Apuntes del Curso
Matemática Discreta para la Computación (CC3101)

Índice general

Capítulo 1

Nociones Básicas y Lógica

1.1. Crecimiento Asintótico

Usualmente al analizar el costo de un procedimiento computacional nos interesa describir tal costo a grandes rasgos, sin entrar en detalle sobre los factores y las constantes involucradas. Esto es lo que intenta capturar la noción de crecimiento asintótico descrito en esta sección.

1.1.1. La notación O

Considere funciones $f, g : \mathbb{R} \to \mathbb{R}$. Decimos que:

$$f(n) \text{ es } O(g(n)),$$

si existen $c, k \geq 0$ tal que:

$$f(n) \leq c \cdot g(n), \text{ para todo } n \geq k.$$

Esto se ejemplifica gráficamente en la Figura 1.1a.

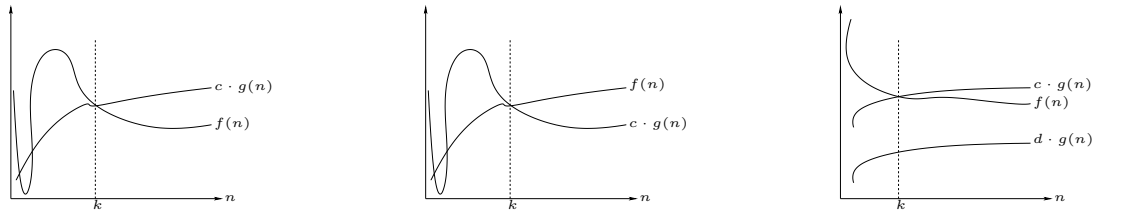
Ejemplo 1.1.1. Se tiene que $x^2 + 2x + 1$ es $O(x^2)$ ya que:

$$x^2 + 2x + 1 \leq 4x^2, \text{ para todo } x \geq 1.$$

Utilizando una generalización de este mismo razonamiento se obtiene que todo polinomio de grado $m \geq 1$ en la variable x es $O(x^m)$. □

Ejercicio. *Demuestre que x^m no es $O(x^{m-1})$ para ningún $m > 1$.*

Pregunta. *¿Es cierto que 2^{2n} es $O(2^n)$? ¿Es cierto que $n!$ es $O(n^n)$? ¿Es cierto que n^n es $O(n!)$? ¿Cómo se comparan $\log n$ y $\sqrt{n}$ en términos de la notación O?*



(a) $f(n)$ es $O(g(n))$. (b) $f(n)$ es $\Omega(g(n))$. (c) $f(n)$ es $\Theta(g(n))$.

Figura 1.1: Ejemplos de crecimiento asintótico de funciones

1.1.2. Crecimiento de la combinación de funciones

El siguiente resultado entrega una cota para el crecimiento asintótico de la suma y el producto de dos funciones.

Proposición 1.1.1. *Asuma que $f_1(n)$ es $O(g_1(n))$ y f_2 es $O(g_2(n))$. Entonces:*

- $(f_1 + f_2)(n) := f_1(n) + f_2(n)$ *es* $O(\max\{g_1(n), g_2(n)\})$.
- $(f_1 \cdot f_2)(n) := f_1(n) \cdot f_2(n)$ *es* $O(g_1(n) \cdot g_2(n))$.

Ejercicio. *Demuestre el resultado anterior.*

Ejemplo 1.1.2. Se tiene que $(x+1)\log(x^2+1) + 3x^2$ es $O(x^2)$ □

1.1.3. Notación Ω y Θ

Análogamente a la notación O, que define un límite superior al crecimiento de una función, utilizamos la notación Ω para expresar un límite inferior a tal crecimiento. En particular, $f(n)$ es $\Omega(g(n))$ si existen $c, k > 0$ tal que:

$$f(n) \geq c \cdot g(n), \text{ para todo } n \geq k.$$

Esto se ejemplifica gráficamente en la Figura 1.1b.

Ejercicio. *Demuestre que:*

$$f(n) \text{ es } \Omega(g(n)) \quad \text{ssi } g(n) \text{ es } O(f(n)).$$

Igualmente, mediante la notación Θ podemos describir que $f(n)$ está acotado por "arriba" y por "abajo" por $g(n)$. En particular, $f(n)$ es $\Theta(g(n))$ si $f(n)$ es ambos $O(g(n))$ y $\Omega(g(n))$. Esto se ejemplifica gráficamente en la Figura 1.1c.

Ejercicio. *Demuestre que $O(n \log n)$ es $\Theta(\log n!)$.*

1.2. Relaciones

Sea A un conjunto (finito o infinito). Una *relación n-aria sobre A* es un subconjunto de:

$$A^n \quad = \quad \underbrace{A \times \cdots \times A}_{n \text{ veces}}.$$

Ejemplo 1.2.1. Las siguientes son relaciones de distinta aridad sobre $\mathbb{N}$:

1. La *primalidad* es una relación unaria que contiene a todos los números naturales primos.
2. El orden $<$ es una relación binaria.
3. La suma $+$ es una relación ternaria definida por todas las tuplas $(a, b, c) \in \mathbb{N}^3$ tal que $a + b = c$.

□

Pregunta. ¿Cuántas relaciones n-arias existen en un conjunto A de cardinalidad m?

1.2.1. Propiedades de las relaciones binarias

Las siguientes condiciones definen propiedades importantes de algunas relaciones binarias:

- **Reflexividad:** Una relación R es *refleja* si para todo $a \in A$ se tiene que $(a, a) \in R$.
- **Simetría:** La relación R es *simétrica* si $(a, b) \in R$ implica $(b, a) \in R$, para todo $a, b \in A$.
- **Antisimetría:** La relación R es *antisimétrica* si para todo $a, b \in A$ tal que $(a, b) \in R$ y $(b, a) \in R$, se tiene que $a = b$.
- **Transitividad:** La relación R es *transitiva* si para todo $a, b, c \in A$, si $(a, b) \in R$ y $(b, c) \in R$, entonces $(a, c) \in R$.

Pregunta. ¿Cuáles de estas propiedades son satisfechas por $<$, $\leq$, y la relación de divisibilidad en $\mathbb{N}$?

1.2.2. Operaciones con relaciones

Dado que las relaciones son conjuntos, podemos definir todas las operaciones Booleanas básicas sobre ellas. Es decir, podemos definir la *unión* $R \cup R'$, la *intersección* $R \cap R'$, y la *diferencia* $R \setminus R'$ de relaciones R, R'.

Es posible definir otras operaciones de interés en el importante caso cuando las relaciones son binarias. Por ejemplo, el *inverso* R^{-1} de R contiene precisemante los pares $(a, b) \in A \times A$ tal que $(b, a) \in R$. Esta operación nos permite caracterizar la simetría de una relación:

Ejercicio. *Demuestre que R es simétrico ssi $R = R^{-1}$.*

Otra operación importante es la *composición* de las relaciones binarias R y R', que definimos como:

$$R \circ R' \quad = \quad \{(a, b) \mid \text{existe } c \in A \text{ tal que } (a, c) \in R \text{ y } (c, b) \in R'\}.$$

Esta operación nos permite caracterizar la transitividad de una relación:

Ejercicio. *Demuestre que R es transitivo ssi $R \circ R \subseteq R$.*

1.2.3. Clausuras

Si R no es refleja, fácilmente podemos "volverla" refleja agregando todos los pares $(a, a) \in A \times A$ que no están en R. Llamamos al conjunto R_{refl} obtenido la *clausura refleja* de R. Tal relación no solo contiene a R y es refleja, sino que además, intuitivamente, R_{refl} se obtiene desde R agregando la "menor" cantidad posible de pares para volverla refleja. Formalmente, decimos que R_{refl} es la *menor* relación refleja que contiene a R. Esto quiere decir que para toda relación R' que es refleja y contiene a R, se tiene que $R_{\text{refl}} \subseteq R'$.

Ejercicio. *Demuestre que efectivamente R_{refl} es la menor relación refleja que contiene a R.*

Pregunta. ¿Cómo es posible definir análogamente la clausura simétrica R_{sim} de R?

El caso de la clausura transitiva es más interesante. Note que si R no es transitiva, podemos intentar volverla transitiva agregando los pares que violan la condición de transitividad (es decir, agregamos todos los pares en $R \circ R$ que no pertenecen a R). Note, sin embargo, que este proceso debe realizarse recursivamente, ya que al insertar el par (a, c) en $R \circ R \setminus R$ podríamos generar una nueva violación de la condición de transitividad. Es natural entonces agregar entonces los pares en $R \circ (R \circ R) \setminus R \circ R$, en $R \circ (R \circ (R \circ R)) \setminus R \circ (R \circ R)$, y así sucesivamente. Si el conjunto A es infinito este proceso podría no detenerse. En caso contrario, en algún momento se estabilizará.

Definimos entonces recursivamente $R^1 := R$ y $R_{n+1} := R^n \circ R$, para $n \geq 1$. A partir de esto definimos $R_{\text{cl}} := \bigcup_{i \geq 1} R^i$. Note que, intuitivamente, R_{cl} corresponde a la relación obtenida como se explica en el párrafo anterior para eliminar desde R todas las violaciones de la condición de transitividad.

Ejercicio. *Demuestre que R_{cl} es la clausura transitiva de R. Es decir, demuestre lo siguiente:*

1. $R \subseteq R_{\text{cl}}$.
2. R_{cl} *es transitiva.*
3. *Para toda relación transitiva R^* tal que $R \subseteq R^*$, se tiene que $R_{\text{cl}} \subseteq R^*$.*

Pregunta. Si R es la relación de sucesor en $\mathbb{N}$, es decir, $R = \{(i, i+1) \mid i \geq 1\}$, ¿a qué corresponde R_{cl}?

Ejercicio. *Demuestre que si A tiene n elementos, entonces $R_{\text{cl}} = \bigcup_{1 \leq i \leq n} R^i$.*

Pregunta. ¿Cuál es el costo de computar la clausura transitiva de una relación con n elementos?

Observación. Note que por definición R_{refl}, R_{sim}, y R_{cl} deben ser únicas.

1.2.4. Relaciones de equivalencia

R es una *relación de equivalencia* si es refleja, simétrica, y transitiva. Si R es relación de equivalencia, escribimos $a \sim b$ si $(a, b) \in R$. Además, para $a \in A$ definimos su *clase de equivalencia en R* como $[a]_R = \{b \mid a \sim b\}$.

Pregunta. ¿Puede encontrar una relación de equivalencia no trivial en $\mathbb{N}$? ¿Cuáles son las clases de equivalencia definidas por tal relación?

Interesantemente, toda clase de equivalencia R "particiona" el conjunto A sobre la que está definida. Es decir, todo elemento de A pertenece a alguna clase de equivalencia de R y las clases de equivalencia de R son mutuamente disjuntas. Se le pide demostrar formalmente esto en el próximo ejercicio.

Ejercicio. *Sea R una clase de equivalencia en A. Demuestre que las siguientes condiciones son equivalentes:*

1. $a \sim b$.
2. $[a]_R = [b]_R$.
3. $[a]_R \cap [b]_R \neq \emptyset$.

1.2.5. Órdenes

Un *orden parcial* en A es una relación binaria refleja, antisimétrica, y transitiva. Comúnmente se denota como $(A, \preceq)$.

Ejemplo 1.2.2. La relación $\leq$ es un orden parcial en $\mathbb{N}$. Lo mismo la divisibilidad. La relación $\subseteq$ es una orden parcial en $2^{\mathbb{N}}$. □

Un *orden total* $(A, \leq)$ es un orden parcial tal que para todo $a, b \in A$ se tiene que $a \leq b$ o $b \leq a$.

Pregunta. ¿Cuál de los anteriores órdenes parciales son también totales?

Ejercicio. *Sea A un conjunto finito y suponga que $(A, \preceq)$ es un orden parcial. Demuestre que existe un orden total $(A, \leq)$ tal que $a \preceq b$ implica $a \leq b$, para todo $a, b \in A$.*

Un *buen orden de A* es un orden total que cumple que todo subconjunto no vacío S de A tiene un menor elemento con respecto a $\leq$.

Pregunta. ¿Es $(\mathbb{N}, \leq)$ un buen orden? ¿Y $(\mathbb{Z}, \leq)$?

Ejercicio. *Demuestre que la relación lexicográfica en $\mathbb{N}$ definida como*

$$(a_1, b_1) \preceq (a_2, b_2) \quad \iff \quad a_1 < b_1 \text{ o } (a_1 = b_1 \text{ y } a_2 < b_2)$$

en un buen orden.

Ejercicios Finales

1. Sea R una relación binaria en un conjunto A. La *relación de equivalencia generada por R en A* es la "menor" relación S en A que es de equivalencia y contiene a R. En otras palabras, para todo S' en A que contiene a R y es de equivalencia se tiene que $S \subseteq S'$.

   Demuestre que para todo $a, b \in A$ se tiene que $(a, b) \in S$ si y solo si existen elementos $a_1, \ldots, a_n \in A$ tal que $a_1 = a$, $a_n = b$, y $(a_i, a_{i+1}) \in R \cup R^-$, para todo $1 \leq i < n$.

2. Sea R_e la clausura transitiva de la clausura simétrica de la clausura refleja de una relación R. Demuestre que R_e es la *menor* relación de equivalencia que contiene a R (donde "menor"se define con respecto al orden parcial definido por $\subseteq$ sobre el conjunto de relaciones en A).

3. *a*) ¿Es cierto que existen un conjunto A y relaciones binarias R y S sobre A tal que el resultado de la siguiente expresión es no vacío?

$$\big((R \circ R) \circ R\big) \; - \; \Big(\big(((R \circ R) - S) \circ R\big) \cup (S \circ R)\Big).$$

   b) Demuestre que existe un conjunto A y una relación binaria R tal que el resultado de la siguiente expresión es no vacío:

$$\big(R \circ (R \cap (R \circ R))\big) \; - \; \big(((R \circ R) - R) \circ R\big).$$

4. Sea R una relación en A. Además, definimos a R^+ como $R^* \cup I$, donde $I = \{(a, a) \mid a \in A\}$ es la identidad en A.

