, 7, 9, 12, 13, 14\} = 4 = w(\{e, h\})$
$V_6 = \{c, a, b, e, d, h\}$
$E_6 = E_5 \cup \{\{e, h\}\}$



$w_7 = \text{mín}\{3, 7, 8, 9, 12, 13, 14\} = 3 = w(\{h, g\})$
$V_7 = \{c, a, b, e, d, h, g\}$
$E_7 = E_6 \cup \{\{h, g\}\}$



$w_8 = \text{mín}\{7, 8, 14\} = 7 = w(\{b, f\})$
$V_8 = \{c, a, b, e, d, h, g, f\}$
$E_8 = E_7 \cup \{\{b, f\}\}$



El subgrafo resaltado en rojo en el último dibujo es el árbol generador minimal. □

Ejemplo con Maxima 3.2.97 También disponemos de operadores para trabajar con grafos ponderados. Para definirlos, usamos también el operado `create_graph`, pero en este caso, las aristas se definirán con una lista de dos elementos, siendo el primero de ellos la arista propiamente dicha y el segundo el peso de la misma:

```
(%i1) load(graphs)$
(%i2) grp:create_graph([1,2,3,4,5,6],[
          [[1,2],2],
          [[1,3],3],
```

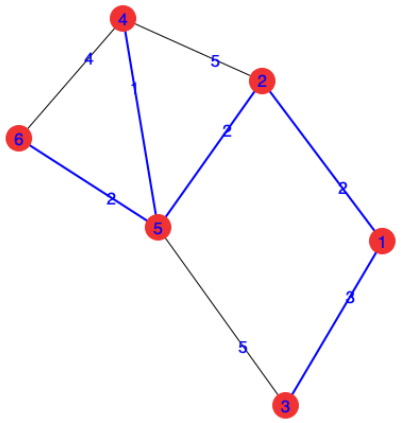
```
[[2,4],5],
[[2,5],2],
[[3,5],5],
[[4,5],1],
[[4,6],4],
[[5,6],2]
]);
```

Podemos calcular árboles generadores minimales usando el operador `minimun_spaning_tree`.

```
(%i3) mstgrp: minimum_spanning_tree(grp);
```

Vamos a utilizar las opciones `show_edges` y `show_weight` para visualizar el grafo con los pesos de cada arista y destacando el subárbol generador minimal.

```
(%i4) draw_graph(grp, vertex_size=4,show_id=true,
                 show_weight=true,
                 show_edges=edges(mstgrp));
(%o88)  done
```



□

Camino de longitud mínima: Algoritmo de Dijkstra. Nos planteamos ahora el problema de encontrar el camino de longitud mínima entre dos vértices de un grafo ponderado, y para ello vamos a utilizar el Algoritmo de Dijkstra.

Dado un grafo ponderado $G = (V, E, w)$ y un vértice $v_0 \in V$, el algoritmo de Dijkstra es un proceso iterativo tal que, en cada iteración, va a determinar el camino más corto de v_0 a un segundo vértice.

En cada paso, partimos del conjunto de vértices S_i, para los cuales hemos encontrado el camino más corto desde v_0 en algún paso anterior al i-ésimo. También

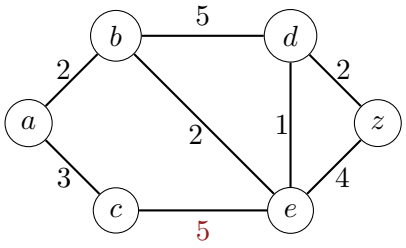
determinamos la función ℓ_i, tal que $\ell_i(v)$ es la longitud del camino más corto desde v_0 hasta v, pasando exclusivamente por vértices de S_i; escribiremos $\ell_i(v) = -$ si $v \in S_i$ y $\ell_i(v) = \infty$ si v no es adyacente a ningún vértice de S_i.

- $S_1 = \{v_0\}$;
  $\ell_1(v_0) = -$, $\ell_1(v) = w(\{v_0, v\})$ si $\{v_0, v\} \in E$, $\ell_1(v) = \infty$ en otro caso.
  Si $\ell_1(v_1) = \text{mín}\{\ell_1(v); v \in V\}$, entonces $S_2 = \{v_0, v_1\}$ y la arista $\{v_0, v_1\}$ es el camino más corto de v_0 a v_1.
- Supongamos que hemos determinado $S_i = S_{i-1} \cup \{v_{i-1}\}$ y ℓ_{i-1}.

$$\ell_i(v) = \begin{cases} \min\{\ell_{i-1}(v), \ell_{i-1}(v_{i-1}) + w(\{v_{i-1}, v\})\} & \text{Si } \{v, v_{i-1}\} \in E \\ \ell_{i-1}(v) & \text{Si } \{v, v_{i-1}\} \notin E \end{cases}$$

  Es decir, después de añadir el vértice v_{i-1} volvemos a calcular las distancias hasta cada vértice para ver si, utilizando el camino más corto hasta v_{i-1}, conseguimos un camino más corto a los otros vértices.
  Tomamos v_i tal que $\ell_i(v_i) = \text{mín}\{\ell_i(v); v \in V\}$; este número es la longitud del camino más corto de v a v_i.
- Continuamos hasta conseguir $S_n = V$ o hasta conseguir el camino más corto hasta el vértice deseado.

Ejemplo 3.2.98 Para el grafo



vamos a hallar el camino más corto desde el vértice a hasta cada uno de los otros vértices. Para seguir el desarrollo del algoritmo, vamos a construir una tabla, en cuya primera columna vamos a escribir los elementos de los conjuntos S_i, destacando el último vértice añadido. El resto de las columnas corresponde a cada vértice del grafo, de tal forma que las filas estarán ocupadas con el correspondiente valor de la función ℓ_i.

	Vértices	a	b	c	d	e	z	
1	$\{a\}$	$-$	$\boxed{2}(a)$	$3(a)$	∞	∞	∞	$a-b$

Los vértices b y c son los únicos adyacentes a a, y por eso son los únicos que aparecen en la primera fila con un número, el peso de la correspondiente arista.

Además, hemos indicado entre paréntesis el vértice anterior en el camino que permite obtener esta longitud; naturalmente, en este primer paso es a. Marcamos el menor valor en la fila, que corresponde a la columna de b, es decir, el camino más corto desde a hasta b es la aristas $a-b$.

	Vértices	a	b	c	d	e	z	
1	$\{a\}$	–	$\boxed{2}(a)$	$3(a)$	∞	∞	∞	$a-b$
2	$\{a\}\cup\{b\}$	–	–	$\boxed{3}(a)$	$7(b)$	$4(b)$	∞	$a-c$

Dado que $a, b \in S_2$, escribimos un guión en las casillas de estos vértices en la segunda fila. Dado que c y z no son adyacentes a b, para estos vértices, copiamos las casillas de la fila superior. Dado que la arista $\{b, d\}$ tiene peso 5, el camino $a-b-d$ tiene peso 7 y por eso escribimos $7(b)$ en la casilla correspondiente a d en la segunda fila. Dado que la arista $\{b, e\}$ tiene peso 2, el camino $a-b-e$ tiene peso 4 y por eso escribimos $4(b)$ en la casilla correspondiente a e en la segunda fila. Por lo tanto, el menor valor en la fila es 3 y nos indica que el camino $a-c$ es el más corto de los que unen a y c.

	Vértices	a	b	c	d	e	z	
1	$\{a\}$	–	$\boxed{2}(a)$	$3(a)$	∞	∞	∞	$a-b$
2	$\{a\}\cup\{b\}$	–	–	$\boxed{3}(a)$	$7(b)$	$4(b)$	∞	$a-c$
3	$\{a,b\}\cup\{c\}$	–	–	–	$7(b)$	$\boxed{4}(b)$	∞	$a-b-e$

Dado que $a, b, c \in S_3$, escribimos un guión en las casillas de estos vértices en la tercera fila. Dado que d y z no son adyacentes a c, copiamos las casillas de la fila superior en las columnas correspondientes a estos vértices. Dado que la arista $\{c, e\}$ tiene peso 5, el camino $a-c-e$ tiene peso 8, que es mayor que el dado por la casilla superior, por eso volvemos a escribir $4(b)$ en la casilla correspondiente a e en la tercera fila.

El valor menor en la tercera fila es $4(b)$, en la columna de e. Esto indica que 4 es la longitud del camino más corto que une a y e; además, este camino viene de b, es decir, $a-b-e$ es el camino más corto desde a hasta e.

	Vértices	a	b	c	d	e	z	
1	$\{a\}$	–	$\boxed{2}(a)$	$3(a)$	∞	∞	∞	$a-b$
2	$\{a\}\cup\{b\}$	–	–	$\boxed{3}(a)$	$7(b)$	$4(b)$	∞	$a-c$
3	$\{a,b\}\cup\{c\}$	–	–	–	$7(b)$	$\boxed{4}(b)$	∞	$a-b-e$
4	$\{a,b,c\}\cup\{e\}$	–	–	–	$\boxed{5}(e)$	–	$8(e)$	$a-b-e-d$

Dado que $a, b, c, e \in S_4$, escribimos un guión en las casillas de estos vértices en la segunda fila. Dado que la arista $\{e, d\}$ tiene peso 1, la longitud del camino $a-b-e-d$

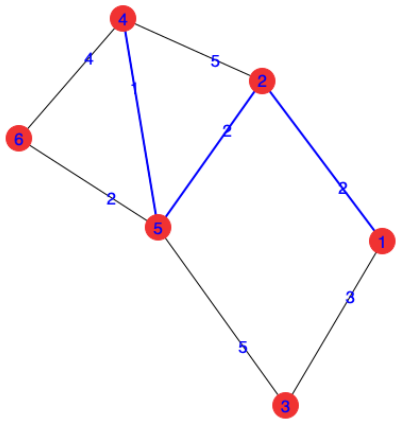
es 5, que es menor que el camino indicado en la fila superior, por eso escribimos $5(e)$ en la casilla correspondiente a d en la cuarta fila. Dado que la arista $\{e, z\}$ tiene peso 4, la longitud del camino $a - b - e - z$ es 8, y por eso escribimos $8(e)$ en la casilla correspondiente a z en la cuarta fila. El menor número en la cuarta fila es 5, en la columna de d; por lo tanto, 5 es la longitud del camino más corto de a hasta d; este camino viene del vértice e y por lo tanto $a - b - e - d$ es el camino más corto desde a hasta d.