   Una relación R es *euclideana* si satisface la siguiente fórmula:

$$\forall x \forall y \forall z \big((R(x, y) \wedge R(x, z)) \; \rightarrow \; R(y, z)\big).$$

   Demuestre lo siguiente:

   a) Si R es una relación en A, entonces la relación $(R^{-1} \circ (R \cup R^{-1})^+ \circ R)$ es simétrica, transitiva, y euclideana.

   b) Si R es euclideana, entonces $(R^{-1} \circ (R \cup R^{-1})^+ \circ R) \subseteq R$.

   Ayuda: Note que $S \circ I = S$. Además, $(S \circ T)^{-1} = T^{-1} \circ S^{-1}$. En particular, $(R \circ R^{-1})$ es simétrica.

5. Sea $\mathcal{F}$ el conjunto de todas las funciones de $\mathbb{R}$ en $\mathbb{R}$.

   a) Defina una relación binaria $\preceq_O$ en $\mathcal{F}$ tal que para cada $f_1, f_2 \in \mathcal{F}$, $f_1 \preceq_O f_2$ si y sólo si f_1 es $O(f_2)$. ¿Es $\preceq_O$ un orden parcial en $\mathcal{F}$?

   b) Defina una relación binaria $\preceq_\Theta$ en $\mathcal{F}$ tal que para cada $f_1, f_2 \in \mathcal{F}$, $f_1 \preceq_\Theta f_2$ si y sólo si f_1 es $\Theta(f_2)$. Demuestre que $\preceq_\Theta$ una relación de equivalencia en $\mathcal{F}$.

   c) Defina $\mathcal{F}_\Theta$ como el conjunto de las clases de equivalencia de $\mathcal{F}$ con respecto a la relación de equivalencia Θ. Defina la relación $\preceq_{O,\Theta}$ en $\mathcal{F}_\Theta$ de la siguiente forma: Para todo $[f_1], [f_2] \in \mathcal{F}_\Theta$, $[f_1] \preceq_{O,\Theta} [f_2]$ si y sólo si existe $f \in [f_1]$ y $f' \in [f_2]$ tal que f es $O(f')$.
   Demuestre que $\preceq_{O,\Theta}$ es un orden parcial en $\mathcal{F}_\Theta$.

   d) Demuestre que $\preceq_{O,\Theta}$ no es un orden total en $\mathcal{F}_\Theta$.

6. Una relación R en A es un *cuasi-orden* si es refleja y transitiva.

   - Demuestre que si R es un cuasi-orden en A, entonces $R \cap R^{-1}$ es una relación de equivalencia en A.

- Sea S una relación en las clases de equivalencia de $R \cap R^{-1}$ tal que $(C, D) \in S$ si y solo si existen elementos $c \in C$ y $d \in D$ tal que $(c, d) \in R$. Demuestre que S es un orden parcial.

7. Sea $(A, \preceq)$ un orden parcial (i.e. $\preceq$ es una relación refleja, antisimétrica y transitiva). Decimos que el elemento $a \in A$ es una *cota inferior* para $B \subseteq A$, si para todo $b \in B$ se tiene que $a \preceq b$. Además, a es un *ínfimo* para B, si es una cota inferior para B y para toda cota inferior $a' \in A$ para B se tiene que $a' \preceq a$.

   Asuma que $(A, \preceq)$ es un orden parcial que satisface lo siguiente:

   - $(A, \preceq)$ es *inferiormente completo*; esto quiere decir que todo subconjunto no vacío B de A, que tiene una cota inferior en A, también tiene un ínfimo en A;
   - $(A, \preceq)$ tiene un *máximo*; esto es, existe elemento $a \in A$ tal que para todo elemento $a' \in A$ se tiene que $a' \preceq a$; y
   - $(A, \preceq)$ tiene un *mínimo*; esto es, existe elemento $a \in A$ tal que para todo elemento $a' \in A$ se tiene que $a \preceq a'$.

   Sea $f : A \to A$ una función *monótona*; esto es, f satisface que $f(a) \preceq f(b)$ cada vez que $a \preceq b$ $(a, b \in A)$.

   Demuestre lo siguiente):

   a) El conjunto $B = \{a \in A \mid f(a) \preceq a\}$ tiene ínfimo.

   b) Sea b^* el ínfimo de B. Demuestre que $f(b^*)$ es una cota inferior para B.

   c) Demuestre que $f(b^*)$ está en B.

   d) Demuestre finalmente que $b^* = f(b^*)$.

8. Sea R una relación en el conjunto A que es refleja y transitiva.

   a) Demuestre que $R \cap R^{-1}$ es una relación de equivalencia en A.

   b) Defina una relación S sobre el conjunto de clases de equivalencia definidos por $R \cap R^{-1}$, de tal forma que $(C, D) \in S$, donde C y D son clases de equivalencia de $R \cap R^{-1}$, si y solo si existe elemento $c \in C$ y $d \in D$ tal que $(c, d) \in R$. Demuestre que S es un orden parcial.

9. Si X es un conjunto, entonces 2^X es el conjunto potencia de X (es decir, $2^X = \{Y \mid Y \subseteq X\}$).

   Sea E un conjunto y A un subconjunto no vacío de E. Defina una relación $\mathcal{R}$ sobre 2^E, de tal forma que $(E_1, E_2) \in \mathcal{R}$ si y solo si $A \setminus E_1 = A \setminus E_2$, para todo $E_1, E_2 \in 2^E$.

   - Demuestre que $\mathcal{R}$ es una relación de equivalencia en 2^E.
   - Demuestre que el conjunto de clases de equivalencia definidas por $\mathcal{R}$ en 2^E es igual a $\{[X]_{\mathcal{R}} \mid X \subseteq A\}$.

10. Sean $(S_1, \preceq_1)$ y $(S_2, \preceq_2)$ órdenes (parciales). Definimos su *producto cartesiano* $(S_{1,2}, \preceq_{1,2})$ de tal forma que: (i) $S_{1,2} = S_1 \times S_2$, y (ii) $(a, b) \preceq_{1,2} (c, d)$ si y solo si $a \preceq_1 c$ y $b \preceq_2 d$. Demuestre que si $|S_1|, |S_2| \geq 2$, entonces $(S_{1,2}, \preceq_{1,2})$ no es un orden total.

11. Una relación binaria R sobre A es *acíclica* si no existe $a \in A$ tal que $(a, a) \in R^*$.

    Sean R y S dos relaciones acíclicas sobre el mismo conjunto A tal que $R^* = S^*$. Asuma que existe un par $(a, b) \in R - S$. Demuestre entonces que $(R - \{(a, b)\})^* = R^*$.

1.3. Lógica Proposicional

La lógica es una herramienta que se utiliza para formalizar el razonamiento. También sirve para expresar formalmente propiedades complejas de las relaciones. Esta última corresponde a la *lógica relacional*, que estudiaremos más adelante en el capítulo. A modo de introducción, es útil comenzar estudiando una lógica mucho menos poderosa, llamada *lógica proposicional*, la que permite expresar propiedades de las *proposiciones*; es decir, oraciones que tienen un valor de verdad (en este caso *verdadero* o *falso*, también denotados por 0 y 1, respectivamente).

1.3.1. Sintaxis

Supongamos que tenemos un conjunto $P = \{p, q, r, \dots\}$ de proposiciones. La *lógica proposicional sobre* P, denotada por $\mathcal{L}(P)$, se define inductivamente de la siguiente forma:

- **Caso base:** Si $p \in P$, entonces $p \in \mathcal{L}(P)$.
- **Casos inductivos:** Si $\phi, \psi \in \mathcal{L}(P)$, entonces:
  - $(\neg\phi) \in \mathcal{L}(P)$.
  - Si $\phi, \psi \in \mathcal{L}(P)$, entonces $(\phi \star \psi) \in L(P)$, donde $\star \in \{\vee, \wedge, \to\}$.

Ejercicio. *Demuestre que toda fórmula en $\mathcal{L}(P)$ tiene tantos paréntesis izquierdos como derechos.*

De ahora en adelante, evitaremos escribir los paréntesis externos de las fórmulas (ya que sacarlos no causa ambigüedad). Además, asumiremos que la negación siempre aplica a la fórmula más cercana que se tiene a la derecha. Esto quiere decir, por ejemplo, que la fórmula $\neg\alpha \vee \beta$ debe leerse como $\neg(\alpha) \vee \beta$.

1.3.2. Semántica

Las fórmulas en $\mathcal{L}(P)$ tienen un valor de verdad. Lo importante es que tal valor solo depende del valor de verdad de las proposiciones en P y la interpretación de los conectivos lógicos. Formalizaremos esta idea a continuación. Sea

$$\sigma : P \to \{0, 1\}$$

una *valuación*, es decir una función que le asigna valores de verdad a las proposiciones en P. Definimos entonces

$$\hat{\sigma} : \mathcal{L}(P) \to \{0, 1\}$$

como sigue:

- **Caso base:** Si $p \in P$, entonces $\hat{\sigma}(p) = \sigma(p)$.
- **Casos inductivos:** Si $\phi, \psi \in \mathcal{L}(P)$, entonces:
  1. $\hat{\sigma}(\neg\phi) = 1 - \hat{\sigma}(\phi)$.
  2. $\hat{\sigma}(\phi \vee \psi) = \text{máx}\{\hat{\sigma}(\phi), \hat{\sigma}(\psi)\}$.
  3. $\hat{\sigma}(\phi \wedge \psi) = \text{mín}\{\hat{\sigma}(\phi), \hat{\sigma}(\psi)\}$.
  4. $\hat{\sigma}(\phi \to \psi) = \text{máx}\{1 - \hat{\sigma}(\phi), \hat{\sigma}(\psi)\}$.

Dado que $\hat{\sigma}$ está únivocamente definido por σ, de ahora en adelante no distinguimos entre $\sigma : P \to \{0, 1\}$ y $\hat{\sigma} : \mathcal{L}(P) \to \{0, 1\}$.

Pregunta. ¿Es posible expresar la *disyunción exclusiva* con estos conectivos?

1.3.3. Equivalencia

Dos fórmulas α y β son *equivalentes*, denotado $\alpha \equiv \beta$, si $\sigma(\alpha) = \sigma(\beta)$ para todo $\sigma : P \to \{0, 1\}$.

Ejercicio. *Demuestre que:*

$$\alpha \vee (\beta \vee \gamma) \equiv (\alpha \vee \beta) \vee \gamma \qquad \neg(\neg\phi) \equiv \phi$$
$$\alpha \wedge (\beta \wedge \gamma) \equiv (\alpha \wedge \beta) \wedge \gamma \qquad (\alpha \to \beta) \equiv (\neg\alpha \vee \beta)$$
$$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta) \qquad \neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$$

Dado que $\vee$ y $\wedge$ son asociativos, de ahora en adelante escribiremos simplemente $\alpha \vee \beta \vee \gamma$ y $\alpha \wedge \beta \wedge \gamma$, respectivamente, sin importar como los paréntesis están distribuídos. Note que la equivalencia $(\alpha \to \beta) \equiv (\neg\alpha \vee \beta)$ nos dice que podemos prescindir del operador $\to$ y no perder expresividad en el lenguaje básico (aunque es conveniente utilizarlo ya que facilita la lectura de algunas fórmulas). Las últimas dos equivalencias, llamadas "leyes de Morgan", expresan que de igual forma podemos eliminar a $\vee$ o $\wedge$, respectivamente, del lenguaje básico sin perder poder expresivo.

Pregunta. *¿Es cierto que $\to$ es un conectivo asociativo?*

Pregunta. *¿Cuánto cuesta verificar si dos fórmulas son equivalentes?*

1.3.4. Expresividad de los conectivos

Sea P un conjunto finito de proposiciones. Cada fórmula en $\mathcal{L}(P)$ tiene una "tabla de verdad" asociada, la que corresponde al valor de verdad que esta toma en cada una de las valuaciones $\sigma : P \to \{0, 1\}$. (Recuerde que si dos fórmulas tienen la misma tabla de verdad, entonces decimos que son equivalentes, es decir, son semánticamente la misma fórmula).

Pregunta. *¿Cuántas tablas de verdad distintas existen sobre $P = \{p_1, \ldots, p_n\}$?*

¿Es cierto por otro lado que cada tabla de verdad sobre $P = \{p_1, \ldots, p_n\}$ puede ser "expresada" por medio de una fórmula en $\mathcal{L}(P)$? Es decir, dada una tabla de verdad, ¿existe una fórmula en $\mathcal{L}(P)$ que tiene precisamente esa tabla de verdad? La respuesta a esta pregunta es afirmativa, por lo que se dice que el conjunto de conectivos lógicos $\{\neg, \vee, \wedge, \to\}$ es *funcionalmente completo*. Explicamos a continuación por qué esto es el caso. Es conveniente comenzar con un ejemplo.

Ejemplo 1.3.1. Considere la tabla de verdad que se muestra a continuación:

	p_1	p_2	p_3	Valor de verdad
σ_0	0	0	0	0
σ_1	0	0	1	1
σ_2	0	1	0	0
σ_3	0	1	1	0
σ_4	1	0	0	1
σ_5	1	0	1	0
σ_6	1	1	0	1
σ_7	1	1	1	0

Concentrémonos en las valuaciones σ_i tal que el valor de verdad en σ_i es 1. Esto sucede, precisamente, para $i = 1, 4, 6$. Cada una de estas valuaciones puede ser "descrita" mediante una formula ϕ_i en $\mathcal{L}(P)$, donde $P = \{p_1, p_2, p_3\}$. En particular:

$$\phi_1 := \neg p_1 \wedge \neg p_2 \wedge p_3 \qquad \phi_4 := p_1 \wedge \neg p_2 \wedge \neg p_3 \qquad \phi_6 := p_1 \wedge p_2 \wedge \neg p_3.$$

Es fácil demostrar entonces que la fórmula $\phi := \phi_1 \vee \phi_4 \vee \phi_6$ tiene precisamente la tabla de verdad arriba descrita. □

	p_1	p_2	...	p_n	Valor de verdad
σ_1	0	0	...	0	b_1
σ_2	0	0	...	1	b_2
...	...	...	...	...	...
...	...	...	...	...	...
σ_{2^n}	1	1	...	1	b_{2^n}

Figura 1.2: Tabla de verdad arbitraria sobre $P = \{p_1, \ldots, p_n\}$.

Suponga ahora, en general, que se nos entrega una tabla de verdad arbitraria definida sobre $P = \{p_1, \ldots, p_n\}$ como en la Figura 1.2. En tal figura cada valor b_j, para $1 \leq j \leq 2^n$, representa un valor de verdad en $\{0, 1\}$. Podemos "describir" esta tabla de verdad concentrándonos en las valuaciones σ_j tal que $b_j = 1$. Cada una de aquellas valuaciones puede ser "descrita" mediante una fórmula:

$$\phi_j \quad := \quad \Big(\bigwedge_{\{i \,|\, 1 \leq i \leq n,\, \sigma_j(p_i) = 1\}} p_i \Big) \wedge \Big(\bigwedge_{\{k \,|\, 1 \leq k \leq n,\, \sigma_j(p_k) = 0\}} \neg p_k \Big).$$

Esta fórmula ϕ_j determina el valor de verdad de cada proposición $p \in P$ con respecto al valor que le asigna σ_j. En el próximo ejercicio se pide demostrar que al tomar la disyunción de todas las fórmulas ϕ_j donde $b_j = 1$ es posible definir precisamente la tabla de verdad mostrada en la Figura 1.2.

Ejercicio. *Demuestre que la semántica determinada por la tabla de verdad que se muestra en la Figura 1.2 corresponde precisamente a la definida por:*

$$\bigvee_{\{j \,|\, 1 \leq j \leq 2^n,\, b_j = 1\}} \phi_j.$$

De lo anterior concluimos que basta utilizar los conectivos $\{\neg, \vee, \wedge\}$ para expresar todas las posibles tablas de verdad (equivalentemente, todas las fórmulas en $\mathcal{L}(P)$). Dado que es posible reescribir $(\alpha \vee \beta)$ como $(\neg\alpha \wedge \neg\beta)$, y equivalentemente $(\alpha \wedge \beta)$ como $(\neg\alpha \vee \neg\beta)$, obtenemos el siguiente importante resultado:

Teorema 1.3.1. *Los conjuntos $\{\neg, \vee\}$ y $\{\neg, \wedge\}$ son funcionalmente completos.*

En el siguiente ejercicio se le pide demostrar que al eliminar la negación de nuestro conjunto original de conectivos perdemos esta propiedad:

Ejercicio. *Demuestre que $\{\vee, \wedge, \rightarrow\}$ no es funcionalmente completo.*

Por otro lado, existe un conectivo binario que por sí solo es funcionalmente completo:

Ejercicio. *Demuestre que $\alpha \mid \beta$, definido como $\neg(\alpha \wedge \beta)$, es funcionalmente completo.*

1.3.5. Consecuencia Lógica y Satisfacibilidad

Llegamos a la noción más importante de nuestro estudio: *consecuencia lógica*. Esta nos dice cuando una fórmula ϕ – la *conclusión* – se sigue lógicamente de un conjunto Δ de otras fórmulas – las *premisas*. Esto ocurre cuando la verdad de ϕ es implicada por la verdad de las fórmulas en Δ. De ahora en adelante, escribimos $\sigma(\Delta)$ para representar que $\sigma(\alpha) = 1$ para cada $\alpha \in \Delta$.

Definición 1.3.1. *Decimos que ϕ es consecuencia lógica de Δ, denotado por $\Delta \models \phi$, si para cada valuación $\sigma : P \to \{0, 1\}$ se cumple que:*

$$\textit{Si } \sigma(\Delta) = 1 \textit{ entonces } \sigma(\phi) = 1.$$

Ejemplo 1.3.2. No es difícil observar que $\{p, p \to q\} \models q$. Esto se conoce como *modus ponens*. La demostración se obtiene directamente de observar la tabla de verdad del conjunto de proposiciones $\{p, q\}$ y la fórmula $p \to q$:

	p	q	$p \to q$
σ_0	0	0	1
σ_1	0	1	1
σ_2	1	0	0
σ_3	1	1	1

En efecto, concentrémonos en el conjunto de valuaciones σ_j tal que $\sigma_j(\{p, p \to q\}) = 1$; es decir, aquellas que satisfacen el conjunto de premisas. Observando la tabla notamos que esto ocurre únicamente cuando $j = 3$. Dado que $\sigma_3(q) = 1$ concluímos que q es consecuencia lógica de $\{p, p \to q\}$. □

Ejercicio. *Demuestre que* $\{p \to q,\ q \to r\} \models p \to r$.

Ejercicio. *Formalice el siguiente argumento en el cálculo proposicional:*

> *"Si Superman fuera capaz y deseara prevenir el mal, entonces lo haría. Si Superman fuera incapaz de prevenir el mal, entonces sería impotente, y si no deseara prevenir el mal, entonces sería malévolo. Si Superman existe, no es ni impotente ni malévolo. Superman no previene el mal. Entonces, Superman no existe."*

Demuestre que 'Superman no existe' es consecuencia lógica de esta formalización.

Pregunta. ¿Qué fórmulas son consecuencia lógica de un conjunto "contradictorio", como por ejemplo $\Delta = \{p, \neg p\}$? ¿Qué fórmulas son consecuencia lógica del conjunto vacío $\Delta = \emptyset$? (*Ayuda:* En el primer caso son todas las fórmulas; en el segundo caso solo las *tautologías*, es decir, aquellas cuyo valor de verdad es 1 en todas las valuaciones. Explique esto).

Ejercicio. *Demuestre que:*

$$\Delta \models \phi \to \psi \quad \iff \quad \Delta \cup \{\phi\} \models \psi.$$

Ejercicio. *En este ejercicio se le pide demostrar que la noción de consecuencia lógica es* monótona*; es decir:*

$$\textit{Si } \Delta \models \phi \textit{ entonces para toda fórmula } \psi \textit{ se tiene que } \Delta \cup \{\psi\} \models \phi.$$

Pregunta. *¿Le merece algún comentario el hecho de que la noción de consecuencia lógica sea monótona?*

1.3.6. Satisfacibilidad

La consecuencia lógica está directamente relacionada con la importante noción de satisfacibilidad de una fórmula. Decimos que un conjunto Δ de fórmulas en $\mathcal{L}(P)$ es *satisfacible* si existe una valuación $\sigma : P \to \{0, 1\}$ tal que $\sigma(\Delta) = 1$.

Pregunta. *¿Cuánto cuesta verificar si una fórmula es satisfacible?*

A continuación se le pide demostrar la siguiente propiedad que relaciona la consecuencia lógica con la satisfacibilidad de una fórmula:

Ejercicio. *Demuestre que:*

$$\Delta \models \phi \quad \iff \quad (\Delta \cup \{\neg\phi\}) \textit{ es insatisfacible.}$$

1.3.7. Expresando propiedades en la lógica proposicional

Suponga que tenemos un conjunto de proposiciones que expresan los estados de un semáforo en diferentes instantes de tiempo. Tal conjunto de proposiciones se define como sigue:

$$P \quad := \quad \{v_i \mid 1 \leq i \leq n\} \cup \{a_i \mid 1 \leq i \leq n\} \cup \{r_i \mid 1 \leq i \leq n\},$$

donde v_i, a_i, y r_i representan que el semáforo está en color verde, amarillo o rojo, respectivamente, en el estado $1 \leq i \leq n$. La lógica proposicional nos permite entonces expresar varias propiedades interesantes. Por ejemplo:

- En cada instante de tiempo el semáforo está en al menos un color:

$$\bigwedge_{1 \leq i \leq n} v_i \vee a_i \vee r_i.$$

- En ningún instante de tiempo el semáforo tiene dos colores:

$$\bigwedge_{1 \leq i \leq n} \neg(v_i \wedge a_i) \wedge \neg(v_i \wedge r_i) \wedge \neg(a_i \wedge r_i).$$

  (Note que hay varias maneras equivalentes de expresar esto).

- Cada vez que el semáforo está en amarillo en el siguiente instante de tiempo está en rojo:

$$\bigwedge_{1 \leq i < n} a_i \rightarrow r_{i+1}.$$

- Si el semáforo está en rojo, entonces estará en verde en a lo más 5 unidades de tiempo:

$$\bigwedge_{1 \leq i < n-5} (r_i \rightarrow v_{i+1} \vee v_{i+2} \vee v_{i+3} \vee v_{i+4} \vee v_{i+5}).$$

Pregunta. ¿Cuál es el tamaño de la fórmula en términos de n?

Ejercicio. *Una relación binaria R sobre un conjunto finito A se dice* 3-coloreable*, si es posible asignarle uno de tres colores $\{c_1, c_2, c_3\}$ a cada elemento $a \in A$, de tal forma que si $(a, b) \in R$, para cualquier $a, b \in A$, entonces a y b reciben colores distintos. Exprese mediante una fórmula en $\mathcal{L}(P)$ el hecho de que R es 3-coloreable, donde P es el conjunto de proposiciones:*

$$\{p_{a,c_i} \mid a \in A,\, i = 1, 2, 3\}.$$

Intuitivamente, la proposición p_{a,c_i} expresa el hecho de que el elemento $a \in A$ ha sido coloreado con c_i.

Más formalmente, dada una relación binaria R sobre A, construya una fórmula ϕ_R tal que R es 3-coloreable ssi ϕ_R es satisfacible. Todo el proceso de construcción de ϕ_R debe tomar a lo más $O(n^2)$ pasos, donde $n = |A|$.

1.3.8. Circuitos digitales

Una tabla de verdad puede ser vista como un "circuito digital" que toma como entrada las proposiciones en P y entrega como salida uno (o más) valores de verdad. Como cada tabla de verdad puede ser representada por una fórmula en $L(P)$, podemos entonces utilizar nuestros conectivos lógicos (los que corresponden al símil de las compuertas lógicas `AND`, `OR`, and `NOT`) para diseñar circuitos con funcionalidades interesantes. Esto es lo que se le pide realizar en el siguiente ejercicio.

Ejercicio. *Diseñe un circuito digital que tome como entrada números binarios*

$$a_n a_{n-1} \cdots a_1 a_0 \quad y \quad b_n b_{n-1} \cdots b_1 b_0 \quad (n \geq 0),$$

y compute como salida un número binario $c_{n+1} c_n \cdots c_1 c_0$ *tal que* $c_{n+1} c_n \cdots c_1 c_0$ *es la representación binaria de* $u + v$*, donde:*

- u *es el entero cuya representación binaria es* $a_n a_{n-1} \cdots a_1 a_0$*, y*
- v *es el entero cuya representación binaria es* $b_n b_{n-1} \cdots b_1 b_0$*.*

Ejercicios Finales

1. Dado oraciones α, β de la lógica proposicional, demuestre que $\alpha \models \beta$ y $\beta \models \alpha$ si y sólo si $\models (\alpha \leftrightarrow \beta)$ (es decir, $\alpha \models \beta$ y $\beta \models \alpha$ si y sólo si $(\alpha \leftrightarrow \beta)$ es una tautología).

2. Dado conjunto de oraciones Σ de la lógica proposicional, además de oraciones α y β, demuestre que si α es una tautología, entonces $\Sigma \cup \{\alpha\} \models \beta$ si y sólo si $\Sigma \models \beta$.

3. Dado conjunto de oraciones Σ de la lógica proposicional, además de oraciones φ, ψ y θ, demuestre que si $\varphi \rightarrow \psi$ es una tautología, entonces $\Sigma \cup \{\varphi, \psi\} \models \theta$ si y sólo si $\Sigma \cup \{\varphi\} \models \theta$.

4. Dado conjunto de oraciones Σ de la lógica proposicional, además de oraciones α y β tal que α y $\Sigma \cup \{\beta\}$ no tienen variables proposicionales en común. ¿Es cierto que $\Sigma \models \beta$ si y sólo si $\Sigma \cup \{\alpha\} \models \beta$?

5. Sea Σ un conjunto de oraciones en la lógica proposicional y α, β oraciones cualesquiera. ¿Es cierto que si $\Sigma \models \alpha \vee \beta$ entonces $\Sigma \models \alpha$ o $\Sigma \models \beta$? Fundamente su respuesta.

6. Sea Σ un conjunto finito de oraciones en la lógica proposicional y α una oración cualquiera. Demuestre que existe una oración θ tal que $\Sigma \models \alpha$ si y solo si $\models \theta$.

7. Sean $p_1, \ldots, p_n$ variables proposicionales distintas, $n \geq 1$. Demuestre que la fórmula

   $$(\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \leftrightarrow \cdots \leftrightarrow p_n)$$

   es cierta en una valuación σ si y solo si el número de p_i's, $1 \leq i \leq n$, a los cuales σ le asigna el valor 0 es par.

8. Sea p una variable proposicional. Para oraciones proposicionales α, β defina recursivamente la *oración obtenida desde* α *reemplazando* p *por* β, denotado por $\alpha[\beta/p]$, como sigue:

   - $\alpha[\beta/p] \;=\; \begin{cases} \alpha & \text{si } \alpha \text{ es proposición atómica y } \alpha \neq p \\ \beta & \text{si } \alpha = p \end{cases}$
   - $(\alpha_1 \vee \alpha_2)[\beta/p] \;=\; \alpha_1[\beta/p] \vee \alpha_2[\beta/p]$
   - $(\neg\alpha)[\beta/p] \;=\; \neg\alpha[\beta/p]$

   Demuestre que si α_1 y α_2 son equivalentes entonces para todo oración β se tiene que $\beta[\alpha_1/p]$ y $\beta[\alpha_2/p]$ también son equivalentes.

9. Defina recursivamente el *dual* de una oración proposicional ϕ, denotado por ϕ^*, como sigue:

   - $p^* = \neg p$, si $p \in P$;
   - $(\alpha \vee \beta)^* = \alpha^* \wedge \beta^*$;
   - $(\alpha \wedge \beta)^* = \alpha^* \vee \beta^*$;

- $(\neg\alpha)^* = \neg\alpha^*$.

Demuestre que para toda oración proposicional ϕ se tiene que ϕ^* es equivalente a $\neg\phi$.

10. Sea $\oplus$ el conectivo lógico binario definido como sigue: El valor de verdad de $\alpha \oplus \beta$ es 1 si y sólo si el valor de verdad de α es distinto del valor de verdad de β.

    ¿Es el conjunto $\{\oplus, \leftrightarrow\}$ de conectivos lógicos funcionalmente completo?