	Vértices	a	b	c	d	e	z	
1	$\{a\}$	–	$\boxed{2}(a)$	$3(a)$	∞	∞	∞	$a - b$
2	$\{a\} \cup \{b\}$	–	–	$\boxed{3}(a)$	$7(b)$	$4(b)$	∞	$a - c$
3	$\{a, b\} \cup \{c\}$	–	–	–	$7(b)$	$\boxed{4}(b)$	∞	$a - b - e$
4	$\{a, b, c\} \cup \{e\}$	–	–	–	$\boxed{5}(e)$	–	$8(e)$	$a - b - e - d$
5	$\{a, b, c, e\} \cup \{d\}$	–	–	–	–	–	$\boxed{7}(d)$	$a - b - e - d - z$

En este último paso, solo tenemos que calcular la longitud del camino hasta z sumando 5 al peso de la arista $\{d, z\}$, es decir, $5 + 2 = 7$, que es menor que la indicada en la casilla superior. Por lo tanto, el camino $a - b - e - d - z$ es el más corto desde a hasta z. □

Ejemplo con Maxima 3.2.99 El algoritmo de Dijkstra está implementado en `Maxima` en el operador `shortest_weighted_path`:

```
(%i1) load(graphs)$
(%i2) grp:create_graph([1,2,3,4,5,6],[
          [[1,2],2],
          [[1,3],3],
          [[2,4],5],
          [[2,5],2],
          [[3,5],5],
          [[4,5],1],
          [[4,6],4],
          [[5,6],2]
          ])$
(%i3) dijkgrp: shortest_weighted_path(1,4,grp);
(%o3) [5,[1,2,5,4]]
(%i4) draw_graph(grp,vertex_size=4,show_id=true,
                 show_weight=true,
                 show_edges=vertices_to_path(dijkgrp[2]));
(%o86)  done
```


4
5
2
4
1
6
2
2
2
5
1
3
5
3

□

Relación de ejercicios 6

1. En el conjunto $A = \{a, b, c, d\}$ se establece una relación binaria

$$\mathcal{R} = \{(a, b), (b, d), (c, b), (d, a)\}$$

Estudia qué propiedades cumple la relación binaria $\mathcal{R}$.

2. En el conjunto $A = \{1, 2, 3, 4\}$ se establece una relación binaria

$$\mathcal{R} = \{(1, 1), (2, 2), (2, 3), (2, 4), (3, 2), (3, 3), (3, 4), (4, 2), (4, 3), (4, 4)\}$$

Justifica que $\mathcal{R}$ es una relación de equivalencia y halla el conjunto cociente.

3. Utiliza el algoritmo de Warshall para hallar el cierre transitivo de las siguientes relaciones:

   a) $\mathcal{R} = \{(a, b), (a, d), (b, c), (b, d), (d, a), (d, d)\}$ definida en $A = \{a, b, c, d\}$.

   b) $\mathcal{S} = \{(a, b), (b, c), (c, b), (d, b), (d, e), (e, a)\}$ definida en $A = \{a, b, c, d, e\}$.

4. En el conjunto $A = \{1, 2, 3, 4, 5\}$ se define la relación $\mathcal{R} = \{(1, 2), (3, 4), (5, 2)\}$.

   a) Usa el algoritmo de Warshall para hallar la mínima relación de equivalencia que contiene a $\mathcal{R}$.

   b) Determina la partición inducida por dicha relación.

5. En el conjunto $A = \{1, 2, 3, 4, 5\}$ se define la relación

$$\mathcal{R} = \{(1, 2), (2, 1), (3, 3), (4, 5)\}$$

   a) Prueba que $S = \mathcal{R} \cup \mathcal{R}^2$ es una relación transitiva.

   b) Usa el algoritmo de Warshall para hallar la mínima relación de equivalencia que contiene a S y determina el conjunto cociente.

6. En el conjunto $A = \{2, 3, 4, 5, 6\}$ se establece la relación binaria $\mathcal{R}$ definida de la siguiente forma

$$x \mathcal{R} y \iff \text{mcd}(x, y) = 1$$

   a) Escribe el conjunto de pares ordenados de $\mathcal{R}$ y halla la matriz asociada.

   b) Calcula r($\mathcal{R}$), s($\mathcal{R}$) y t($\mathcal{R}$).

Relación de ejercicios 7

1. Dibuja, si existen, grafos simples de cuatro vértices que tengan grados respectivos:

   a) 2, 2, 2 y 4; *b*) 2, 1, 2 y 1; *c*) 2, 2, 2 y 3.

2. Dibuja, si existen, grafos simples de cinco vértices que tengan grados respectivos:

   a) 1, 2, 3, 1 y 5; *b*) 0, 1, 2, 3 y 4; *c*) 2, 2, 2, 3 y 3.

3. Da ejemplos, si existen, de:

   a) Un grafo completo con 36 aristas.

   b) Un grafo bipartito completo $K_{m,12}$ con 72 aristas.

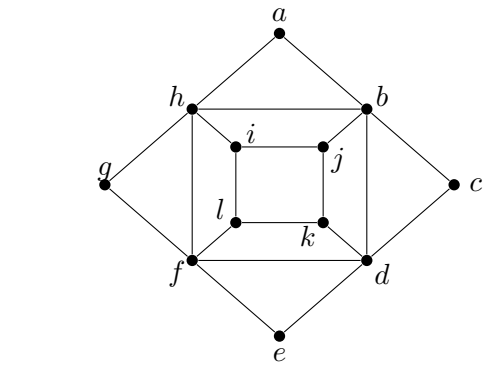
4. Sea G el grafo dado por

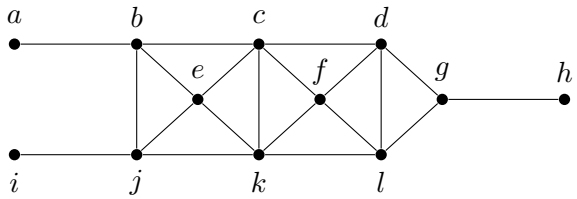
a	b	c	d	e	f
b	a	b	a	b	a
d	c	d	c	d	c
f	e	f	e	f	e

   Estudia si G es:

   a) bipartito, *b*) conexo, *c*) euleriano, *d*) hamiltoniano.

5. Para cada uno de los grafos siguientes, halla árboles generadores haciendo una búsqueda en anchura y haciendo una búsqueda en profundidad.





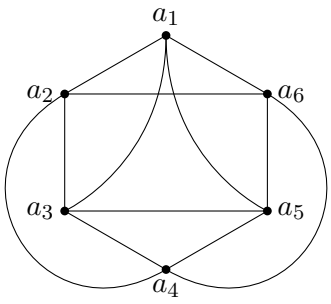
6. Considera el grafo dado por las listas de adyacencia

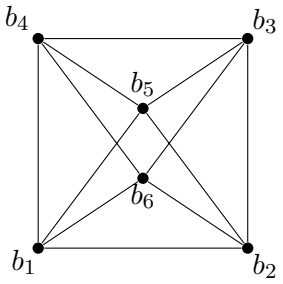
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	1	4	3	1	2	6	9	3	4	6	7	8	9	12	13
5	6	9	10	6	5	12	13	8	9	12	11	12	13	16	15
					7			10			13	14			
					11			14			15	16			

   a) Determina si es conexo haciendo una búsqueda en profundidad y representa la secuencia de búsqueda de los vértices con un árbol con raíz.

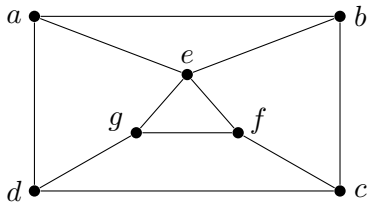
   b) Repite el apartado anterior pero mediante una búsqueda en anchura.

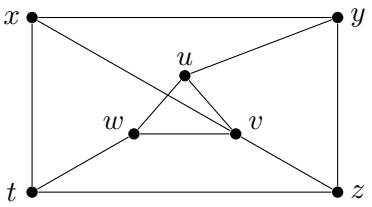
7. Estudia si los siguientes grafos son isomorfos y, en tal caso, determina un isomorfismo entre ellos.



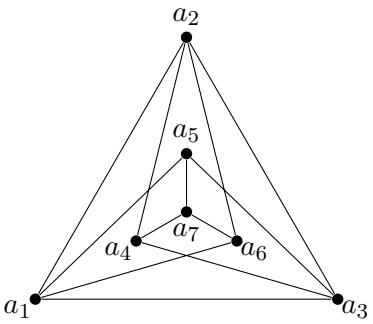


8. Estudia si los siguientes grafos son isomorfos y, en tal caso, determina un isomorfismo entre ellos:





9. Sea G el grafo de la siguiente figura:

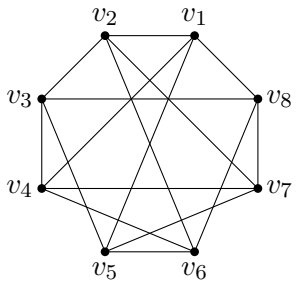


a) Estudia si G tiene un camino o un circuito de Euler. En caso afirmativo, utiliza un algoritmo adecuado para determinarlo.

b) Estudia si G tiene un camino o un ciclo de Hamilton. En caso afirmativo, determínalo.

c) Sea H el subgrafo de G obtenido al eliminar la arista $\{a_5, a_7\}$. Estudia si H tiene un camino o un circuito de Euler y en caso afirmativo utiliza un algoritmo adecuado para determinarlo.

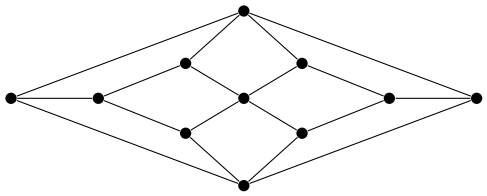
10. Sea G el grafo de la siguiente figura:



a) Estudia si G contien un camino o un ciclo de Hamilton y en caso afirmativo determinalo.

b) Estudia si G es bipartito.

11. Estudia si el siguiente grafo es o no es hamiltoniano:



12. Un grafo simple tiene 16 aristas y sus vértices tienen grado 3 ó 4. ¿Cuántos vértices de grado 3 y cuántos de grado 4 debe tener? Indica todas las soluciones posibles.

13. Estudia para qué valores de n los grafos C_n, K_n y $K_{n,n}$ tienen un circuito de Euler.

14. ¿Para qué valores de m y n el grafo $K_{m,n}$ tiene un circuito de Euler? ¿Y un ciclo de Hamilton?

15. En cada uno de los apartados siguientes dibuja un ejemplo de un grafo simple conexo G, con 5 ó 6 vértices, que verifique las condiciones que se indican:

a) G es euleriano y hamiltoniano.

b) G es euleriano, pero no es hamiltoniano.

c) G no es euleriano pero sí es hamiltoniano.

d) G no es euleriano ni hamiltoniano.

e) G tiene un camino euleriano, pero no tiene un camino hamiltoniano.

f) G no tiene un camino euleriano, pero sí tiene un camino hamiltoniano.

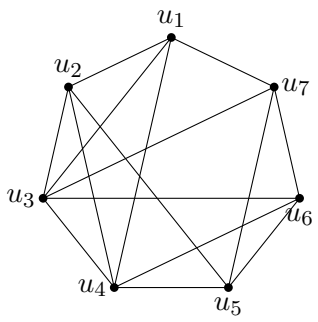
g) G tiene un camino euleriano y un camino hamiltoniano.