11. El conectivo ternario `MAYORIA` es definido de la siguiente forma:

p	q	r	`MAYORIA`(p, q, r)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

    Demuestre que el conectivo `MAYORIA` por si solo no puede expresar todas las oraciones proposicionales que utilizan las variables proposicionales p, q, r.

12. ¿Es $\{\neg, \texttt{MAYORIA}\}$ funcionalmente completo?

13. El conectivo unario $\perp$ es definido de la siguiente forma:

p	$\perp p$
0	0
1	0

    Este conectivo usualmente se denota sin la letra proposicional porque su valor de verdad es siempre 0 (por ejemplo, denotamos $p \wedge (\perp q)$ como $p \wedge \perp$).

    Demuestre que con los conectivos $\{\neg, \texttt{MAYORIA}, \perp\}$ es posible expresar todas las oraciones proposicionales (es decir, todas las oraciones que pueden ser escritas con los conectivos usuales $\{\neg, \vee, \wedge\}$).

14. Diseñe un circuito digital que tome como entrada dos números binarios $a_n a_{n-1} \cdots a_1 a_0$ y $b_n b_{n-1} \cdots b_1 b_0$, $n \geq 0$, y compute como salida un número binario $c_n c_{n-1} \cdots c_1 c_0$ tal que $c_n c_{n-1} \cdots c_1 c_0$ es la representación binaria de $u - v$, donde:

    - u es el entero cuya representación binaria es $a_n a_{n-1} \cdots a_1 a_0$, y
    - v es el entero cuya representación binaria es $b_n b_{n-1} \cdots b_1 b_0$.

    Asuma para esto que $u \geq v$.

15. Diseñe un circuito digital que tome como entrada dos números binarios

$$a_n a_{n-1} \cdots a_1 a_0 \quad \text{y} \quad b_m b_{m-1} \cdots b_1 b_0 \quad (n, m \geq 0),$$

    y compute como salida el número binario $c_{n+m} c_{n+m-1} \cdots c_1 c_0$ tal que $c_{n+m} c_{n+m-1} \cdots c_1 c_0$ es la representación binaria de $u \cdot v$, donde:

    - u es el entero cuya representación binaria es $a_n a_{n-1} \cdots a_1 a_0$, y
    - v es el entero cuya representación binaria es $b_m b_{m-1} \cdots b_1 b_0$.

16. Sea $P = \{p, q, \dots\}$ un conjunto de proposiciones y sea $\sigma : P \to \{0, 1\}$ una valuación para P. Defina Σ_f como el conjunto de todas las oraciones de la lógica proposicional que utilizan proposiciones en P y cuyo valor de verdad es 1 en σ.

    Demuestre que para cualquier conjunto Σ de oraciones que utilizan proposiciones en P, si $\Sigma_f \subseteq \Sigma$ y Σ es satisfacible, entonces $\Sigma_f = \Sigma$.

17. Recuerde que un conjunto de fórmulas Θ en la lógica proposicional es satisfacible si y solo si existe una valuación hace verdadera a cada fórmula θ en Θ.

    Sea Σ un conjunto de fórmulas en la lógica proposicional. Asuma que el conjunto de variables proposicionales que son mencionadas en Σ es infinito (y, por tanto, Σ es también infinito).

    Decimos que Σ es *finitamente* satisfacible si y solo si todo subconjunto finito de Σ es satisfacible.

    Sea p una variable proposicional cualquiera. Demuestre que si Σ es finitamente satisfacible entonces $\Sigma \cup \{p\}$ o $\Sigma \cup \{\neg p\}$ es finitamente satisfacible (note que es posible que ambos lo sean).

18. La propiedad de *compacidad* de la lógica proposicional establece lo siguiente:

    > Sea Σ un conjunto infinito de oraciones (en la lógica proposicional) sobre conjunto P de variables proposicionales (note que P también puede ser infinito). Entonces Σ es satisfacible si y solo si todo subconjunto finito de Σ es satisfacible.

    Demuestre, utilizando la propiedad de compacidad de la lógica proposicional, que lo siguiente se cumple para cualquier conjunto infinito Σ de oraciones sobre P y oración $\phi \in L(P)$:

    $$\Sigma \models \phi \iff \text{existe subconjunto finito } \Sigma_0 \text{ de } \Sigma \text{ tal que } \Sigma_0 \models \phi.$$

19. Una *cláusula* es una oración de la lógica proposicional de la forma $\ell_1 \vee \cdots \vee \ell_n$, donde cada ℓ_i $(1 \leq i \leq n)$ es un *literal*, es decir, una variable proposicional p o su negación $\neg p$. Por ejemplo, $p \vee \neg q \vee r$ es una cláusula.

    Sea P un conjunto de variables proposicionales. Demuestre que toda oración $\phi \in L(P)$ es equivalente a una conjunción de cláusulas sobre P.

20. Decimos que p es un literal *positivo* y que $\neg p$ es un literal *negativo*. Una *cláusula* es una disyunción de literales. Una cláusula es de *Horn* si y solo si contiene a lo más un literal positivo. Una fórmula es de *Horn* si y solo si es una conjunción de cláusulas de Horn.

    Demuestre que existe una cláusula que no es equivalente a ninguna fórmula de Horn.

21. La *regla de resolución* de la lógica proposicional establece lo siguiente: Si C_1 y C_2 son cláusulas y p es una variable proposicional, entonces desde las cláusulas $(C_1 \vee p)$ y $(C_2 \vee \neg p)$ es posible deducir la cláusula $C_1 \vee C_2$. Por ejemplo, desde las cláusulas $(p \vee \neg q \vee r)$ y $(\neg r \vee s)$ es posible deducir la cláusula $p \vee \neg q \vee s$.

    Demuestre que la regla de resolución es correcta. Esto es, si C_1 y C_2 son cláusulas y p es una variable proposicional, entonces

    $$\{(C_1 \vee p), (C_2 \vee \neg p)\} \models C_1 \vee C_2.$$

22. Una oración de la lógica proposicional está en CNF si es de la forma $\bigwedge_{1 \leq i \leq n} \alpha_i$, donde cada α_i es una cláusula $(1 \leq i \leq n)$.

    Si ℓ es un literal, denotamos por $\bar{\ell}$ el *complemento* de ℓ; esto es, si ℓ es la variable proposicional p entonces $\bar{\ell} = \neg p$, y si $\ell = \neg p$ entonces $\bar{\ell} = p$.

    Sea $\phi = \bigwedge_{1 \leq i \leq n} \alpha_i$ oración proposicional en CNF (es decir, los α_i son cláusulas) y ℓ un literal. Sea I el conjunto de los índices i tal que $1 \leq i \leq n$ y α_i no menciona a ℓ (esto es, ℓ no es un literal en la

disyución que forma la cláusula α_i). Denote como $\phi(\ell)$ la fórmula que se obtiene de $\phi' = \bigwedge_{i \in I} \alpha_i$ al borrar el literal $\bar{\ell}$ de cada cláusula α_i que menciona a $\bar{\ell}$.

Demuestre que si ϕ es insatisfacible entonces $\phi(\ell)$ también lo es.

23. Sean ϕ y ψ fórmulas de la lógica proposicional, y suponga que $V(\phi)$ y $V(\psi)$ denotan el conjunto de variables proposicionales mencionadas en ϕ y ψ, respectivamente.

    Asuma que $\models \phi \to \psi$, i.e. $\phi \to \psi$ es tautología. Demuestre que existe fórmula θ de la lógica proposicional que solo utiliza variables proposicionales en $V(\phi) \cap V(\psi)$, tal que

    $$\models \phi \to \theta \quad \text{y} \quad \models \theta \to \psi.$$

    Explique detalladamente por qué θ cumple esta propiedad.

24. Sea Σ un conjunto de cláusulasde Horn sobre un conjunto finito de variables proposicionales P. Demuestre que si procedimiento Consistencia que se especifica abajo falla entonces Σ es insatisfacible.

    Consistencia

    1. $L_{\text{old}} = P$; $L_{\text{new}} = \emptyset$
    2. **while** $L_{\text{new}} \neq L_{\text{old}}$ **do**
    3. $L_{\text{old}} = L_{\text{new}}$
    4. **if** existe cláusula de Horn $(\neg q_1 \vee \neg q_2 \vee \cdots \vee \neg q_n) \in \Sigma$ tal que
       $\{q_1, \ldots, q_n\} \subseteq L_{\text{old}}$
    5. **then** el procedimiento falla y se detiene
    6. **else**
    7. **if** L es el conjunto de variables $p \in P$ tal que:
       (i) existe cláusula de Horn p en Σ tal que $p \notin L_{\text{old}}$, ó
       (ii) existe cláusula de Horn $(p \vee \neg q_1 \vee \neg q_2 \vee \cdots \vee \neg q_n)$ en Σ tal que
       $\{q_1, \ldots, q_n\} \subseteq L_{\text{old}}$ y $p \notin L_{\text{old}}$
    8. **then** $L_{\text{new}} = L_{\text{old}} \cup L$

25. Sea P un conjunto de variables proposicionales. Un *literal* es una fórmula de la forma p o una de la forma $\neg p$, para $p \in P$. Una *cláusula* es una disyunción de literales. Por ejemplo, $p \vee \neg q \vee r \vee p$ es una cláusula. El *ancho* de una cláusula es el número de literales que aparecen en ella (si aparecen repetidos se cuentan dos veces). La cláusula $p \vee \neg q \vee r \vee p$ tiene ancho cuatro.

    Una fórmula está en CNF si es una conjunción de cláusulas. Por ejemplo, $(p \vee \neg q \vee r) \wedge (s \vee r) \wedge (\neg t \vee r \vee p \vee r)$ es una fórmula en CNF. El ancho de una fórmula en CNF es el máximo ancho de las cláusulas que aparecen en ella. Por ejemplo, $(p \vee \neg q \vee r) \wedge (s \vee r) \wedge (\neg t \vee r \vee p \vee r)$ tiene ancho cuatro. Una fórmula está en kCNF si está en CNF y su ancho es k.

    Demuestre que existe un entero $c \geq 1$ y un procedimiento que realiza a lo más $O(n^c)$ pasos, el cual toma como entrada una fórmula ϕ en CNF con a lo más n símbolos y construye una fórmula ϕ' en 3CNF tal que

    $$\phi \text{ es satisfacible} \quad \Longleftrightarrow \quad \phi' \text{ es satisfacible.}$$

26. Sea R una relación binaria sobre A. Un *cubrimiento* de R es un $S \subseteq A$ tal que para cada par $(a, b) \in R$ se tiene que $a \in S$ o $b \in S$.

    Dada una fórmula ϕ en 3CNF con a lo más n símbolos se le pide construir un conjunto A y una relación binaria R sobre A tal que:

    $$\phi \text{ es satisfacible} \quad \Longleftrightarrow \quad R \text{ tiene un cubrimiento de tamaño } 2n.$$

El costo de todos los pasos efectuados en su construcción debe ser a lo más $O(n^k)$, para un $k \geq 1$ fijo (es decir, que no depende de ϕ).

27. Decimos que dos grafos $G_1 = (N_1, A_1)$ y $G_2 = (N_2, A_2)$ son *isomorfos* si existe una biyección $f : N_1 \to N_2$ tal que para todo a y b en A_1 se tiene que $(a, b) \in A_1$ si y sólo si $(f(a), f(b)) \in A_2$.

    Construya una oración proposicional φ tal que G_1 y G_2 son isomorfos si y sólo si φ es satisfacible. Su fórmula debe ser de tamaño a lo más $O(n^c)$, donde n es el tamaño combinado de G_1 y G_2 y c es una constante.

28. Dada una matriz C de 3×3 que contiene números entre 0 y 3, decimos que C es *completable* si es que existe una manera de reemplazar los números 0 por números entre 1 y 3 de tal forma que la suma de cada fila y de cada columna es la misma. Por ejemplo, la siguiente matriz es completable:

2	0	0
0	2	0
0	0	3

    puesto que podemos reemplazar los valores 0 por los siguientes valores:

2	2	1
2	2	1
1	1	3

    de manera tal que la suma de cada fila y de cada columna es 5. En cambio, la siguiente matriz no es completable:

1	1	1
0	0	0
3	0	0

    Dada una matriz C de 3×3, construya una oración φ en lógica proposicional tal que C es completable si y sólo si φ es satisfacible. En particular, φ tiene que ser construida de tal forma que cada valuación σ que satisface a φ represente una forma de completar C.

29. El *principio de los cajones* establece que si $n + 1$ objetos son distribuidos en n cajones, entonces al menos habrá un cajón con más de un objeto.

    Demuestre el principio para $n = 2$ usando lógica proposicional y la noción de consecuencia lógica.

30. El largo $l(\phi)$ de una fórmula ϕ en la lógica proposicional $\mathcal{L}(p)$ es el número de símbolos que aparecen en la fórmula al leerla de izquierda a derecha. (Asumimos que todos los paréntesis están presentes, incluidos los externos). Por ejemplo, $l(p) = 1$, si $p \in P$, y $l((p \wedge (\neg q))) = 8$, si $p, q \in P$. Del mismo modo, definimos $v(\phi)$ como el número de proposiciones en P que observamos al leer ϕ de izquierda a derecha. Por ejemplo, $v(p) = 1$ y $v((p \vee p)) = 2$, si $p \in P$.

    a) Demuestre que para toda fórmula ϕ en $\mathcal{L}(P)$ que no utiliza el conectivo $\neg$ se cumple que:

    $$l(\phi) \ \leq \ 3 \cdot v(\phi)^2.$$

    b) Lo anterior dice que el largo de una fórmula sin negación está acotado polinomialmente por el número de ocurrencias de proposiciones en ella. Demuestre que esto no se sigue cumpliendo si ahora permitimos utilizar la negación $\neg$.

    c) Encuentre una restricción sintáctica del lenguaje $\mathcal{L}(P)$ que sea funcionalmente completa (es decir, para cada fórmula ϕ en $\mathcal{L}(P)$ existe una fórmula ϕ' en el lenguaje restringido tal que $\phi \equiv \phi'$), y para la cual se cumpla la propiedad de que el largo de una fórmula está acotado polinomialmente por el número de ocurrencias de proposiciones en ella. Es decir, existe $k \geq 1$ tal que para toda fórmula ϕ en el lenguaje restringido se cumple que $l(\phi)$ es $O(v(\phi)^k)$.

1.4. Lógica Relacional

Como su nombre lo dice, la lógica relacional permite expresar propiedades de las relaciones. Análogamente al caso de la lógica proposicional, en el que inicialmente se determina un conjunto P de proposiciones las cuales el lenguaje $\mathcal{L}(P)$ puede nombrar, tenemos ahora un vocabulario $\nu = \{R, S, T, \dots\}$ compuesto por *símbolos de relación*. Cada uno de estos símbolos de relación tiene asociada una aridad, que es el número de argumentos que tal símbolo de relación utiliza. Es importante recalcar que los símbolos de relación se utilizan para nombrar relaciones, pero no corresponden en sí mismos a relaciones. Al momento de darle semántica al lenguaje deberemos interpretar tales símbolos como relaciones (de la aridad apropiada) sobre un conjunto A dado (el cual corresponde al dominio de discurso).

1.4.1. Ejemplos

No es sencillo asimilar la definición de la lógica relacional sin antes haber visto ejemplos. Aquí damos varios de ellos. Es importante entender intuitivamente, sin embargo, qué es lo que la lógica relacional puede expresar. De partida, tenemos a disposición un conjunto (infinito, contable) de variables $x, y, z, \dots$ que podemos nombrar. Es posible verificar si dos variables representan el mismo objeto en nuestro dominio de discurso A por medio de la fórmula $x = y$. También se permite verificar, dado un símbolo de relación R en el vocabulario ν de aridad $n > 0$, si una tupla $(x_1, \dots, x_n)$ de variables pertenece a la interpretación de R (tal interpretación, como dijimos antes, corresponde a una relación sobre el dominio A). Esto se expresa mediante la fórmula $R(x_1, \dots, x_n)$. El conjunto de fórmulas es, además, cerrado bajo la aplicación de todos los conectivos lógicos que vimos antes; es decir, $\{\neg, \vee, \wedge, \rightarrow\}$. Finalmente, podemos decir que un elemento x del dominio satisface una propiedad, mediante el cuantificador existencial $\exists x$, y que todo elemento satisface la propiedad, mediante el cuantificador universal $\forall x$.

Antes de ver los ejemplos es importante recalcar dos cosas. Primero, la lógica relacional solo permite aplicar símbolos de relación a variables, mediante fórmulas de la forma $R(x_1, \dots, x_n)$. Es decir, no es posible anidar símbolos de relación dentro de otros (por ejemplo, como en la fórmula $R(S, x, \dots)$). Lo segundo es que la lógica relacional también se conoce como de *primer orden*. La razón es que solo se permite aplicar los cuantificadores $\exists$ y $\forall$ sobre variables, las cuales representan elementos u objetos dentro de nuestro dominio de discurso; es decir, su *primer nivel*. Existe también una lógica de *segundo orden*, que permite cuantificar sobre relaciones en el dominio (es decir, permite expresar que existe una relación R que tiene cierta propiedad), pero no estudiaremos esa lógica en este curso.

Ejemplo 1.4.1. Sea R un símbolo de relación binario. Podemos expresar que R es refleja, simétrica, antisimétrica, o transitiva, mediante las siguientes fórmulas de la lógica relacional sobre el vocabulario $\sigma = \{R\}$:

- **Reflexividad:** $\forall x R(x, x)$.
- **Simetría:** $\forall x \forall y (R(x, y) \rightarrow R(y, x))$.
- **Antisimetría:** $\forall x \forall y \big((R(x, y) \wedge R(y, x)) \rightarrow x = y\big)$.
- **Transitividad:** $\forall x \forall y \forall z \big((R(x, y) \wedge R(y, z)) \rightarrow R(x, z)\big)$.

También podemos representar que R "codifica" una función mediante la siguiente fórmula:

$$\forall x \big(\exists y R(x, y) \wedge \neg \exists y_1 \exists y_2 (\neg(y_1 = y_2) \wedge R(x, y_1) \wedge R(x, y_2))\big).$$

El hecho de que para todo elemento x se cumpla que $\exists y R(x, y)$ representa que todo elemento tiene al menos una imagen en R, mientras que el hecho de que se cumpla que $\neg \exists y_1 \exists y_2 (\neg(y_1 = y_2) \wedge R(x, y_1) \wedge R(x, y_2))$ representa que esa imagen es única. □

Para simplificar la notación de ahora en adelante escribimos $x \neq y$ en vez de $\neg(x = y)$.

Pregunta. *¿Es posible expresar que la clausura transitiva de la relación contiene todos los pares? ¿Que la relación es finita? Y aún sabiendo esto último, ¿que su cardinalidad es par?*

Ejemplo 1.4.2. Consideremos un vocabulario compuesto por un solo símbolo de relación ternario S. Podemos expresar que S representa una función binaria sobre el dominio de discurso A por medio de la fórmula:

$$\forall x \forall y \big(\exists z S(x,y,z) \ \wedge \ \neg \exists z_1 \exists z_2 ((z_1 \neq z_2) \wedge S(x,y,z_1) \wedge S(x,y,z_2)) \big).$$

Es decir, cada par (x, y) de elementos en el dominio de discurso tiene una, y solo una, imagen z bajo S.

Suponga que f es la función binaria definida por S. Es decir, $f(x, y) = z$ ssi la tupla (x, y, z) pertenece a la interpretación de S. Podemos entonces avanzar y verificar si f representa la operación binaria que define un grupo sobre el dominio de discurso. Para ello debemos verificar que f es asociativa, que existe un "neutro" en el dominio, y que todo elemento tiene un "inverso" bajo f.

- **Asociatividad:** Para esto necesitamos expresar que:

  $$f(x, f(y, z)) \ = \ f(f(x, y), z),$$

  para todos los elementos x, y, z en el dominio. Recuerde, sin embargo, que no podemos expresar nada sobre la función f directamente, sino tan solo sobre la relación S que la codifica. Una forma de expresar esto es mediante la fórmula:

  $$\forall x \forall y \forall z \forall u \forall v \forall w \big(S(y,z,u) \wedge S(x,u,v) \wedge S(x,y,w) \ \rightarrow \ S(w,z,v) \big).$$

  (¿Qué pasaría si reemplazáramos el conectivo $\rightarrow$ en esta fórmula por otra conjunción $\wedge$? ¿Seguiría siendo una forma válida de expresar esta propiedad?)

- **Existencia de neutro:** Decimos que el elemento x es un neutro si satisface la fórmula:

  $$\text{Neutro}(x) \ := \ \forall y (S(x,y,y) \wedge S(y,x,y)).$$

  Note que en esta fórmula x aparece como *variable libre*; es decir, no cuantificada. Tal fórmula es satisfecha por todos los elementos x del dominio que son "neutros". Para expresar la existencia de un neutro entonces basta escribir: $\exists x \text{Neutro}(x)$. (¿Cómo expresaría que tal neutro es único?)

- **Existencia de inversos:** Debería ser claro a esta altura que basta expresar lo siguiente:

  $$\exists x (\text{Neutro}(x) \ \wedge \ \forall y \exists z (S(y,z,x) \wedge S(z,y,x)).$$

□

Observación. *Es importante recalcar que las menciones al dominio de discurso A en la lógica relacional son siempre implícitas. Es decir, no podemos expresar en el lenguaje fórmulas del tipo $\exists x \in A$ o $\forall x \in A$, ya que la sintaxis de la lógica no puede referirse explícitamente a su semántica (el dominio donde será interpretada). Cada vez que escribimos $\exists x$ (resp., $\forall x$) estamos expresando implícitamente que existe un elemento en el dominio que satisface algo (resp., todo elemento en el dominio satisface algo). La fórmula podrá luego ser interpretada sobre múltiples dominios distintos; en algunos de ellos será verdadera, y en otros falsa.*

Ejemplo 1.4.3. En este caso nuestro vocabulario consta de dos símbolos de relación ternarios S y P. Suponemos que el dominio es $\mathbb{N}$. El símbolo S se interpreta como la suma, es decir, como todas las tuplas (x, y, z) tal que $x + y = z$, mientras que P se interpreta como el producto, es decir, como las tuplas (x, y, z) tal que $x \cdot y = z$. Podemos entonces definir el cero como:

$$\text{Cero}(x) \ := \ \forall y S(x,y,y).$$

Esta fórmula tiene una variable libre x, y define en $\mathbb{N}$ la relación unaria de todos los "ceros". (En este caso específico solo hay un cero, pero eso es irrelevante para nuestro análisis.) Podemos ahora definir la relación binaria "menor" como sigue:

$$\text{Menor}(x,y) \ := \ \exists z (\neg \text{Cero}(z) \ \wedge \ S(x,z,y)).$$

Esta fórmula tiene dos variables libres, x e y, y define en $\mathbb{N}$ la relación binaria de todos los pares (x, y) de naturales tal que $x < y$. Con esta relación podemos entonces definir la relación "sucesor" como sigue:

$$\text{Sucesor}(x, y) := \text{Menor}(x, y) \wedge \neg\exists z(\text{Menor}(x, z) \wedge \text{Menor}(z, y)).$$

Por tanto, el "uno" es aquel elemento definido por la fórmula:

$$\text{Uno}(x) = \forall y(\text{Cero}(y) \rightarrow \text{Sucesor}(y, x)).$$

Para definir el conjunto de los primos utilizamos entonces la fórmula:

$$\text{Primo}(x) := \forall y \forall z\big(P(y, z, x) \rightarrow (\text{Uno}(y) \vee \text{Uno}(z))\big).$$

Si queremos decir que los primos son infinitos basta escribir:

$$\forall x \exists y(\text{Menor}(x, y) \wedge \text{Primo}(y)).$$

□

Ejemplo 1.4.4. Suponga que tenemos una base de datos que almacena información cinematográfica. Esta base de datos consta de tres tablas ternarias. La primera tabla es `Peliculas`, y almacena tuplas de la forma (t, d, a) tal que t es una película dirigida por d y donde actúa a. La segunda tabla es `Direccion`. Esta contiene todas las tuplas (c, r, p) tal que c es un cine localizado en la dirección r y cuyo teléfono es p. Por último, tenemos una tabla llamada `Cartelera`, la que contiene las tuplas (c, t, h) tal que c es un cine que está dando la película t en el horario h.

Para hallar el conjunto de las películas donde el director también actúa podemos utilizar la fórmula:

$$\exists y \texttt{Pelicula}(x, y, y),$$

la cual tiene como variable libre a x. Para extraer las películas con un solo actor utilizamos la fórmula:

$$\exists y \exists z\big(\texttt{Pelicula}(x, y, z) \wedge \neg\exists z'(z \neq z' \wedge \texttt{Pelicula}(x, y, z'))\big).$$

Para acceder a las películas en cartelera junto con su director y el cine donde se están proyectando, usamos la fórmula:

$$\exists z \exists u\big(\texttt{Pelicula}(x, y, z) \wedge \texttt{Cartelera}(v, x, u)\big),$$

la que tiene como variables libres a x, y, v. Las películas en cartelera, pero para las cuales no hay información acerca de la dirección de ninguno de los cines donde se muestra se puede expresar como:

$$\exists y \exists z\big(\texttt{Cartelera}(y, x, z) \wedge \forall y' \forall z'(\texttt{Cartelera}(y', x, z') \rightarrow \neg\exists u \exists v(\texttt{Direccion}(y', u, v)))\big).$$

□

1.4.2. Sintaxis

Ahora definimos la sintaxis de la lógica relacional siguiendo las observaciones previas. Sea ν un vocabulario, es decir, un conjunto de símbolos de relación, cada uno de los cuales se halla asociado con una aridad $n > 0$. El lenguage $\mathcal{L}(\nu)$ de la lógica relacional sobre ν se define inductivamente como sigue:

- **Casos base:**
  1. Si x, y son variables, entonces $x = y$ pertenece a $\mathcal{L}(\nu)$.
  2. Si R es un símbolo de relación en ν de aridad $n > 0$ y $x_1, \ldots, x_n$ es una tupla de variables (no necesariamente distintas), entonces $R(x_1, \ldots, x_n)$ pertenece a $\mathcal{L}(\nu)$.
- **Casos inductivos:** Si $\phi, \psi \in \mathcal{L}(\nu)$, entonces:
  1. $\neg\phi$, $\phi \vee \psi$, $\phi \wedge \psi$, y $\phi \rightarrow \psi$ están en $\mathcal{L}(\nu)$.
  2. Si x es una variable, entonces $\exists x\phi$ y $\forall x\phi$ están en $\mathcal{L}(\nu)$.

1.4.3. Semántica

Para interpretar una fórmula $\phi \in \mathcal{L}(\nu)$ utilizamos el concepto de *estructura* $\mathcal{A}$. Tal estructura consta de un *dominio* A no vacío – aquel donde tomarán valor las variables mencionadas en ϕ – además de una *interpretación* de cada símbolo de relación R en el vocabulario ν. Asuma que $\sigma = \{R_1, \ldots, R_m\}$ y que R_i es de aridad $n_i > 0$, para cada $1 \leq i \leq m$. Entonces tal interpretación asocia con cada R_i una relación en A de aridad n_i (es decir, un subconjunto de A^{n_i}). Denotamos tal relación por $R_i^{\mathcal{A}}$. Esta estructura puede entonces verse como la tupla:

$$\mathcal{A} \quad := \quad (A;\, R_1^{\mathcal{A}}, \ldots, R_m^{\mathcal{A}}).$$

Note que las estructuras sirven el mismo propósito que las valuaciones en la lógica proposicional: representan las diferentes formas de interpretar nuestro vocabulario (las relaciones en el caso de las primeras, las proposiciones en el caso de las segundas).

Si una fórmula no tiene variables libres podemos determinar si es cierta o falsa en $\mathcal{A}$ utilizando la noción intuitiva que tenemos de satisfacción de una fórmula (desarrollada en los ejercicios anteriores). Si ϕ es cierta en $\mathcal{A}$, lo que comúnmente se expresa como que $\mathcal{A}$ *satisface* ϕ, escribimos $\mathcal{A} \models \phi$. Por otro lado, si la fórmula tiene variables libres, entonces al evaluarla sobre una estructura no obtenemos un valor de verdad (es decir, la fórmula no es simplemente cierta o falsa). Como muestran nuestros ejemplos, en tales casos nuestra fórmula define una relación (de la aridad adecuada) sobre el dominio de la estructura. En particular, si la fórmula tiene n variables libres, entonces define en la estructura $\mathcal{A}$ una relación de aridad n sobre su dominio A. Esta relación contiene precisamente las *valuaciones* de las variables libres de la fórmula que hacen a la fórmula verdadera.