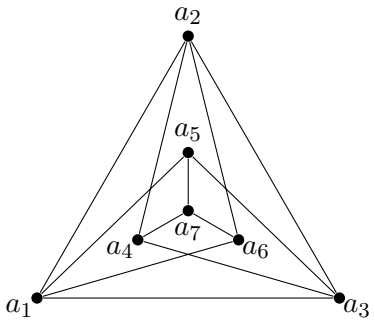
h) G no tiene un camino euleriano ni un camino hamiltoniano.

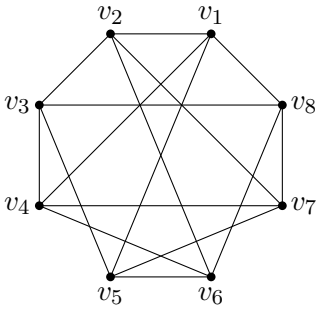
Relación de ejercicios 8

1. Dibuja, si es posible, un grafo plano con 8 aristas, 7 vértices y 4 regiones. Si no es posible, justifícalo.

2. Estudia si los siguientes grafos son planos:







3. Se desea diseñar una placa con 6 componentes electrónicos, c_1, c_2, c_3, c_4, c_5 y c_6, de manera que no se corten las pistas y que todos estén conectados entre sí, salvo c_1 con c_3 y c_4 con c_6. Razona si es posible.

4. En un laboratorio hay una serie de compuestos químicos, $1, 2, 3, 4, 5, 6, 7, 8$ que hay que almacenar en cajas para su traslado. No pueden ser almacenados en una misma caja dos compuestos que reaccionen entre sí (como ácidos y bases). Los productos que reaccionan vienen dados por la siguiente tabla:

1	2	3	4	5	6	7	8
2	1	1	1	2	2	3	2
3	5	4	3	4	5	5	5
4	6	7	5	6	8	8	6
	8			7			7
				8			

¿Cómo podemos elegir los elementos que hemos de introducir en cada caja? ¿Cuántas cajas serán necesarias para poder trasladar los productos?

5. El jefe de estudios de una escuela tiene que programar las fechas de los exámenes finales de 7 asignaturas: $a_1, a_2, a_3, a_4, a_5, a_6, a_7$. Se sabe que los siguientes pares de asignaturas tienen alumnos en común:

$$\{a_1, a_2\}, \{a_1, a_3\}, \{a_1, a_4\}, \{a_1, a_7\}, \{a_2, a_3\}, \{a_2, a_4\}, \{a_2, a_5\}, \{a_2, a_7\},$$
$$\{a_3, a_4\}, \{a_3, a_6\}, \{a_3, a_7\}, \{a_4, a_5\}, \{a_4, a_6\}, \{a_5, a_6\}, \{a_5, a_7\}, \{a_6, a_7\}$$

   ¿Cuántos días son necesarios para realizar todos los exámenes de modo que ningún estudiante tenga dos exámenes el mismo día?

6. Dibuja, si es posible, el grafo que corresponde a cada una de las propiedades descritas a continuación. Si no es posible explicar por qué:

   a) Un grafo cuyo número de vértices sea igual al número de aristas más uno y no sea un árbol.

   b) Un árbol con 5 vértices con grados: 1, 1, 2, 2, 4.

   c) Un árbol con 4 vértices internos y 6 hojas.

   d) Un árbol binario completo con 9 hojas y altura 3.

7. Un árbol cuaternario completo tiene 27 vértices internos. ¿Cuántas aristas tiene? ¿Cuántas hojas?

8. ¿Cuántos vértices internos tiene un árbol ternario completo con 817 hojas?

9. ¿Cuál es el número máximo de vértices internos que puede tener un árbol cuaternario completo de altura 8?

10. En una compañía donde trabajan 125 ejecutivos se instala un nuevo sistema de comunicación telefónica. Lo inaugura la presidenta, quien llama a sus cuatro vicepresidentes. A continuación, cada vicepresidente llama a otros cuatro ejecutivos; éstos, a su vez, a otros cuatro y así sucesivamente.

    a) ¿Cuántas llamadas son necesarias para comunicar con los 125 ejecutivos?

    b) ¿Cuántos ejecutivos hacen llamadas?

11. Consideramos la expresión: $((p_1 \vee p_2) \to q) \leftrightarrow ((p_1 \to q) \wedge (p_2 \to q))$

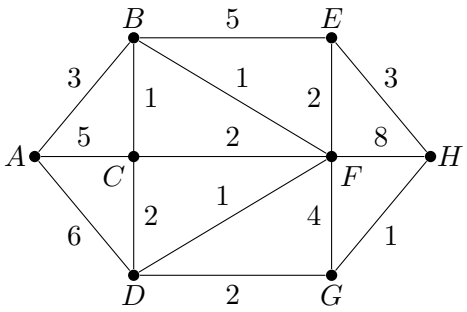
    a) Represéntala mediante un árbol con raíz ordenado.

    b) Determina los recorrido en orden previo y en orden posterior del árbol anterior.

12. Traza un árbol binario para la expresión postfija que sigue y escribe su representación prefija:

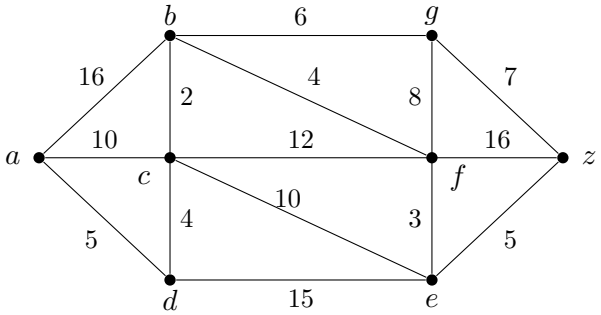
$$AB + CD * EF/ - -A*$$

13. Usa el algoritmo de Prim para hallar un árbol generador minimal del grafo de la siguiente figura:



14. El grafo del ejercicio anterior muestra la conexión entre 8 centros de comunicación. Los vértices representan a los centros y las aristas a los canales de comunicación. Los tiempos de transmisión están representados por los pesos de las aristas. Supongamos que a las 7:00 el centro de comunicaciones A transmite una noticia a través de todos sus canales. Los otros centros transmitirán esa noticia tan pronto como la reciban. Usa el algoritmo de Dijkstra para determinar el menor tiempo en que cada uno de los centros B, C, D, E, F, G y H recibe la noticia.

15. En el grafo de la figura se representa una red ferroviaria donde la distancia entre cada par de ciudades se expresa en km:



   a) Halla el camino más corto para viajar de a hasta z.

   b) Se quiere renovar la red ferroviaria de manera que el coste en km sea mínimo y que cada par de ciudades tenga conexión por tramos renovados. ¿Qué tramos deben renovarse?

TEMA 4

Lógica Clásica Proposicional

4.1. Lógica y Computación

La **Lógica** es la ciencia que tiene como objetivo el análisis de los métodos de razonamiento. Es decir, la lógica analiza la validez de las construcciones del lenguaje natural que llamamos **razonamientos** o **inferencias**

Si ... y ... y ... y ..., entonces ...

Estos esquemas se utilizan para establecer que cierta afirmación, llamada conclusión, se deduce o infiere de otras, llamadas hipótesis. Es importante tener en cuenta que la Lógica se interesa por la *forma* y no por el contenido de los razonamientos.

De forma más concreta, la lógica quiere construir **sistemas formales** como herramienta para contestar a la siguiente cuestión: ¿es correcto un razonamiento como el siguiente?

> *Si hay petróleo en Poligonia, entonces o los expertos tienen razón o el gobierno está mintiendo. No hay petroleo en Poligonia, o si no los expertos se equivocan. Así pues, el gobierno no está mintiendo.*

En este curso, estamos interesados en la lógica como sistema deductivo, es decir, para describir, implementar y mecanizar tareas donde interviene la capacidad deductiva del hombre y en las que se requiere tener conocimiento sobre el dominio, razonar con tal conocimiento y conocer cómo dirigir o guiar tal razonamiento.

En informática, la lógica se usa además como modelo de computación y es la base de un paradigma de programación llamado **programación lógica**. También nos la encontramos en otras áreas como: análisis, síntesis y verificación de programas, teoría de la especificación, inteligencia artificial, control de procesos, robótica, entre otros.

En este curso y en este tema, nos vamos a centrar en la **Lógica Clásica**, la lógica fundamental y más importante, que tiene las siguientes características:

- Considera únicamente construcciones declarativas, sobre las que podemos pronunciarnos acerca de su **verdad** o **falsedad**.
- No tiene en cuenta aspectos de contexto (tiempo, posibilidad, creencia,...)
- La validez de las construcciones compuestas queda determinada por funciones de verdad sobre las afirmaciones elementales.

En esta lógica, como en muchas otras, el estudio se realiza en dos niveles de análisis estructural. La **Lógica Clásica Proposicional** considera únicamente construcciones declarativas simples y compuestas. La **Lógica Clásica de Predicados** distingue *qué* se declara y *de qué* o *de quién* se declara. En este curso, estudiaremos solamente el caso proposicional.

El resto de las lógicas se denominan genéricamente lógicas *no-clásicas* y pueden extender a la Lógica Clásica o desviarse de ella en alguna de sus características. La *Lógica Temporal* considera contextos temporales; la *Lógica Modal* considera contextos de necesidad o posibilidad; la *Lógica Doxástica* considera contextos de creencia; la *Lógica Intuicionista* no incluye la ley del tercero excluido ("A o no A"); las *Lógicas multivaluadas* consideran tres o más valores de verdad; la *Lógica difusa* establece de forma "difusa" la distinción entre verdad y falsedad.

Una lógica o sistema lógico queda determinado con los siguientes elementos:

1. Un **Lenguaje Formal**, que usamos para representar los razonamientos en lenguaje natural como esquemas formales.
2. Una **Semántica** o **Teoría de Modelos**, que dota de significado a las expresiones o fórmulas del lenguaje formal y establece, en términos semánticos, los conceptos básicos asociados a una lógica: **validez** y **satisfacibilidad**.
3. Una **Teoría de la Demostración**, que establece las nociones de validez y satisfacibilidad en términos sintácticos, es decir, a partir de transformaciones descritas en el lenguaje y basadas en reglas puramente formales.

Aunque es deseable disponer tanto de una teoría de modelos como de una teoría de demostración, es posible trabajar en sistemas lógicos definidos solamente con una de las dos. Por otra parte, desde el punto de vista computacional, es conveniente que sea posible, además, **automatizar las deducciones**, es decir, que la propiedad de validez pueda determinarse mediante algoritmos definidos a partir de la teoría de modelos o a partir de una teoría de demostración.