Formalmente, la semántica de lógica relacional se define como sigue. Sea $\mathcal{A} = (A;\, R_1^{\mathcal{A}}, \ldots, R_m^{\mathcal{A}})$ una estructura sobre el vocabulario ν y σ una valuación del conjunto de variables sobre $\mathcal{A}$; es decir, σ asigna un valor en el dominio A a cada variable x. Entonces definimos la noción de satisfacción de una fórmula de la lógica relacional con respecto al par $(\mathcal{A}, \sigma)$ como sigue:

- $(A, \sigma) \models x = y$ ssi $\sigma(x) = \sigma(y)$.
- $(\mathcal{A}, \sigma) \models R_i(x_1, \ldots, x_n)$ ssi $(\sigma(x_1), \ldots, \sigma(x_n)) \in R_i^{\mathcal{A}}$.
- $(\mathcal{A}, \sigma) \models \neg\phi$ ssi no es cierto que $(\mathcal{A}, \sigma) \models \phi$.
- $(\mathcal{A}, \sigma) \models \phi \vee \psi$ ssi $(\mathcal{A}, \sigma) \models \phi$ o $(\mathcal{A}, \sigma) \models \psi$.
- $(\mathcal{A}, \sigma) \models \exists x\phi$ ssi existe $a \in A$ tal que $(\mathcal{A}, \sigma[x/a]) \models \phi$, donde $\sigma[x/a]$ es la valuación que toma valor a en la variable x y coincide con σ en todas las otras variables.
- $(\mathcal{A}, \sigma) \models \forall x\phi$ ssi para todo $a \in A$ se tiene que $(\mathcal{A}, \sigma[x/a]) \models \phi$.

Si $(x_1, \ldots, x_n)$ es la tupla que contiene a todas las variables libres de una fórmula ϕ, entonces la relación definida por ϕ en $\mathcal{A}$ corresponde precisamente al conjunto de tuplas $(\sigma(x_1), \ldots, \sigma(x_n))$ tal que $(\mathcal{A}, \sigma) \models \phi$.

1.4.4. Equivalencia

Dos fórmulas $\phi, \psi \in \mathcal{L}(\nu)$ sin variables libres son equivalentes, escrito como antes $\phi \equiv \psi$, si en cada estructura $\mathcal{A}$ se cumple que:

$$\mathcal{A} \models \phi \quad \iff \quad \mathcal{A} \models \psi.$$

Una equivalencia particularmente interesante es la siguiente:

$$\exists x\phi \;\equiv\; \neg\forall x\neg\phi.$$

De hecho, decir que un elemento tiene la propiedad ϕ es lo mismo que decir que ninguno de sus elementos satisface $\neg\phi$. Esta equivalencia nos muestra que cualquiera de los dos cuantificadores puede eliminarse del lenguaje sin perder poder expresivo (aunque es conveniente contar con ellos para poder expresar algunas propiedades de una forma más natural).

Pregunta. *¿Es cierto que* $\exists x(P(x) \wedge Q(x)) \equiv \exists x P(x) \wedge \exists x Q(x)$*? ¿Es cierto que* $\exists x(P(x) \vee Q(x)) \equiv \exists x P(x) \vee \exists x Q(x)$*?*

Pregunta. *¿Puede diseñar un procedimiento que determine si dos fórmulas son equivalentes?*

1.4.5. Consecuencia lógica

Como antes escribimos $\Delta \models \phi$ para representar que ϕ es consecuencia lógica de Δ. Esto quiere decir que para toda estructura $\mathcal{A}$, si $\mathcal{A} \models \Delta$ entonces $\mathcal{A} \models \phi$. (Aquí, $\mathcal{A} \models \Delta$ representa que $\mathcal{A} \models \alpha$ para todo $\alpha \in \Delta$).

Pregunta. *¿Es cierto que* $\{\exists x \forall y P(x,y)\} \models \forall y \exists x P(x,y)$*? ¿Es cierto lo opuesto, es decir que* $\{\forall y \exists x P(x,y)\} \models \exists x \forall y P(x,y)$*?*

Pregunta. *¿Puede diseñar un procedimiento que determine si* $\Delta \models \phi$*?*

Ejercicios Finales

1. Demuestre que las siguientes fórmulas son equivalentes:

$$(1)\ \ \exists x(A(x) \to B(x)) \qquad (2)\ \ \forall x A(x) \to \exists x B(x)$$

2. Demuestre que existe un dominio de discurso A y una interpretación de la relación binaria P sobre A que satisface la siguiente fórmula:

$$\forall x \exists y P(x,y) \wedge \forall x \forall y_1 \forall y_2 (P(x,y_1) \wedge P(x,y_2) \to y_1 = y_2) \wedge \forall x_1 \forall x_2 \forall y (P(x_1,y) \wedge P(x_2,y) \to x_1 = x_2) \wedge \neg \forall y \exists x P(x,y).$$

   ¿Es posible que A sea un conjunto finito?

3. Sea $E(x,y)$ un predicado binario utilizado para representar la noción de adyacencia en grafos. En cada una de las siguientes preguntas escriba una oración de la lógica relacional que represente la propiedad mencionada.

   (a) El grafo es un clique.
   (b) El grafo contiene un clique con 4 nodos.
   (c) El grafo tiene un ciclo con 4 nodos.
   (d) Existen elementos en el grafo cuya distancia es 4.
   (e) La distancia máxima entre dos nodos del grafo es 3.
   (f) El grafo contiene exactamente 6 nodos.

4. ¿Cuáles de las siguientes afirmaciones son válidas? Justifique su respuesta.

   (a) $\forall x\, \varphi \ \equiv\ \neg \exists x \neg \varphi$.
   (b) $\exists x\, \varphi \ \equiv\ \neg \forall x \neg \varphi$.
   (c) $\exists x\, (\varphi \vee \psi) \ \equiv\ (\exists x\, \varphi) \vee (\exists x\, \psi)$.
   (d) $\forall x\, (\varphi \vee \psi) \ \equiv\ (\forall x\, \varphi) \vee (\forall x\, \psi)$.
   (e) $\exists x\, (\varphi \wedge \psi) \ \equiv\ (\exists x\, \varphi) \wedge (\exists x\, \psi)$.
   (f) $\forall x\, (\varphi \wedge \psi) \ \equiv\ (\forall x\, \varphi) \wedge (\forall x\, \psi)$.

5. Demuestre que las siguientes oraciones de la lógica relacional son equivalentes: $\forall x P(x) \wedge \exists x Q(x)$ y $\forall x \exists y (P(x) \wedge Q(y))$.

6. Demuestre que las siguientes fórmulas son equivalentes:

$$(1)\ \ \exists x(A(x) \to B(x)) \qquad (2)\ \ \forall x A(x) \to \exists x B(x)$$

7. Considere como dominio de discurso un conjunto P de seres humanos, y sean *Amante* y *Jefe* dos predicados binarios interpretados sobre P como el conjunto de pares (p_1, p_2) tal que p_1 es el jefe de p_2 y p_1 es el amante de p_2, respectivamente.

   Represente, usando solo los predicados binarios mostrados anteriormente, las siguientes propiedades del dominio de discurso P usando lógica de primer orden:

   a) La relación *Amante* es conmutativa (o simétrica).

   b) La relación *Jefe* es transitiva (i.e. el jefe de mi jefe es también mi jefe) pero no conmutativa (i.e. nadie es jefe de su propio jefe).

   c) Nadie es jefe de su amante.

   d) Ningún amante tiene más de dos jefes.

   e) Si dos personas tienen los mismos jefes entonces tienen los mismos amantes.

   f) Existe una persona que es jefe y tal que cada uno de sus empleados es amante de alguien.

8. Considere el dominio de discurso que contiene los primeros n números naturales $\{1, \ldots, n\}$, y asuma que S es la relación binaria de sucesor sobre estos números (es decir, S contiene los pares $(j, j+1)$, para $1 \leq j < n$).

   Asuma que existen otras dos relaciones binarias R_a y R_b sobre este dominio. Construya una fórmula de la lógica de predicados que exprese que todas las siguientes condiciones se cumplen:

   a) La unión de las relaciones R_a y R_b corresponde a S.

   b) La intersección entre R_a y R_b es vacía.

   c) $R_a = \{(j, j+1) \mid 1 \leq j < n \text{ y } j \text{ es impar}\}$.
   Hint: Exprese que el par $(1, 2)$ pertenece a R_a y que si un par $(i, i+1)$ pertenece a R_a (resp. R_b) entonces $(i+1, i+2)$ pertenece a R_b (resp. R_a).

   La fórmula que usted construya solo puede utilizar los predicados S, R_a y R_b.

   ¿Es posible detectar con una fórmula en la lógica de predicados si n es par en el caso que (1-3) se cumpla?

9. Sea $x \subseteq y$ un predicado binario. Intuitivamente $\subseteq$ representa la relación de subconjunto; es decir, $x \subseteq y$ representa que x es un subconjunto de y.

   Exprese en lógica relacional las siguientes propiedades del predicado $x \subseteq y$:

   - La relación $x \subseteq y$ es un orden parcial; i.e. es refleja, antisimétrica y transitiva.
   - Existe un único elemento $\bot$ que está contenido en cualquier otro conjunto (i.e. el conjunto vacío).
   - Existe un único conjunto $\top$ que contiene a todo otro conjunto (es decir, el conjunto universo).
   - La unión de dos conjuntos siempre existe, y además es única (note que $x \cup y = z$ si y sólo si z es el "menor" conjunto con respecto al orden parcial $\subseteq$ que contiene tanto a x como a y).
   - La intersección de dos conjuntos siempre existe, y además es única (note que $x \cap y = z$ si y sólo si z es el "mayor" conjunto con respecto al orden parcial $\subseteq$ que está contenida tanto en x como en y).
   - Todo conjunto tiene un *complemento*; es decir, para todo conjunto x existe un conjunto $\bar{x}$ tal que $x \cap \bar{x} = \bot$ y $x \cup \bar{x} = \top$.

10. Asuma que el dominio de discurso son los números naturales, y que contamos con (a) un predicado binario $<$ que es intepretado como el orden lineal estándar en $\mathbb{N}$, y (b) dos predicados ternarios $\cdot$ y $+$ que definen a la multiplicación y suma en $\mathbb{N}$, respectivamente.

    Exprese en lógica de primer orden las siguientes propiedades de los números naturales usando solo los predicados mencionados en el párrafo anterior:

    - Todo número natural positivo es par o impar, pero no ambos.
    - El sucesor de todo número par es impar.
    - Existe un número infinito de números primos.
    - Para todo par (n, n') de números naturales positivos, existe un único par (p, c) tal que $p \geq 0$, $0 \leq c \leq n-1$ y $n' = pn + c$.

11. Considere un dominio formado por líneas y puntos. Estos se hallan en una relación de incidencia I tal que $(a, b) \in I$ si y solo si a es incidente a b.

    Las líneas son aquellos elementos para los cuales podemos encontrar un elemento (un punto) que sea incidente a ellas. Los puntos son aquellos elementos que son incidentes a algún otro elemento (una línea). Dos líneas se intersectan si tienen un punto incidente en común.

    Exprese las siguientes propiedades en lógica de predicados utilizando solo el predicado binario I (recuerde que puede utilizar la igualdad $=$ entre variables):

    a) Todo elemento del dominio es un punto o una línea, pero no ambos.

    b) Todo par de líneas se intersectan en un punto.

    c) Si las líneas son distintas entonces se intersectan en exactamente un punto.

12. Sea σ el vocabulario que contiene una sola relación ternaria $B(\cdot, \cdot, \cdot)$. Considere como dominio D al conjunto de todos los puntos en $\mathbb{R} \times \mathbb{R}$, y asuma que interpretamos a la relación B en D como *estar entre medio*; formalmente esto quiere decir que interpretamos a B como el subconjunto de $D \times D \times D$ que contiene todos aquellos triples (x, y, z) tales que y pertenece a la recta que une x con z. Exprese lo siguiente en la lógica de primer orden:

    a) Para todo punto x, el único punto que está entre medio de x y x es x mismo.

    b) Existen tres puntos que no son colineales.

    c) Para cualquiera tres puntos no colineales x, y, z, cualquier punto u en el segmento xy y cualquier punto v en el segmento yz, los segmentos xv y zu deben intersectarse.

    d) Asuma que nuestro vocabulario se extiende con una relación C de aridad cuatro que se interpreta en D como la *congruencia*; es decir, como todas aquellas tuplas $(x, y, w, z) \in D \times D \times D \times D$ tal que los segmentos xy y wz son del mismo largo.

    Exprese lo siguiente: Sean x, y, z puntos en D. Entonces x, y, z son colineales si existe un par (u, v) de puntos distintos tal que tanto x como y como z son equidistantes de u y v.

13. Suponga que tiene una relación binaria R interpretada como un orden lineal. Sea $C = \{c_1, \ldots, c_k\}$ un conjunto finito de colores, y H, V dos relaciones en C llamadas *restricciones horizontales* y *verticales*, respectivamente. Intentaremos determinar si el plano $\mathbb{Z} \times \mathbb{Z}$ puede ser coloreado con colores en C respetando las restricciones horizontales y verticales. Esto significa que existe una función $f : \mathbb{Z} \times \mathbb{Z} \to C$ tal que para todo $(i, j) \in \mathbb{Z} \times \mathbb{Z}$ se cumple que:

$$\big(f(i,j), f(i+1,j)\big) \in H \quad \text{y} \quad \big(f(i,j), f(i,j+1)\big) \in V.$$

    Para esto extenderemos el vocabulario que contiene a la relación binaria R con las relaciones binarias P_ℓ, para $1 \leq \ell \leq k$. Intuitivamente, $P_\ell(i, j)$ es cierto si y solo si la coloración del plano le asigna color ℓ a la posición $(i, j) \in \mathbb{Z} \times \mathbb{Z}$.

Escriba una fórmula que exprese lo siguiente: (1) Toda posición recibe un, y solo un, color en el conjunto C, y (2) el color que reciben las posiciones respeta las restricciones H y V, respectivamente. Para ello puede utilizar la relación R definida en *(a)*, asumiendo que su interpretación cumple las condiciones ahí descritas.