4.1.1. Lenguajes formales

Definición 4.1.1 (Lenguaje) *Sea Σ un conjunto, que en este contexto denominamos **alfabeto** (y cuyos elementos son los símbolos del lenguaje).*

- *Llamamos* ***Lenguaje Universal*** *sobre* Σ *al conjunto* $\Sigma^* = \bigcup_{n\in\mathbb{N}} \Sigma^n$*; es decir, el conjunto formado por todas las cadenas finitas de elementos del alfabeto.*
- *Un* ***Lenguaje*** *sobre* Σ *es cualquier subconjunto del lenguaje universal.*

Hablamos de **Lenguaje Formal** si sus elementos, es decir, las fórmulas del lenguaje, se definen sin atender al posible significado de los elementos del alfabeto. Es decir, un lenguaje formal L viene determinado por:

1. ***Alfabeto:*** Conjunto de **símbolos** admitidos en el lenguaje.
2. ***Gramática:*** Conjunto de **reglas** de formación (sintácticas) que determinan qué cadenas se consideran fórmulas bien formadas del lenguaje.

Ejemplo 4.1.2

- Lenguaje 'mg'. El alfabeto es $\{m, g, -\}$ y la gramática queda determinada por las siguientes reglas: las fórmulas son aquellas cadenas que contienen exactamente un símbolo m, exactamente un símbolo g y m aparece a la izquierda de g.
- El conjunto de los números naturales escritos en base 10 es un lenguaje formal sobre el alfabeto $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.
- El conjunto formado por los números múltiplos de tres NO es un lenguaje formal, ya que está definido usando el significado de los números. □

4.1.2. Semántica o Teoría de modelos

La semántica tiene como fin dar significado a las fórmulas del lenguaje a partir de su estructura sintáctica y establecer la noción semántica de deducción.

Definición 4.1.3 (Teoría de modelos) *Una* ***teoría de modelos*** *sobre un lenguaje* L *es una terna* $(S, D, \mathcal{I})$*, en donde* S *es un conjunto, cuyos elementos se denominan* ***valores semánticos****,* D *es un subconjunto de* S*, cuyos elementos se denominan* ***valores destacados****, e* $\mathcal{I}$ *es un conjunto de aplicaciones de* L *en* S*, cuyos elementos se denominan* ***interpretaciones****.*

Una lógica queda determinada por un lenguaje formal y una teoría de modelos sobre él.

Ejemplo 4.1.4 En un ejemplo anterior hemos definido el lenguaje mg: su alfabeto es $\{m, g, -\}$ y las fórmulas son aquellas cadenas que contienen exactamente un símbolo m, exactamente un símbolo g y m aparece a la izquierda de g. Sobre este lenguaje definimos la teoría de modelos $MG = (S, D, \mathcal{I})$ como

- Valores semánticos: $S = \mathbb{N}^3$
- Valores destacados: $D = \{(m, n, k) \in \mathbb{N}^3 \mid m + n = k\}$
- Interpretaciones: $\mathcal{I} = \{\mathrm{I}\}$ en donde, si α, β y γ son cadenas de 'guiones' de longitud arbitraria:

$$\mathrm{I}(\alpha m \beta g \gamma) = (\mathrm{long}(\alpha), \mathrm{long}(\beta), \mathrm{long}(\gamma))$$

(la función long devuelve la longitud de una cadena.)

Por ejemplo, $\mathrm{I}(--m--g-) = (2, 2, 1)$. □

Aunque en este ejemplo el conjunto de interpretaciones solo contiene una interpretación, en cualquier sistema lógico este conjunto estará formado por muchas funciones e incluso podrá ser infinito.

Las interpretaciones dan un "significado" a cada fórmula, de forma que el conjunto de interpretaciones $\mathcal{I}$ da todas las posibles "lecturas" de una misma fórmula. La noción de validez se determina a partir de los valores destacados.

Definición 4.1.5 *Sea $(S, D, \mathcal{I})$ una teoría de modelos para un lenguaje L.*

- *Decimos que $\mathrm{I} \in \mathcal{I}$ es un **modelo** de una fórmula A si $\mathrm{I}(A) \in D$. El conjunto de modelos de A se denota $\mathtt{Mod}(A)$.*
- *Decimos que $\mathrm{I} \in \mathcal{I}$ es un **contramodelo** de una fórmula A si $\mathrm{I}(A) \notin D$.*
- *Decimos que una fórmula A es **satisfacible** si alguna interpretación es modelo de A; es decir, si existe $\mathrm{I} \in \mathcal{I}$ tal que $\mathrm{I}(A) \in D$; es decir, $\mathtt{Mod}(A) \neq \varnothing$*
- *Decimos que una fórmula A es **insatisfacible** si para toda interpretación $\mathrm{I} \in \mathcal{I}$ se tiene que $\mathrm{I}(A) \notin D$; es decir, $\mathtt{Mod}(A) = \varnothing$*
- *Decimos que una fórmula A es **válida** si para toda $\mathrm{I} \in \mathcal{I}$ se tiene que $\mathrm{I}(A) \in D$; es decir, si todas las interpretaciones son modelos de A; es decir, $\mathtt{Mod}(A) = \mathcal{I}$*

Ejemplo 4.1.6 Con la teoría de modelos que hemos definido anteriormente sobre el lenguaje mg en el podemos hacer las siguientes afirmaciones:

- La fórmula $--m-g--$ no es válida en MG, ya que

$$\mathrm{I}(--m-g---) = (2, 1, 2) \notin D, \qquad (\text{porque } 2 + 1 \neq 2)$$

- La fórmula $-m--g---$ sí es válida en MG, ya que

$$\mathrm{I}(-m--g----) = (1, 2, 3) \in D, \qquad (\text{porque } 1 + 2 = 3)$$

Ahora es fácil reconocer el significado que la teoría de modelos le ha dado a los símbolos del lenguaje: m representa al operador suma, g representa a la relación de igualdad y cada guión representa una unidad. □

4.1.3. Teorías de demostración

Las teorías de demostración son mecanismos deductivos, es decir, mecanismos que permiten obtener fórmulas válidas mediante transformaciones sintácticas, descritas de manera puramente formal, a partir de otras fórmulas válidas. Los **Sistemas axiomáticos** son los sistemas de demostración más sencillos y habitualmente se considera fundamental disponer de un sistema axiomático para determinar de forma efectiva una lógica. Existen otros tipos de teorías de demostración, como la **Deducción Natural** y los **Sistemas de Gentzen**.

Ejemplo 4.1.7 Un sistema axiomático queda determinado por un conjunto de fórmulas que se consideran válidas en el sistema, y que se denominan axiomas, y una o varias reglas de inferencia, que transforman fórmulas válidas en fórmulas válidas. Por ejemplo, para el lenguaje mg que hemos utilizado en varios ejemplos, podemos definir el siguientes sistemas axiomático.

- *Axiomas*: $\alpha m g \alpha$, para cada cadena de guiones α.
- *Regla de inferencia*: de $\alpha m \beta g \gamma$ se deriva $\alpha m \beta - g \gamma -$, para cualesquiera cadenas de guiones α, β y γ.

De esta forma, la fórmula $-m-g--$ es una fórmula válida en el sistema axiomático:

$$\begin{array}{ll} 1. & -mg- \qquad \text{Ax} \\ 2. & -m-g-- \quad \text{RI}:1 \end{array}$$

No es difícil observar que todas las fórmulas válidas en este sistema axiomático lo son en la teoría de modelos MG, por eso decimos que el sistema axiomático es **correcto**.

También es fácil observar que todas las fórmulas válidas en la teoría de modelos MG pueden ser derivadas en el sistema axiomático, por eso decimos que el sistema axiomático es **completo**. □

4.2. El lenguaje de la Lógica Clásica Proposicional: Cl

El lenguaje Cl, de la Lógica Clásica Proposicional está determinado por:

Alfabeto: Conjunto de símbolos admitidos en el lenguaje:

- **variables** o **símbolos proposicionales**:

$$Q = \{p, q, r, \ldots, p_1, q_1, r_1, \ldots, p_n, q_n, r_n, \ldots\}$$

- **conectivos** u **operadores lógicos**: $\{\neg, \vee, \wedge, \rightarrow, \leftrightarrow\}$
- **delimitadores**: "(", ")".

Gramática: Conjunto de **reglas** de formación de fórmulas (bien formadas):

1. Los elementos de Q son fórmulas, que se denominan **fórmulas atómicas**.
2. Si A es una fórmula, $\neg A$ es una fórmula.
3. Si A y B son fórmulas, $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, $(A \leftrightarrow B)$ son fórmulas.
4. Sólo las cadenas obtenidas aplicando las reglas anteriores son fórmulas.

Ejemplo 4.2.1 Las siguientes cadenas de símbolos son fórmulas bien formadas de la Lógica Clásica Proposicional

$$\big((p \wedge q) \rightarrow \neg r\big) \qquad \big(p \wedge (q \rightarrow \neg r)\big) \qquad \big((p \wedge \neg r) \rightarrow q\big) \qquad \square$$

Obsérvese que las letras A y B que hemos usado en la definición anterior (y otras letras mayúsculas que utilizaremos más adelante), no son símbolos del lenguaje. Estas variables representan fórmulas del lenguaje y permiten hablar sobre ellas: se denominan **metasímbolos**. En particular, una expresión como $(A \vee B)$ no es una fórmula sino un *esquema de* fórmula.

Es frecuente reducir el número de delimitadores en las fórmulas usando convenios de simplificación. El único convenio que utilizaremos en este curso es la eliminación de los paréntesis inicial y final de una fórmula, si los tuviera; es decir, en lugar de $(p \rightarrow (q \vee r))$ escribiremos $p \rightarrow (q \vee r)$.

Un conjunto definido tal y como hemos definido el lenguaje anterior se dice que es un **conjunto inductivo**. Esta forma de definir el conjunto, determina la manera en la que se definen funciones en el lenguaje: por **recursividad**:

(i) Se define sobre las fórmulas atómicas (elementos de Q).

(ii) Se establece el comportamiento de la función con las reglas de formación.

Definición 4.2.2 (Función Grado) *La función* ***grado****,* $\mathrm{Gr}\colon Cl \to \mathbb{N}$*, devuelve el número de conectivos lógicos que aparecen en una fórmula:*

$$\begin{aligned}\mathrm{Gr}(A) &= 0 \quad \textit{para todo } A \in Q\\ \mathrm{Gr}(\neg B) &= 1 + \mathrm{Gr}(B)\\ \mathrm{Gr}(C * D) &= 1 + \mathrm{Gr}(C) + \mathrm{Gr}(D), \quad \textit{si } * \in \{\vee, \wedge, \to, \leftrightarrow\}\end{aligned}$$

Ejemplo 4.2.3

$$\begin{aligned}\mathrm{Gr}(p \to (r \vee \neg q)) &= 1 + \mathrm{Gr}(p) + \mathrm{Gr}(r \vee \neg q)\\ &= 1 + \mathrm{Gr}(p) + 1 + \mathrm{Gr}(r) + \mathrm{Gr}(\neg q)\\ &= 1 + \mathrm{Gr}(p) + 1 + \mathrm{Gr}(r) + 1 + \mathrm{Gr}(q) = 3\end{aligned}$$ □

Definición 4.2.4 (Función subfórmula) $\mathrm{Subf}\colon Cl \to \wp(Cl)$ *es la función* ***subfórmula****, que devuelve el conjunto de subfórmulas de cada fórmula:*

$$\begin{aligned}\mathrm{Subf}(A) &= \{A\} \quad \textit{para todo } A \in Q\\ \mathrm{Subf}(\neg B) &= \{\neg B\} \cup \mathrm{Subf}(B)\\ \mathrm{Subf}(C * D) &= \{C * D\} \cup \mathrm{Subf}(C) \cup \mathrm{Subf}(D), \quad \textit{si } * \in \{\vee, \wedge, \to, \leftrightarrow\}\end{aligned}$$

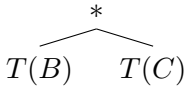
Si $F \in \mathrm{Subf}(A)$*, decimos que* F *es* ***subfórmula*** *de* A *y escribimos* $F \sqsubseteq A$*. Si* $F \in \mathrm{Subf}(A)$ *y* $F \neq A$*, decimos que* F *es* ***subfórmula propia*** *de* A *y escribimos* $F \sqsubset A$ □

Ejemplo 4.2.5

$$\begin{aligned}\mathrm{Subf}(p \to (r \vee \neg q)) &= \{p \to (r \vee \neg q)\} \cup \mathrm{Subf}(p) \cup \mathrm{Subf}(r \vee \neg q)\\ &= \{p \to (r \vee \neg q)\} \cup \{p\} \cup \{r \vee \neg q\} \cup \mathrm{Subf}(r) \cup \mathrm{Subf}(\neg q)\\ &= \{p \to (r \vee \neg q), p, r \vee \neg q, r, \neg q, q\}\end{aligned}$$

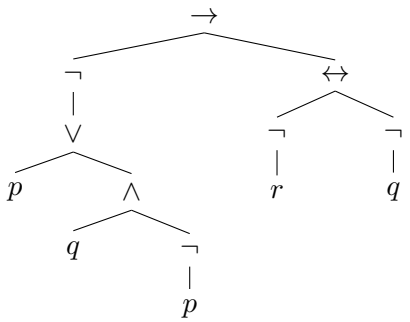
Por lo tanto, podemos escribir $r \vee \neg q \sqsubset p \to (r \vee \neg q)$ y podemos decir que $\neg q$ es subfórmula propia de $p \to (r \vee \neg q)$. □

Definición 4.2.6 (Árbol sintáctico) *El* ***árbol sintáctico*** *de las fórmulas de* Cl *se define como:*

- *Si* $A \in Q$*,* $T(A)$ *es el árbol hoja etiquetado con* A*.*
- *Si* $* \in \{\vee, \wedge, \to, \leftrightarrow\}$ *y* $A = B * C$*,* $T(A) =$ 

- *Si* $A = \neg B$*,* $T(A) =$ 

Ejemplo 4.2.7 El árbol $T(A)$ para la fórmula $A = \neg(p \vee (q \wedge \neg p)) \to (\neg r \leftrightarrow \neg q)$ es



□

4.3. La Lógica Clásica Proposicional $\mathscr{C}\ell$

Recordemos que en la Lógica Clásica Proposicional solo consideramos construcciones declarativas, sobre las que podemos pronunciarnos acerca de su **verdad** o **falsedad**; por esa razón, vamos a considerar solo dos valores semánticos. Por otra parte, la validez de las construcciones compuestas debe determinarse funcionalmente a partir de la validez de las afirmaciones elementales, por eso, en el lenguaje, hemos dejado fuera aspectos dependientes del contexto, como el tiempo, modalidades, creencias,...

Definición 4.3.1 *La **Lógica Clásica Proposicional** se define sobre el lenguaje Cl con la siguiente teoría de modelos: $\mathscr{C}\ell = (\{0,1\}, \{1\}, \mathcal{I})$, en donde el conjunto de interpretaciones $\mathcal{I}$ está formado por todas las funciones* $\mathrm{I}\colon Cl \to \{0,1\}$ *que verifican:*

- $\mathrm{I}(\neg A) = 1$ *si, y sólo si,* $\mathrm{I}(A) = 0$
- $\mathrm{I}(A \wedge B) = 1$ *si, y sólo si,* $\mathrm{I}(A) = \mathrm{I}(B) = 1$
- $\mathrm{I}(A \vee B) = 0$ *si, y sólo si,* $\mathrm{I}(A) = \mathrm{I}(B) = 0$
- $\mathrm{I}(A \to B) = 0$ *si, y sólo si,* $\mathrm{I}(A) = 1$ *e* $\mathrm{I}(B) = 0$
- $\mathrm{I}(A \leftrightarrow B) = 1$ *si, y sólo si,* $\mathrm{I}(A) = \mathrm{I}(B)$

Teniendo en cuenta la definición recursiva, cada interpretación queda determinada por una función $\mathrm{I}\colon Q \to \{0,1\}$. Aunque Q contiene todos las posibles variables proposicionales, en cada problema o aplicación, solo necesitaremos definir las interpretaciones sobre las variables proposicionales que intervengan en él. Si en una fórmula (o un conjunto de fórmulas) intervienen n variables proposicionales, el número de posibles interpretaciones es 2^n.

Ejemplo 4.3.2 Si consideramos dos variables proposicionales, tendremos cuatro posibles interpretaciones. En las siguientes tabla mostramos las evaluaciones de las

interpretaciones de varias fórmulas usando un representación en forma de **tablas de verdad**:

	p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \to q$	$p \leftrightarrow q$
I_1	1	1	0	1	1	1	1
I_2	1	0	0	0	1	0	0
I_3	0	1	1	0	1	1	0
I_4	0	0	1	0	0	1	1

	p	q	$\neg p$	$\neg p \vee q$	$\neg p \wedge q$	$(\neg p \vee q) \to (\neg p \wedge q)$
I_1	1	1	0	1	0	0
I_2	1	0	0	0	0	1
I_3	0	1	1	1	1	1
I_4	0	0	1	1	0	0

□

La siguiente proposición muestra más claramente que las interpretaciones son funciones en el conjunto de valores semánticos y nos da, además, unos métodos operacionales alternativos para evaluar las interpretaciones.

PROPOSICIÓN 4.3.3 *Si* $\mathrm{I}\colon \mathrm{Cl} \to \{0,1\}$ *es una interpretación de la Lógica Clásica Proposicional:*

1. $\mathrm{I}(\neg A) = 1 - \mathrm{I}(A)$
2. $\mathrm{I}(A \wedge B) = \mathrm{mín}\{\mathrm{I}(A), \mathrm{I}(B)\}$
3. $\mathrm{I}(A \wedge B) = \mathrm{I}(A) \cdot \mathrm{I}(B)$
4. $\mathrm{I}(A \vee B) = \mathrm{máx}\{\mathrm{I}(A), \mathrm{I}(B)\}$
5. $\mathrm{I}(A \vee B) = \mathrm{I}(A) + \mathrm{I}(B) - \mathrm{I}(A) \cdot \mathrm{I}(B)$
6. $\mathrm{I}(A \to B) = 1 - \mathrm{I}(A) + \mathrm{I}(A) \cdot \mathrm{I}(B)$
7. $\mathrm{I}(A \to B) = 1$ *si, y sólo si,* $\mathrm{I}(A) \leq \mathrm{I}(B)$
8. $\mathrm{I}(A \leftrightarrow B) = 1 - |\mathrm{I}(A) - \mathrm{I}(B)|$

La siguiente definición es la misma que hicimos para una teoría de modelos general, pero la incluimos aquí para facilitar el estudio y la comprensión de la lógica.

DEFINICIÓN 4.3.4

- *Una interpretación* $\mathrm{I} \in \mathcal{I}$ *es un* ***modelo*** *de una fórmula* A *si* $\mathrm{I}(A) = 1$*; decimos también que la interpretación* I ***satisface*** *la fórmula* A*. Denotaremos por* $\mathtt{Mod}(A)$ *al conjunto de los modelos de* A*.*

- *Decimos que* $\mathrm{I} \in \mathcal{I}$ *es un* ***contramodelo*** *de una fórmula* A *si* $\mathrm{I}(A) = 0$
- $I \in \mathcal{I}$ *es un* ***modelo*** *de un conjunto de fórmulas* Ω*, si* $I(A) = 1$ *para todo* $A \in \Omega$*. Denotaremos por* $\texttt{Mod}(\Omega)$ *al conjunto de los modelos de* Ω*.*
- *Decimos que una fórmula* A *es* ***satisfacible*** *o* ***consistente*** *si admite algún modelo, es decir, si* $\texttt{Mod}(A) \neq \varnothing$
- *Decimos que un conjunto de fórmulas* Ω *es* ***satisfacible*** *o* ***consistente*** *si admite un modelo, es decir, si* $\texttt{Mod}(\Omega) \neq \varnothing$*.*
- *Decimos que una fórmula* A *es* ***insatisfacible*** *o* ***inconsistente*** *si no admite ningún modelo, es decir, si* $\texttt{Mod}(A) = \varnothing$*.*
- *Decimos que una fórmula* A *es* ***válida*** *o que es una* ***tautología*** *si cada interpretación es modelo de* A*, es decir, si* $\texttt{Mod}(A) = \mathcal{I}$*.*

Ejemplo 4.3.5 En la siguiente tabla, evaluamos todas las interpretaciones de la fórmula $A = (\neg p \vee q) \to (\neg p \wedge q)$.

	p	q	$A = (\neg p \vee q) \to (\neg p \wedge q)$
I_1	1	1	0
I_2	1	0	1
I_3	0	1	1
I_4	0	0	0

$\mathrm{I}_2(A) = 1$,
I_2 es modelo de A,
A es satisfacible.

$\mathrm{I}_4(A) = 0$,
I_4 es contramodelo de A,
A no es válida.

Por lo tanto, $\texttt{Mod}(A) = \{\mathrm{I}_2, \mathrm{I}_3\}$, y podemos afirmar que A es satisfacible pero no es válida. □

Ejemplo 4.3.6 En la siguiente tabla mostramos todas las interpretaciones de la fórmula $A = (p \wedge (p \to q)) \to q$:

	p	q	$A = (p \wedge (p \to q)) \to q$
I_1	1	1	1
I_2	1	0	1
I_3	0	1	1
I_4	0	0	1

Por lo tanto, $\texttt{Mod}(A) = \{\mathrm{I}_1, \mathrm{I}_2, \mathrm{I}_3, \mathrm{I}_4\} = \mathcal{I}$, y podemos afirmar que A es una tautología. □

Las propiedades de satisfacibilidad de fórmulas y de conjuntos de fórmulas están relacionadas por el siguiente resultado.