14. Una fórmula de la lógica de predicados está en *forma normal* si es de la forma $Q_1x_1Q_2x_2 \dots Q_mx_m\varphi$, donde cada Q_i es un cuantificador $\forall$ o $\exists$ $(1 \leq i \leq m)$ y φ es una fórmula sin cuantificación. Por ejemplo, $\forall x \exists y P(x, y)$ está en forma normal, pero $\forall x P(x) \wedge \exists y R(y)$ no lo está.

    Sean ϕ, θ fórmulas en forma normal. Demuestre que existe α en forma normal tal que $\alpha \equiv \phi \wedge \theta$.

Capítulo 2

La Noción de Inducción

El principio de inducción es una herramienta fundamental para demostrar propiedades acerca de todos los números naturales. Intuitivamente, si queremos demostrar que una propiedad P es satisfecha por todos los elementos en $\mathbb{N}$, entonces basta demostrar que la cumple el primer elemento (el 0 o el 1, según corresponda), y además que cada vez que un número natural arbitrario n satisface la propiedad P también se tiene que $(n+1)$ satisface P. Formalmente, podemos expresar el principio de inducción en la lógica relacional como sigue:

$$\big(\, P(0) \,\wedge\, \forall n(P(n) \to P(n+1))\,\big) \quad \to \quad \forall k P(k).$$

Más que ser capaces de demostrar propiedades por inducción en este capítulo – la que es una habilidad que ya dominan – nos enfocaremos en entender mejor qué es el principio de inducción, cómo se relaciona con otros principios fundamentales de los números naturales (por ejemplo, el del buen orden), y cómo es posible utilizar la inducción en problemas fundamentales relacionados con la ciencia de la computación; e.g., para realizar definiciones recursivas de objetos, para diseñar algoritmos recursivos, y para demostrar inductivamente la correctitud de estos.

2.1. El principio de inducción

Comenzaremos con un simple ejemplo que muestra el poder del principio de inducción para demostrar propiedades interesantes de los números naturales.

Ejemplo 2.1.1. Demostraremos por inducción la *desigualdad de Bernoulli*. Esta dice que si $h > -1$, entonces para todo $n \geq 0$ se cumple que:

$$(1+nh) \,\leq\, (1+h)^n.$$

Por inducción en $n \geq 0$:

- **Caso base:** Para $n = 0$ tenemos que $(1+nh) = 1 = (1+h)^n$.
- **Caso inductivo:** Asumimos por hipótesis inductiva (HI) que la propiedad se cumple para $n \geq 0$. Luego:

$$\begin{aligned}
(1+h)^{n+1} &= (1+h)^n(1+h) \\
&\geq (1+nh)(1+h) \qquad \text{(por HI y } (1+h) > 0) \\
&= 1 + nh + h + nh^2 \\
&= 1 + (n+1)h + nh^2 \\
&\geq 1 + (n+1)h.
\end{aligned}$$

Esto finaliza la demostración. □

Ejercicio. *Un número impar $n \geq 3$ de personas se para en un parque a distancias mutuas distintas. Cada persona le lanza una torta a la persona que tiene más cerca. Demuestre por inducción que existe una persona a la que nadie le lanza una torta.*

2.1.1. La inducción fuerte

En algunos casos es conveniente utilizar una versión "reforzada" del principio de inducción para demostrar una propiedad P acerca los elementos de $\mathbb{N}$. Esta versión se conoce como *principio de inducción fuerte*, y puede formalizarse de la siguiente forma:

$$\big(P(0) \wedge \forall n(\bigwedge_{0\leq j\leq n} P(j) \to P(n+1))\big) \quad \to \quad \forall k P(k).$$

Es decir, el principio de inducción fuerte nos provee más versatilidad al intentar demostrar el caso inductivo, ya que podemos asumir no solo que n tiene la propiedad P, sino que todo elemento j que es menor o igual a n. A continuación veremos ejemplos que muestran la conveniencia del principio de inducción fuerte.

Ejemplo 2.1.2. Demostraremos por inducción fuerte que todo número natural $n \geq 2$ puede ser escrito como un producto de primos.

- **Caso base:** $n = 2$ es primo.
- **Caso inductivo:** Si $n+1$ es primo, no hay nada que demostrar. Suponga, por el contrario, que $n+1 = jk$, para $2 \leq j, k \leq n$. Por HI fuerte tanto j como k pueden ser escritos como el producto de primos, por lo que también $(n+1)$ puede ser escrito como un producto de primos.

Esto finaliza la demostración. □

Ejemplo 2.1.3. Sea $n \geq 1$. Demuestre que si $k = 2^n$, entonces para toda secuencia $a_1, a_2, \ldots, a_k$ de números reales positivos se tiene que su media aritmética es mayor o igual a su media geométrica; es decir:

$$\frac{a_1 + a_2 + \cdots + a_k}{k} \;\geq\; \sqrt[k]{a_1 \cdot a_2 \cdot \cdots \cdot a_k}.$$

La demostración es por inducción en $n \geq 1$:

- **Caso base:** Para $n = 1$ tenemos que $k = 2$. Sean a_1, a_2 reales positivos. Entonces:

$$\frac{a_1 + a_2}{2} \;\geq\; \sqrt{a_1 \cdot a_2} \quad \iff \quad (\frac{a_1 + a_2}{2})^2 \;\geq\; a_1 \cdot a_2 \quad \iff \quad (a_1 - a_2)^2 \;\geq\; 0.$$

  Esta última desigualdad es claramente cierta.

- **Caso inductivo:** Asumimos por HI fuerte que la propiedad se cumple para $1 \leq j \leq n$. Demostramos entonces para $(n+1)$. Sea $k = 2^{n+1} = 2 \cdot 2^n$ y $a_1, \ldots, a_k$ reales positivos. Luego:

$$\begin{aligned}
\frac{a_1 + a_2 + \cdots + a_k}{k} &= \frac{a_1 + \cdots + a_{2^n} + a_{2^n+1} + \cdots + a_{2^{n+1}}}{2^{n+1}} \\
&= \frac{1}{2} \cdot \big(\frac{a_1 + \cdots + a_{2^n}}{2^n} + \frac{a_{2^n+1} + \cdots + a_{2^{n+1}}}{2^n}\big) \\
&\geq \frac{1}{2} \cdot \big(\sqrt[2^n]{a_1 \cdot \cdots \cdot a_{2^n}} + \sqrt[2^n]{a_{2^n+1} \cdot \cdots \cdot a_{2^{n+1}}}\big) \\
&\geq \sqrt{\sqrt[2^n]{a_1 \cdot \cdots \cdot a_{2^n}} \cdot \sqrt[2^n]{a_{2^n+1} \cdot \cdots \cdot a_{2^{n+1}}}} \\
&= \sqrt[2^{n+1}]{a_1 \cdot \cdots \cdot a_{2^n} a_{2^n+1} \cdot \cdots \cdot a_{2^{n+1}}} \\
&= \sqrt[k]{a_1 \cdot \cdots \cdot a_k}.
\end{aligned}$$

  Note que la primera desigualdad se obtiene por HI en n, mientras que la segunda se obtiene por HI fuerte en el caso base.

Esto finaliza la demostración. □

2.1.2. El poder de la inducción fuerte

¿Nos da la inducción fuerte más poder para demostrar que la inducción normal? A primera vista pareciera ser que sí, pero demostraremos que esto no es el caso. Partamos con un ejemplo.

Ejemplo 2.1.4. Demostraremos por inducción (no fuerte) que todo número natural $n \geq 2$ puede ser escrito como un producto de primos. Para esto necesitamos "sobrecargar" la HI. En particular, demostraremos lo siguiente por inducción para todo $n \geq 2$: Para todo $2 \leq j \leq n$ se cumple que j puede ser escrito como el producto de primos. La demostración es como sigue:

- **Caso base:** Tenemos que $n = 2$, por lo que todo $2 \leq j \leq n$ puede ser escrito como producto de primos.
- **Caso inductivo:** Considere un j arbitrario tal que $2 \leq j \leq n+1$. Si $2 \leq j \leq n$, entonces por HI (normal) obtenemos que j puede ser escrito como un producto de primos. Si $j = n+1$ y $n+1$ es primo, entonces la propiedad se cumple trivialmente. Si $j = n+1$ y $n+1 = kp$, para $2 \leq k, p \leq n$, entonces por HI (normal) tanto k como p pueden ser escritos como el producto de primos, por lo que también $(n+1)$ puede ser escrito como un producto de primos.

Esto finaliza la demostración. □

Abstrayendo de esta idea, obtenemos lo siguiente:

Proposición 2.1.1. *Si podemos demostrar que la propiedad P es cierta para todo elemento en $\mathbb{N}$ mediante principio de inducción fuerte, entonces también podemos hacerlo mediante el principio de inducción (normal).*

Demostración. Dado que tenemos una demostración de P sobre $\mathbb{N}$ utilizando inducción fuerte, entonces tenemos una demostración de lo siguiente:

$$P(0) \quad \text{y} \quad \forall n\big((P(0) \wedge \cdots \wedge P(n)) \to P(n+1) \big).$$

Para obtener una demostración de P sobre $\mathbb{N}$ utilizando inducción (normal), demostraremos una propiedad P' más fuerte. Esta propiedad P' es cierta en n si y solo si todo elemento $1 \leq j \leq n$ tiene la propiedad P. Claramente, si demostramos P' por inducción (normal), podemos concluir que todo elemento en $\mathbb{N}$ tiene la propiedad P.

La demostración es como sigue:

- **Caso base:** Claramente, $P(0)$ es cierto si y solo si $P'(0)$ es cierto. Sabemos que lo anterior es cierto por nuestra demostración basada en inducción fuerte.
- **Caso inductivo:** Debemos demostrar que $\forall n(P'(n) \to P'(n+1))$. Considere un $n \geq 1$ cualquiera y asuma que $P'(n)$ es cierto. Por definición, esto implica que $P(0) \wedge P(1) \wedge \cdots \wedge P(n)$ es cierto. Por nuestra demostración por inducción fuerte tenemos entonces que $P(n+1)$ es cierto. Por tanto, $P(0) \wedge \cdots \wedge P(n+1)$ es cierto, lo que implica que $P'(n+1)$ es cierto.

Esto finaliza la demostración. □

De ahora en adelante, por tanto, cuando nos refiramos al principio de inducción nos estaremos refiriendo a su versión fuerte. Por el resultado anterior, claro está, esto no hace ninguna diferencia.

2.2. El principio del buen orden

¿Qué significa que hablemos del *principio* de inducción? Esto es algo que asumimos que es cierto, sin tener realmente una demostración. Existen otros principios que hablan sobre $\mathbb{N}$; uno de ellos es el *principio del buen orden*. Este dice que los naturales están bien ordenados, es decir, que todo subconjunto no vacío de $\mathbb{N}$ tiene un menor elemento. A pesar de su simplicidad, el principio del buen orden puede ser utilizado para demostrar propiedades interesantes acerca de cada número natural en $\mathbb{N}$ como muestra el siguiente ejemplo.

Ejemplo 2.2.1. En un torneo cada jugador juega con cada uno de los otros jugadores exactamente una vez. El juego entrega un ganador y un perdedor. Un *ciclo* de largo $n \geq 3$ en un torneo es una secuencia $p_1, p_2, \ldots, p_n$ tal que los p_i's son jugadores (no necesariamente distintos), el jugador p_i le gana a p_{i+1}, para cada $1 \leq i \leq n-1$, y p_n le gana a p_1. Demostraremos usando el principio del buen orden que para todo $n \geq 3$ si un torneo tiene un ciclo de largo n, entonces tiene un ciclo de largo 3.

Sea A el conjunto de todos los naturales $n \geq 3$ tal que el torneo tiene un ciclo de largo n. Asumamos que el torneo tiene al menos un ciclo. Por tanto, A es no vacío, y por el principio del buen orden tiene un menor elemento; digamos, m. Por contradicción, supongamos además que $m > 3$.

Considere entonces un ciclo $p_1, p_2, \ldots, p_m$. Luego p_1 debe ganarle a p_3, por que de otra forma p_1, p_2, p_3 sería un ciclo de largo 3, lo que es una contradicción. Pero entonces $p_1, p_3, \ldots, p_m$ es un ciclo de largo $(m-1)$, lo que contradice el hecho de que m es el menor elemento de A. □

2.2.1. Buen orden vs inducción

Interesantemente, las propiedades acerca de los elementos de $\mathbb{N}$ que pueden ser demostradas usando el principio de inducción también pueden ser demostradas mediante el principio del buen orden:

Proposición 2.2.1. *Toda demostración por inducción de que los elementos en $\mathbb{N}$ tienen la propiedad P puede ser transformada en una demostración de lo mismo mediante el principio del buen orden.*

Demostración. Nuestra demostración por inducción nos entrega lo siguiente:

$$P(0) \quad \text{y} \quad \forall n(P(n) \to P(n+1)).$$

Demostraremos entonces que todo $n \in \mathbb{N}$ tiene la propiedad P utilizando ahora el principio del buen orden. Suponga, por contradicción, que existe un número natural que no tiene la propiedad P. Por el principio del buen orden hay entonces un menor natural m que no cumple P. Pero $m > 0$ puesto que tenemos una demostración de $P(0)$. Además, por minimalidad de m tenemos que $(m-1)$ tiene la propiedad P. Pero contamos con una demostración de que $P(m-1) \to P(m)$, y por tanto m satisface P, lo que es una contradicción. □

Un corolario de esta demostración es que si asumimos que el principio del buen orden es cierto en $\mathbb{N}$, entonces el principio de inducción también lo es. (Explique por qué). Por otro lado, si asumimos que el principio de inducción es cierto en los naturales entonces el principio del buen orden también lo es. La razón es que por medio del principio de inducción podemos demostrar el principio del buen orden, como mostramos a continuación. Demostraremos que para todo $n \geq 0$, si $A \subseteq \mathbb{N}$ contiene al elemento n entonces A tiene un menor elemento. Claramente esto implica que el principio del buen orden es cierto. El caso base es que $0 \in A$. Por tanto, 0 es menor elemento de A. Para el caso inductivo suponemos que $(n+1) \in A$. Si $j \in A$, para $0 \leq j \leq n$, luego A tiene menor elemento por HI. En caso contrario, $(n+1)$ es el menor elemento de A.