TEOREMA 4.3.7 *$\Omega = \{A_1, \ldots, A_n\}$ es satisfacible si, y sólo si, $A_1 \wedge \cdots \wedge A_n$ es satisfacible.*

El operador `Mod` caracteriza la teoría de modelos y justifica precisamente esa denominación. Hemos visto que las nociones básicas de una teoría de modelos quedan determinadas por este operador, y a continuación vemos un resultado que establece cómo determinar de forma recursiva el conjunto de modelos de una fórmula o un conjunto de fórmulas.

TEOREMA 4.3.8 *Sean A y B fórmulas, entonces:*

- $\texttt{Mod}(\neg A) = \overline{\texttt{Mod}(A)} = \mathcal{I} - \texttt{Mod}(A)$
- $\texttt{Mod}(A \wedge B) = \texttt{Mod}(A) \cap \texttt{Mod}(B)$
- $\texttt{Mod}(A \vee B) = \texttt{Mod}(A) \cup \texttt{Mod}(B)$
- $\texttt{Mod}(A \to B) = \overline{\texttt{Mod}(A)} \cup \texttt{Mod}(B)$
- $\texttt{Mod}(A \leftrightarrow B) = \overline{\texttt{Mod}(A) \mathbin{\triangle} \texttt{Mod}(B)}$
- $\texttt{Mod}(\Omega_1 \cup \Omega_2) = \texttt{Mod}(\Omega_1) \cap \texttt{Mod}(\Omega_2)$

DEFINICIÓN 4.3.9 (INFERENCIA SEMÁNTICA) *Si $\Omega = \{A_1, A_2, \ldots, A_n\}$ es un conjunto de fórmulas y A es otra fórmula, decimos que A* ***es consecuencia, se deriva semánticamente*** *o* ***se infiere*** *de Ω, si todo modelo de Ω es modelo de A, es decir, si $\texttt{Mod}(\Omega) \subseteq \texttt{Mod}(A)$; en tal caso, escribimos $\Omega \models A$, o $A_1, A_2, \ldots, A_n \models A$.*

En particular, si $\Omega = \varnothing$, entonces $\Omega \models A$ si, y sólo si, A es una tautología; por esta razón, escribimos $\models A$ para expresar que A es una fórmula válida. Además, en general, las nociones de validez de una fórmula y de corrección de una inferencia están relacionas según establece el siguiente resultado.

TEOREMA 4.3.10 *$A_1, A_2, \ldots, A_n \models A$ si, y sólo si, $\big(A_1 \wedge A_2 \wedge \cdots \wedge A_n\big) \to A$ es una tautología.*

EJEMPLO 4.3.11 Para $\Omega = \{\neg q \to p,\ \neg q \vee r\}$, y $A = \neg p \to r$, vamos a comprobar que $\Omega \models A$.

	p	q	r	$\neg p$	$\neg q$	$\neg q \to p$	$\neg q \vee r$	$\neg p \to r$
I_1	1	1	1	0	0	1	1	1
I_2	1	0	1	0	1	1	1	1
I_3	0	1	1	1	0	1	1	1
I_4	0	0	1	1	1	0	1	1
I_5	1	1	0	0	0	1	0	1
I_6	1	0	0	0	1	1	1	1
I_7	0	1	0	1	0	1	0	0
I_8	0	0	0	1	1	0	1	0

Entonces, $\mathtt{Mod}(\Omega) = \{\mathrm{I}_1, \mathrm{I}_2, \mathrm{I}_3, \mathrm{I}_6\}$ y $\mathtt{Mod}(A) = \{\mathrm{I}_1, \mathrm{I}_2, \mathrm{I}_3, \mathrm{I}_4, \mathrm{I}_5, \mathrm{I}_6\}$; por lo tanto

$$\mathtt{Mod}(\Omega) \subset \mathtt{Mod}(A) \qquad \text{y} \qquad \Omega \models A$$ □

En una teoría de modelos, el conjunto de modelos de una fórmula determina el significado de la fórmula. De esta forma, si dos fórmulas tienen los mismos modelos, son indistiguibles en la teoría.

Definición 4.3.12 *Decimos que dos fórmulas A y B son **(lógicamente) equivalentes** si $\mathtt{Mod}(A) = \mathtt{Mod}(B)$; en tal caso, escribimos $A \equiv B$.*

Ejemplo 4.3.13 Las fórmulas $\neg p \vee q$ y $p \to q$ son equivalentes, según se deduce de las siguiente tabla:

	p	q	$\neg p$	$\neg p \vee q$	$p \to q$
I_1	1	1	0	1	1
I_2	1	0	0	0	0
I_3	0	1	1	1	1
I_4	0	0	1	1	1

Por lo tanto $\mathtt{Mod}(\neg p \vee q) = \{\mathrm{I}_1, \mathrm{I}_3, \mathrm{I}_4\} = \mathtt{Mod}(p \to q)$ y podemos concluir que $\neg p \vee q \equiv p \to q$. □

Teorema 4.3.14 *$A \equiv B$ si, y sólo si, $A \leftrightarrow B$ es una tautología.*

Ejemplo 4.3.15 Podemos demostrar una equivalencia (o la corrección de una inferencia, o la validez de una fórmula) utilizando las propiedades del operador `Mod` y las propiedades de las operaciones entre conjuntos, en lugar de evaluar las interpretaciones de las fórmulas. Vamos a demostrar de esta forma la equivalencia $\neg(A \wedge B) \equiv \neg A \vee \neg B$:

$$\begin{aligned}
\mathtt{Mod}(\neg(A \wedge B)) &= \overline{\mathtt{Mod}(A \wedge B)} \\
&= \overline{\mathtt{Mod}(A) \cap \mathtt{Mod}(B)} \\
&= \overline{\mathtt{Mod}(A)} \cup \overline{\mathtt{Mod}(B)} \\
&= \mathtt{Mod}(\neg A) \cup \mathtt{Mod}(\neg B) \\
&= \mathtt{Mod}(\neg A \vee \neg B)
\end{aligned}$$ □

TEOREMA 4.3.16 *En la Lógica Clásica Proposicional se verifican las siguientes equivalencias*

Leyes conmutativas:	$A \wedge B \equiv B \wedge A$
	$A \vee B \equiv B \vee A$
Leyes asociativas:	$(A \wedge B) \wedge C \equiv A \wedge (B \wedge C)$
	$(A \vee B) \vee C \equiv A \vee (B \vee C)$
Leyes de Absorción:	$A \wedge (A \vee B) \equiv A$
	$A \vee (A \wedge B) \equiv A$
Leyes de Idempotencia:	$A \equiv A \wedge A$
	$A \equiv A \vee A$
Ley Distributiva de $\wedge$ respecto $\vee$:	$A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$
Ley Distributiva de $\vee$ respecto $\wedge$:	$A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
Leyes de De Morgan:	$\neg(A \wedge B) \equiv \neg A \vee \neg B$
	$\neg(A \vee B) \equiv \neg A \wedge \neg B$
Ley de doble negación:	$\neg\neg A \equiv A$
Ley de Transposición:	$A \to B \equiv \neg B \to \neg A$
Interdefinición *de* $\to$ *y* $\vee$:	$A \to B \equiv \neg A \vee B$
	$A \vee B \equiv \neg A \to B$
Interdefinición *de* $\to$ *y* $\wedge$:	$\neg(A \to B) \equiv A \wedge \neg B$
	$A \wedge B \equiv \neg(A \to \neg B)$
Conmutatividad *de* $\leftrightarrow$:	$A \leftrightarrow B \equiv B \leftrightarrow A$
Asociatividad *de* $\leftrightarrow$:	$A \leftrightarrow (B \leftrightarrow C) \equiv (A \leftrightarrow B) \leftrightarrow C$

Hemos demostrado antes $\neg(p \wedge q) \equiv \neg p \vee \neg q$, pero ¿podemos deducir que entonces $r \to \neg(p \wedge q) \equiv r \to (\neg p \vee \neg q)$? La respuesta es sí: si en una fórmula sustituimos una subfórmula por otra fórmula equivalente a ella, el resultado es una fórmula equivalente a la fórmula inicial. Para establecer formalmente este resultado, necesitamos definir previamente la operación de **sustitución**.

DEFINICIÓN 4.3.17 *Si A, B y C son fórmulas tales que $B \sqsubseteq A$, denotamos por $A[B/C]$ a la fórmula que se obtiene al sustituir en A cada aparición de la subfórmula B por la fórmula C.*

TEOREMA 4.3.18 (PRINCIPIO DE SUSTITUCIÓN) *Si A y C son fórmulas, p es una variable proposicional en A, y A es válida, entonces $A[p/C]$ también es válida.*

TEOREMA 4.3.19 (DE EQUIVALENCIA) *Si $B \sqsubseteq A$ y $B \equiv C$, entonces $A \equiv A[B/C]$.*

4.3.1. Expresividad

Una vez definida una semántica sobre un lenguaje formal, podemos volver a la definición del lenguaje e introducir simplificaciones justificadas por la semántica. Por ejemplo, las leyes asociativas permiten considerar como fórmulas bien formadas

a las expresiones

$$A_1 \vee A_2 \vee \cdots \vee A_n = \bigvee_{i=1}^{n} A_i \qquad A_1 \wedge A_2 \wedge \cdots \wedge A_n = \bigwedge_{i=1}^{n} A_i$$

También podemos reducir el número de conectivos, definiendo unos a partir de otros. Por ejemplo, sería suficiente trabajar solo con la negación y la implicación y definir el resto a partir de ellos

$$\begin{aligned} A \vee B &=_{def} \neg A \to B \\ A \wedge B &=_{def} \neg(A \to \neg B) \\ A \leftrightarrow B &=_{def} \neg((A \to B) \to \neg(B \to A)) \end{aligned}$$

En este caso, decimos que $\{\neg, \to\}$ es un **conjunto adecuado** de conectivos.

La reducción del número conectivos puede ser útil en determinadas áreas, como en el diseño de circuitos digitales, pero si queremos representar información o conocimiento, es preferible aumentar el conjunto de conectivos, aumentando la **expresividad** del lenguaje. Por ejemplo, en la Lógica Clásica podemos utilizar más conectivos, además de los que hemos considerado incialmente en el lenguaje: el operador **nor**, el operador **nand**, la **disyunción exclusiva** y el operador **si**:

p	q	$p \downarrow q$	$p \mid q$	$p \oplus q$	$p \leftarrow q$
1	1	0	0	0	1
1	0	0	1	1	1
0	1	0	1	1	0
0	0	1	1	0	1

4.3.2. Automatización de las demostraciones

Se dice que una lógica es **decidible** si es posible diseñar un algoritmo que determine si cualquier inferencia es válida o no lo es: la Lógica Clásica Proposicional es decidible; por ejemplo, la evaluación de las interpretaciones constituyen un algoritmo de demostración. No todas las lógicas son decidibles, lo que no implica que no sean útiles computacionalmente. Por ejemplo, la lógica Clásica de Predicados es **semidecidible**, es decir, únicamente disponemos de procedimientos cuya finalización está garantizada solamente si la fórmula de entrada es válida.

La mayoría de los algoritmos de deducción automática, y en particular el método de las **tablas semánticas**, que aprenderemos más adelante, hacen uso del **principio de refutación**.

Teorema 4.3.20 (Principio de refutación) *Sea Ω un conjunto de fórmulas y sea A otra fórmula. Entonces:*

$$\Omega \models A \text{ si, y sólo si, } \Omega \cup \{\neg A\} \text{ es insatisfacible.}$$

La demostración es sencilla a partir de las propiedades del operado Mod:

$$\begin{aligned}
\Omega \models A &\iff \mathtt{Mod}(\Omega) \subseteq \mathtt{Mod}(A) \\
&\iff \mathtt{Mod}(\Omega) \cap \overline{\mathtt{Mod}(A)} = \varnothing \\
&\iff \mathtt{Mod}(\Omega) \cap \mathtt{Mod}(\neg A) = \varnothing \\
&\iff \mathtt{Mod}(\Omega \cup \{\neg A\}) = \varnothing \\
&\iff \Omega \cup \{\neg A\} \quad \text{es insatisfacible.}
\end{aligned}$$

La ventaja de utilizar el principio de refutación es que convierte el problema de generar y evaluar interpretaciones en un problema de búsqueda: ¿es posible encontrar una interpretación tal que $\mathrm{I}(A_1 \wedge \cdots \wedge A_n \wedge \neg A) = 1$? Si no es posible, la inferencia $A_1, \ldots, A_n \models A$ es correcta, y en caso contrario, no lo es.

Ejemplo 4.3.21 La inferencia $(p \wedge q) \to r, p \to q \models p \to r$ es correcta, ya que no es posible encontrar un modelo de $(p \wedge q) \to r$, $p \to q$ y $\neg(p \to r)$.

Supongamos que $\mathrm{I}(\neg(p \to r)) = 1$, entonces necesariamente $\mathrm{I}(p) = 1$ e $\mathrm{I}(r) = 0$. Si además, $\mathrm{I}(p \to q) = 1$, entonces necesariamente $\mathrm{I}(q) = 1$. Por lo tanto, $\mathrm{I}((p \wedge q) \to r) = 0$ y podemos concluir que $\{(p \wedge q) \to r, p \to q, \neg(p \to r)\}$ es insatisfacible y que la inferencia $(p \wedge q) \to r, p \to q \models p \to r$ es correcta. □

4.3.3. El método de las Tablas Semánticas

El método de las tablas semánticas es un algoritmo para el estudio de la satisfacibilidad de conjuntos de fórmulas. Gracias al principio de refutación, este algoritmo puede ser usado para estudiar la validez de fórmulas y la corrección de inferencias.

Antes de presentar el método, necesitamos introducir algunos conceptos. Un **literal** es una fórmula atómica o la negación de una fórmula atómica; los literales p y $\neg p$ son **literales opuestos**.

Las fórmulas que responden a los esquemas $A \wedge B$, $\neg(A \to B)$, $\neg(A \vee B)$ y $\neg\neg A$ se denominan fórmulas de tipo α o de comportamiento conjuntivo. En la búsqueda de modelos de una fórmula de tipo α basta determinar los módelos de dos fórmula obtenidas a partir de sus subfórmulas:

$$\begin{aligned}
\mathtt{Mod}(A \wedge B) &= \mathtt{Mod}(\{A, B\}) \\
\mathtt{Mod}(\neg(A \to B)) &= \mathtt{Mod}(\{A, \neg B\}) \\
\mathtt{Mod}(\neg(A \vee B)) &= \mathtt{Mod}(\{\neg A, \neg B\}) \\
\mathtt{Mod}(\neg\neg A) &= \mathtt{Mod}(A)
\end{aligned}$$

Las fórmulas que responden a los esquemas $A \vee B$, $A \to B$ y $\neg(A \wedge B)$ se denominan fórmulas de tipo β o de comportamiento disyuntivo. En la búsqueda de

modelos de una fórmula de tipo β tenemos que considerar, de forma independiente, los modelos de dos fórmula obtenidas a partir de sus subfórmulas:

$$\begin{aligned}\texttt{Mod}(A \vee B) &= \texttt{Mod}(A) \cup \texttt{Mod}(B)\\ \texttt{Mod}(A \to B) &= \texttt{Mod}(\neg A) \cup \texttt{Mod}(B)\\ \texttt{Mod}(\neg(A \wedge B)) &= \texttt{Mod}(\neg A) \cup \texttt{Mod}(\neg B)\end{aligned}$$

Con las propiedades anteriores, podemos construir un árbol de búsqueda de modelos, estos árboles se denominan **tablas semánticas**, y en ellas se transmite el operador `Mod` hasta llegar a los literales.

DEFINICIÓN 4.3.22 *Consideremos las fórmulas $B_1, \ldots, B_m$*

- *El árbol con una única rama*

  $$\begin{array}{c} B_1 \\ B_2 \\ \vdots \\ B_m \end{array}$$

  *es una **tabla semántica**, denominada **tabla inicial**, para $\{B_1, \ldots, B_m\}$*

- *Si T es una **tabla semántica** y T' se obtiene a partir de T aplicando una regla extensión, entonces T' es una **tabla semántica**.*

  ***Extension α:** Si una fórmula de tipo α aparece en la tabla, entonces cada rama 'que contiene' a esa fórmula es extendida con uno o dos nodos según los siguientes esquemas.*

  $$\begin{array}{c} A \wedge B \ \checkmark \\ X_1 \\ \vdots \\ X_n \\ A \\ B \end{array} \qquad \begin{array}{c} \neg(A \to B) \ \checkmark \\ X_1 \\ \vdots \\ X_n \\ A \\ \neg B \end{array} \qquad \begin{array}{c} \neg(A \vee B) \ \checkmark \\ X_1 \\ \vdots \\ X_n \\ \neg A \\ \neg B \end{array} \qquad \begin{array}{c} \neg\neg A \ \checkmark \\ X_1 \\ \vdots \\ X_n \\ A \end{array}$$

- ***Extensión β:** Si una fórmula de tipo β aparece en la tabla, entonces cada rama 'que contiene' a la fórmula es extendida con dos nodos, uno como descendiente izquierdo y otro descendiente derecho según los siguientes esquemas.*

  $$\begin{array}{c} A \vee B \ \checkmark \\ X_1 \\ \vdots \\ X_n \\ \diagup \quad \diagdown \\ A \qquad B \end{array} \qquad \begin{array}{c} A \to B \ \checkmark \\ X_1 \\ \vdots \\ X_n \\ \diagup \quad \diagdown \\ \neg A \qquad B \end{array} \qquad \begin{array}{c} \neg(A \wedge B) \ \checkmark \\ X_1 \\ \vdots \\ X_n \\ \diagup \quad \diagdown \\ \neg A \qquad \neg B \end{array}$$

La marca ✔ que colocamos en las fórmulas indica que esa fórmula ya ha sido usada para extender la tabla y que no es necesario utilizarla de nuevo.

Una rama (el camino que une una hoja con la raíz en recorrido ascencente) en una tabla semántica se dice **(atómicamente) cerrada** si en ella aparece una fórmula atómica y su negación; estas ramas se marcan con el símbolo ✘ por debajo de su hoja. Una rama se dice **abierta** si no es cerrada. Una tabla se dice **cerrada** si todas sus ramas son cerradas. Una rama se dice **completa** si todas sus fórmulas han sido expandidas (es decir, todas las fórmulas de tipo α y β aparecen con la marca ✔). Una tabla se dice **terminada** si todas sus ramas son cerradas o completas.

Teorema 4.3.23 (Corrección y completitud de las Tablas Semánticas)

1. *Un conjunto de fórmulas $\{B_1, \ldots, B_m\}$ es insatisfacible si, y sólo si, es posible construir una tabla cerrada para $\{B_1, \ldots, B_m\}$.*
2. *Si Γ es el conjunto de literales de una rama completa y abierta de una tabla para $\{B_1, \ldots, B_m\}$, entonces cualquier modelo de Γ es un modelo de $\{B_1, \ldots, B_m\}$.*

Corolario 4.3.24

1. *La inferencia $A_1, \ldots, A_n \models A$ es válida si, y sólo si, existe una tabla cerrada para $\{A_1, \ldots, A_n, \neg A\}$.*
2. *Si Γ es el conjunto de literales que aparecen en una rama completa y abierta de un tabla para $\{A_1, \ldots, A_n, \neg A\}$, entonces cualquier modelo de Γ es un contramodelo de $A_1, \ldots, A_n \models A$.*

En la figura 4.1 aparece el diagrama del algoritmo para generar las tablas semánticas a partir de un conjunto de fórmulas, hasta llegar a una tabla terminada.

Dado que solo estamos interesados en la satisfacibilidad, el algoritmo se detiene tan pronto se encuentre una rama completa y abierta. Sin embargo, si seguimos extendiendo la tabla hasta llegar a una en la que todas las ramas están cerradas o completas, entonces podemos determinar todos los modelos del conjunto de fórmulas inicial a partir de las ramas abiertas.

Aunque en términos de corrección, el orden de elección de las fórmulas es indiferente, debemos tener en cuenta que para mejorar la eficiencia debemos darle prioridad a las fórmulas α sobre las fórmulas β. La complejidad del método reside en la necesidad de verificar si las ramas son cerradas o no, y por lo tanto, cuantas más ramas tenga la tabla, más verificaciones tendremos que hacer. Las extensiones α no aumentan el número de ramas, y por eso les damos prioridad sobre la β.

Debemos tener siempre en cuenta que estos algoritmos describen manipulaciones puramente sintácticas y, por lo tanto, deben realizarse tal y como aparecen en las reglas. En particular, las fórmulas que se añaden a la tabla en cada extensión deben construirse a partir de las subfórmulas sin aplicar ninguna equivalencia.

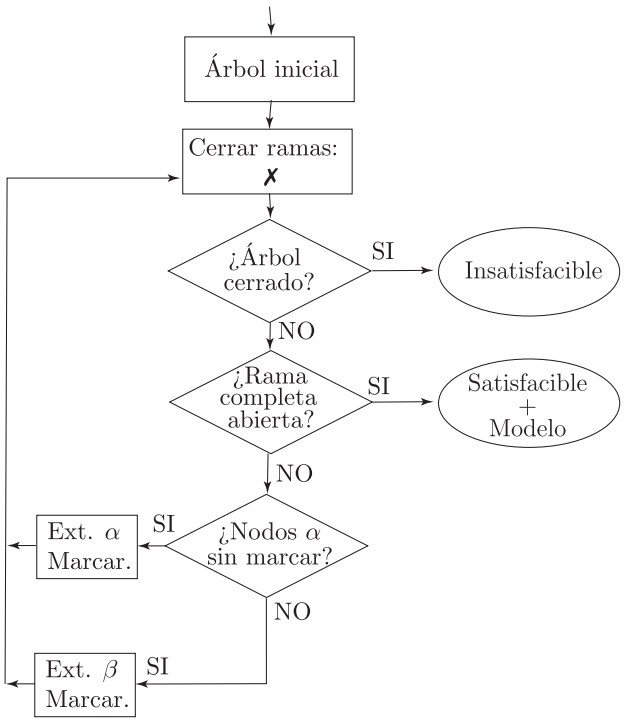
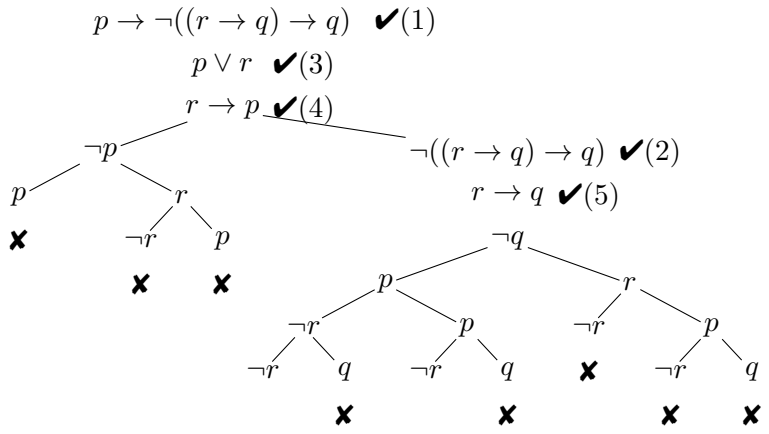


Figura 4.1: Diagrama del algoritmo de las Tablas Semánticas

Ejemplo 4.3.25 Para estudiar la satisfacibilidad del conjunto

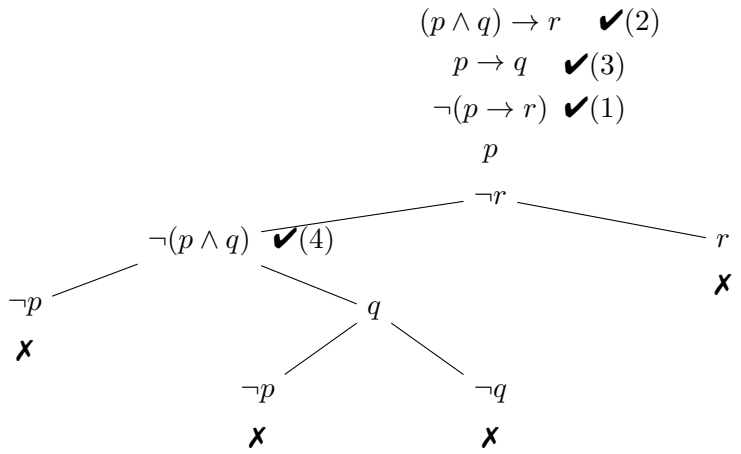
$$\Omega = \{p \to \neg((r \to q) \to q), p \vee r, r \to p\}$$

construimos la siguiente tabla semántica.



Esta tabla es completa y contiene ramas abiertas. Por lo tanto, el conjunto Ω es satisfacible. La rama abierta de más a la izquierda contiene a los literales $\Gamma = \{\neg q, p, \neg r\}$, y por lo tanto, un modelo de Γ y de Ω es es $\mathrm{I}(p) = 1$, $\mathrm{I}(q) = \mathrm{I}(r) = 0$. □

Ejemplo 4.3.26 La corrección de la inferencia $(p \wedge q) \to r,\ p \to q \models p \to r$ es equivalente a la insatisfacibilidad del conjunto $\Omega = \{(p \wedge q) \to r,\ p \to q, \neg(p \to r)\}$ y esta propiedad la estudiamos con la siguiente tabla semántica



Dado que la tabla es cerrada, podemos afirmar que el conjunto Ω es insatisfacible y, en consecuencia, la inferencia inicial es correcta. □

Relación de ejercicios 9

1. De las siguientes cadenas de símbolos, diga cuáles son fórmulas bien formadas de la Lógica Clásica Proposicional, cuáles no y diga por qué:

   a) $p \wedge q \to \neg r$; *b*) $(p \wedge \neg r) \to q$; *c*) $\neg(\neg(p \vee q))$;

   d) $(q \vee r) \to$; *e*) $(\neg p \wedge r) \to \neg(p \to \neg r)$; *f*) $r \leftarrow (\neg p \vee q)$.

2. Determine los modelos y contramodelos de la fórmula $A = (p \vee \neg q) \to (p \wedge q)$:
   ¿Es satisfacible la fórmula A? ¿Es válida la fórmula A?

3. Determine los modelos y contramodelos de la fórmula

   $$B = (p \to \neg q) \to (\neg p \vee \neg q)$$

   ¿Es satisfacible la fórmula B? ¿Es válida la fórmula B?

4. Determine si son insatisfacibles, satisfacibles o válidas las siguientes fórmulas:

   a) $(p \to q) \to p$; *b*) $(p \vee q) \to \neg(q \vee p)$; *c*) $(p \wedge q) \to (q \wedge p)$

5. Determine si son satisfacibles o insatisfacibles los siguientes conjuntos de fórmulas:

   a) $\{p, q, p \vee q\}$; *b*) $\{p, \neg q, p \wedge q\}$;

   c) $\{p \vee q, \neg(\neg p \to q)\}$; *d*) $\{p \to q, (p \wedge q) \to \neg p\}$

6. Construya, si es posible: (a) una fórmula *bien formada* que NO sea *válida*; (b) una fórmula *bien formada* que SÍ sea *válida*; (c) una fórmula *válida* que NO sea *bien formada*.

7. Construya, si es posible: (a) una fórmula *satisfacible* que NO sea *válida*; (b) una fórmula *satisfacible* que SÍ sea *válida*; (c) una fórmula *válida* que NO sea *satisfacible*.

8. Razone con exactitud sobre la veracidad de las siguientes afirmaciones:

   a) Si una fórmula no es válida, su negación sí lo es.

   b) Si una fórmula no es satisfacible, su negación sí lo es.

   c) Si una fórmula no es consecuencia de un conjunto de fórmulas, su negación sí lo es.

   d) Si una fórmula no es consecuencia de un conjunto de fórmulas, su negación tampoco.

e) Si un conjunto de fórmulas es satisfacible, cada elemento del conjunto también es satisfacible.

f) Si cada elemento de un conjunto de fórmulas es satisfacible, el conjunto también es satisfacible.

9. Formalice los siguientes razonamientos:

   a) *Si no hay control de nacimientos, entonces la población crece ilimitadamente. Pero si la población crece ilimitadamente, aumentará el índice de pobreza. Por consiguiente, si no hay control de nacimientos, aumentará el indice de pobreza.*

   b) *Si Valdés ha instalado calefacción central, entonces ha vendido su coche o ha pedido dinero prestado al banco. Por tanto, si Valdés no ha vendido su coche, entonces no ha instalado calefacción central.*

10. Escriba cinco fórmulas con tres variables proposicionales distintas y grado mayor que cinco. Diga cuáles son insatisfacibles, cuáles son satisfacibles y cuáles son válidas.

11. Demuestra la siguiente propiedad: Si $\Omega \supseteq \Omega'$, entonces $\texttt{Mod}(\Omega) \subseteq \texttt{Mod}(\Omega')$

Relación de ejercicios 10

1. Determine los modelos y contramodelos del conjunto de fórmulas

$$\Omega = \{\neg q \to p, \neg p \vee r, \neg q \to \neg r\}$$

¿El conjunto Ω es satisfacible? ¿Es correcta la inferencia $\Omega \models q$? ¿Es correcta la inferencia $\Omega \models r$?

2. Estudie la validez del siguiente razonamiento:

> *Si hay petróleo en Poligonia, entonces o los expertos tienen razón o el gobierno está mintiendo. No hay petroleo en Poligonia, o si no los expertos se equivocan. Así pues, el gobierno no está mintiendo.*

3. Consideremos las siguientes fórmulas: $A = p \to (q \to r)$, $B = (p \wedge q) \to r$.

   a) Determine los conjuntos $\texttt{Mod}(A)$ y $\texttt{Mod}(B)$.

   b) ¿Qué relación hay entre A y B?

4. Razone con exactitud sobre la veracidad de las siguientes afirmaciones:

   a) Si $\Omega \models A$, es posible que exista $\Omega' \supset \Omega$ tal que $\Omega' \not\models A$.

   b) Si $\Omega \not\models A$, es posible que exista $\Omega' \supset \Omega$ tal que $\Omega' \models A$.

5. Demuestre la validez de las siguientes fórmulas utilizando Tablas semánticas.

   a) $(p \to (q \to r)) \to ((p \to q) \to (p \to r))$;
   b) $(\neg p \to \neg q) \to (q \to p)$;
   c) $((p \vee q) \wedge (p \to r) \wedge (q \to r)) \to r$;
   d) $(p \to (q \wedge \neg q)) \to \neg p$;

6. Use Tablas Semánticas para estudiar la validez de las siguientes fórmulas

$$A = (\neg r \vee (p \wedge q)) \to ((r \to p) \wedge (r \to q))$$
$$B = (p \wedge r) \to ((q \to s) \to ((p \vee q) \to s))$$

7. Estudie la validez de las siguientes inferencias utilizando Tablas semánticas.

   a) $p \to (q \vee r), q \to r, r \to s \models p \to s$
   b) $p \to (q \vee r), q \to r, r \to s \models p \to \neg s$
   c) $p \to q, r \to s, (s \wedge q) \to t \models (p \wedge r) \to t$
   d) $p \to (q \to r), p \to q, p \models r$

8. Use Tablas Semánticas para estudiar la satisfacibilidad del conjunto

$$\Omega = \{p \to (q \to r), p \to q, p\}$$

y, en caso afirmativo, determine todos sus modelos.

9. Las fórmulas del tipo $A \leftrightarrow B$ y $\neg(A \leftrightarrow B)$ pueden considerarse como fórmulas de tipo β en el método de las Tablas semánticas usando las siguientes equivalencias $A \leftrightarrow B \equiv (A \wedge B) \vee (\neg A \wedge \neg B)$ y $\neg(A \leftrightarrow B) \equiv (\neg A \wedge B) \vee (A \wedge \neg B)$ y, en consecuencia, las siguientes reglas de expansión:

Utilizando estas reglas, estudie la validez de las siguientes fórmulas

a) $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$

b) $(p \rightarrow q) \leftrightarrow (p \leftrightarrow (p \wedge q))$