Concluímos entonces lo siguiente:

Corolario 2.2.1. *El principio de inducción en los naturales es cierto si y solo si el principio del buen orden en los naturales es cierto.*

2.3. Inducción estructural

La inducción puede ser utilizada como herramienta para demostrar propiedades no solo de los números naturales, sino de todo conjunto definido *recursivamente*. Tales definiciones recursivas constan de dos partes. Con la *regla base* especificamos una colección de elementos que pertenecen al conjunto, mientras que con la *regla inductiva* explicamos cómo formar nuevos elementos en el conjunto desde aquellos que ya han sido generados. Existe también una *regla de exclusión* implícita, que dice que los únicos elementos en el conjunto son aquellos generados mediante las dos reglas anteriores. Podemos entonces demostrar inductivamente propiedades sobre este conjunto al hacer inducción en el número n de aplicaciones de la regla inductiva que son necesarias para generar un objeto. Esto recibe el nombre de *inducción estructural*.

Ejemplo 2.3.1. Sea Σ un alfabeto finito. El conjunto Σ^* de *palabras* sobre Σ se define inductivamente como sigue:

- **Regla base:** La palabra vacía ϵ está en Σ.
- **Regla inductiva:** Si $w \in \Sigma^*$ y $a \in \Sigma$, entonces $wa \in \Sigma^*$.

□

Las definiciones recursivas también pueden ser utilizadas para definir operaciones y funciones. Por ejemplo:

Ejemplo 2.3.2. La operación $\cdot$ de *concatenación* de palabras $w, w' \in \Sigma^*$ se define inductivamente como sigue:

- **Regla base:** Si $w \in \Sigma^*$, entonces $w \cdot \epsilon = w = \epsilon \cdot w$.
- **Regla inductiva:** Si $w, w' \in \Sigma^*$ y $a \in \Sigma$, entonces $w \cdot (w'a) = (w \cdot w')a$ y $(aw') \cdot w = a(w \cdot w')$.

El *largo* $l(w)$ de una palabara $w \in \Sigma^*$ se define inductivamente como sigue:

- **Regla base:** $l(\epsilon) = 0$.
- **Regla inductiva:** Si $w \in \Sigma^*$ y $a \in \Sigma$, entonces $l(wa) = l(w) + 1$.

□

Podemos ahora demostrar propiedades que relacionen estos conceptos mediante el principio de inducción:

Ejemplo 2.3.3. Demuestre que para todo $w, w' \in \Sigma^*$ se cumple que:

$$l(w \cdot w') \quad = \quad l(w) + l(w'). \tag{2.1}$$

Sea $w' \in \Sigma$ una palabra arbitraria en Σ^* y $P(w')$ la propiedad que expresa que para todo $w \in \Sigma^*$ se cumple (2.1). Demostraremos que $P(w')$ es cierta para todo $w' \in \Sigma^*$ mediante inducción estructural en la definición recursiva de Σ^* (o, equivalentemente, mediante inducción en el número de veces que la regla inductiva en la definición de Σ^* ha sido ocupada para producir w'):

- **Caso base:** Sucede cuando w' se obtiene sin ninguna aplicación de la regla recursiva. Por tanto, $w' = \epsilon$. Por definición, $l(w \cdot \epsilon) = l(w) = l(w) + 0 = l(w) + l(\epsilon)$.
- **Caso inductivo:** En este caso w' se obtiene mediante $n + 1$ aplicaciones de la regla inductiva, donde $n \geq 0$. Luego, $w' = va$, donde $v \in \Sigma^*$, $a \in \Sigma$, y v ha sido obtenido mediante n aplicaciones de la regla inductiva. Por definición, $l(w \cdot (va)) = l((w \cdot v)a) = l(w \cdot v) + 1$. Por HI $l(w \cdot v) = l(w) + l(v)$, y por tanto $l(w \cdot (va)) = l(w) + l(v) + 1 = l(w) + l(va)$.

Esto concluye la demostración. □

Ejemplo 2.3.4. El *inverso* w^{-1} de una palabra $w \in \Sigma^*$ se obtiene al leer w de derecha a izquierda. Podemos recursivamente definir tal inverso como sigue:

- **Regla base:** $\epsilon^{-1} = \epsilon$.
- **Regla inductiva:** Si $w \in \Sigma^*$ y $a \in \Sigma$, entonces $(wa)^{-1} = aw^{-1}$.

Demostraremos ahora que:

$$(w_1 \cdot w_2)^{-1} = w_2^{-1} \cdot w_1^{-1}, \tag{2.2}$$

para todo $w_1, w_2 \in \Sigma^*$. Sea $w_2 \in \Sigma$ una palabra arbitraria en Σ^* y $P(w_2)$ la propiedad que expresa que para todo $w_1 \in \Sigma^*$ se cumple (2.2). Demostraremos que $P(w_2)$ es cierta para todo $w_2 \in \Sigma^*$ mediante inducción estructural en la definición recursiva del conjunto Σ^* (o, equivalentemente, mediante inducción en el número de veces que la regla inductiva en la definición de Σ^* ha sido ocupada para producir w_2).

- **Caso base:** Entonces $w_2 = \epsilon$. Luego por definición $(w_1 \cdot \epsilon)^{-1} = w_1^{-1}$, lo que es igual a $\epsilon \cdot w_1^{-1}$. Por definición, esta última expresión es igual a $\epsilon^{-1} \cdot w_1^{-1}$.
- **Caso inductivo:** Tenemos que $w_2 = ya$, donde $y \in \Sigma^*$ y $a \in \Sigma$. Luego, $(w_1 \cdot w_2)^{-1} = (w_1 \cdot (ya))^{-1} = ((w_1 \cdot y)a)^{-1} = a(w_1 \cdot y)^{-1}$. Pero esto último es igual a $a(y^{-1} \cdot w_1^{-1})$ por HI. Por definición de concatenación, esta expresión es igual a $(ay^{-1}) \cdot w_1^{-1}$, lo que corresponde por definición a $(ya)^{-1} \cdot w_1^{-1}$, y por tanto a $w_2^{-1} \cdot w_1^{-1}$.

Esto finaliza la demostración. □

Ejercicio. *Defina recursivamente el conjunto de palabras en Σ^* que son palíndromes. Encuentre una expresión que cuente el número de palíndromes en Σ^* de largo n, para $n \geq 0$.*

2.4. Algoritmos recursivos

A veces podemos reducir la solución de un problema en una entrada de cierto tamaño n, para $n \geq 0$, a la solución del mismo problemas sobre entradas de tamaño $m < n$. Si esto es el caso podemos seguir descomponiendo nuestra entrada, hasta que este haya sido reducida a algún caso inicial para el cual la solución es conocida. En el caso de ciertos problemas que tienen esta propiedad, podemos además diseñar un *algoritmo recursivo* que resuelva el problema eficientemente. Tal algoritmo soluciona el problema reduciéndolo recursivamente a entradas de tamaño menor. Presentaremos dos ejemplos de tales algoritmos en esta sección.

Intervalo de suma máxima En este caso la entrada consiste de una secuencia $a_1, \ldots, a_n$ de enteros positivos y negativos. Queremos determinar un *intervalo de suma máxima*, es decir, un par (i, j), con $1 \leq i \leq j \leq n$, que maximice el valor $\sum_{l=i}^{j} a_l$ de la suma de los elementos en el intervalo $a_i, \ldots, a_j$.

Ejemplo 2.4.1. Consider la secuencia $-2, 11, -4, 13, -5, -2$. Entonces el intervalo de suma máxima va entre 11 y 13, y su valor es 11 - 4 + 13 = 20. □

El algoritmo *naïve* que resuelve este problema realiza lo siguiente:

1. Por cada $1 \leq i \leq j \leq n$, computa $s_{i,j} = \sum_{l=i}^{j} a_l$.
2. Selecciona un (i, j) que maximiza $s_{i,j}$.

Aunque este algoritmo trabaja tiempo polinomial, dista de ser óptimo. En efecto, es fácil ver que su costo es $\Theta(n^3)$, mientras que incluso una modificación sencilla de él alcanza una cota mejor.

Ejercicio. *Encuentre una modificación del procedimiento anterior que encuentre el intervalo de suma máxima en tiempo $O(n^2)$.*

A continuación demostraremos que existe un simple procedimiento recursivo que puede resolver este problema en tiempo lineal $O(n)$. La herramienta principal que utilizamos es la siguiente proposición, que nos dice que la solución del problema en una entrada $a_1, \ldots, a_n$ puede ser computada en términos de la solución en $a_1, \ldots, a_{n-1}$ y el intervalo de suma máxima en $a_1, \ldots, a_n$ que utiliza a a_{n-1}. Formalmente, para $n > 1$ sea $I(a_1, \ldots, a_n)$ un intervalo de suma máxima en $a_1, \ldots, a_n$ y $T(a_1, \ldots, a_n)$ un intervalo de suma máxima en $a_1, \ldots, a_n$ que termina en n, es decir, uno de la forma (i, n), para $1 \leq i \leq n$. Denotamos, además, por $||I(a_1, \ldots, a_n)||$ y $||T(a_1, \ldots, a_n)||$ el valor de la suma de los elementos en esos intervalos. Entonces:

Proposición 2.4.1. *Asuma que $T(a_1, \ldots, a_{n-1}) = (i, n-1)$. Se cumple que:*

$$T(a_1, \ldots, a_n) = \begin{cases} (n, n), & \text{si } a_n > ||T(a_1, \ldots, a_{n-1})|| + a_n, \\ (i, n), & \text{en caso contrario.} \end{cases}$$

Además:

$$I(a_1, \ldots, a_n) = \begin{cases} I(a_1, \ldots, a_{n-1}), & \text{si } ||I(a_1, \ldots, a_{n-1})|| > ||T(a_1, \ldots, a_n)||, \\ T(a_1, \ldots, a_n), & \text{en caso contrario.} \end{cases}$$

Demostración. Comencemos con $T(a_1, \ldots, a_n)$. Entonces el intervalo de suma máxima en $a_1, \ldots, a_n$ de la forma (i, n) puede ser de la forma (n, n) o (j, n) para $j < n$. El primer caso solo ocurre cuando $a_n > \sum_{l=i}^{n-1} a_l + a_n = ||T(a_1, \ldots, a_{n-1})|| + a_n$, lo que se considera en el primer caso de la definición de $T(a_1, \ldots, a_n)$ (note que esto sucede solo si $||T(a_1, \ldots, a_{n-1})|| < 0$). El segundo caso ocurre precisamente cuando $a_n \leq \sum_{l=i}^{n-1} a_l + a_n = ||T(a_1, \ldots, a_{n-1})|| + a_n$, lo que se considera en el segundo caso de la definición de $T(a_1, \ldots, a_n)$.

Analicemos ahora a $I(a_1, \ldots, a_n)$. El intervalo de suma máxima en $a_1, \ldots, a_n$ es de la forma (i, n) o (i, j), para $j < n$. En el primer caso tenemos que $I(a_1, \ldots, a_n) = T(a_1, \ldots, a_n)$, lo que se considera en el primer caso de la definición de $I(a_1, \ldots, a_n)$. En el segundo caso tenemos que $I(a_1, \ldots, a_n) = I(a_1, \ldots, a_{n-1})$, lo que se considera en el segundo caso de la definición de $I(a_1, \ldots, a_n)$. □

Esto nos permite construir una simple definición recursiva para computar el valor de $I(a_1, \ldots, a_n)$. Para ello computamos recursivamente los valores de $I(a_1, \ldots, a_i)$ y $T(a_1, \ldots, a_i)$, para cada $1 \leq i \leq n$, partiendo desde el caso base $i = 1$ y luego utilizando la propiedad expresada en la Proposición 2.4.1:

- **Caso base:** Claramente, $I(a_1) = T(a_1) = (1, 1)$.
- **Caso inductivo:** Para $i + 1$, cuando $1 \leq i < n$, si $T(a_1, \ldots, a_i) = (j, i)$ tenemos que:

$$T(a_1, \ldots, a_{i+1}) = \begin{cases} (i+1, i+1), & \text{si } a_{i+1} > ||T(a_1, \ldots, a_i)|| + a_{i+1}, \\ (j, i+1), & \text{en caso contrario.} \end{cases}$$

  Además:

$$I(a_1, \ldots, a_{i+1}) = \begin{cases} I(a_1, \ldots, a_i), & \text{si } ||I(a_1, \ldots, a_i)|| > ||T(a_1, \ldots, a_{i+1})||, \\ T(a_1, \ldots, a_{i+1}), & \text{en caso contrario.} \end{cases}$$

El intervalo de suma máxima en $a_1, \ldots, a_n$ corresponderá a $I(a_1, \ldots, a_n)$.

¿Cuál es el costo de computar $I(a_1, \ldots, a_n)$? Debemos computar iterativamente $I(a_1, \ldots, a_i)$ y $T(a_1, \ldots, a_i)$, para cada $1 \leq i \leq n$. Para esto debemos realizar comparaciones que involucran a $||I(a_1, \ldots, a_{i-1})||$ y $||T(a_1, \ldots, a_{i-1})||$. Sin embargo, es fácil ir computando estos valores también iterativamente mediante la siguiente definición recursiva:

- **Caso base:** Claramente, $||I(a_1)|| = ||T(a_1)|| = a_1$.
- **Caso inductivo:** Para $i + 1$, cuando $1 \leq i < n$, tenemos que:

$$||T(a_1, \ldots, a_{i+1})|| = \begin{cases} a_{i+1}, & \text{si } a_{i+1} > ||T(a_1, \ldots, a_i)|| + a_{i+1}, \\ ||T(a_1, \ldots, a_i)|| + a_{i+1}, & \text{en caso contrario.} \end{cases}$$

  Además:

$$||I(a_1, \ldots, a_{i+1})|| = \begin{cases} ||I(a_1, \ldots, a_i)||, & \text{si } ||I(a_1, \ldots, a_i)|| > ||T(a_1, \ldots, a_{i+1})||, \\ ||T(a_1, \ldots, a_{i+1})||, & \text{en caso contrario.} \end{cases}$$

Por tanto, cada paso de nuestro algoritmo toma tiempo constante, y podemos llevar a cabo la computación de $I(a_1, \ldots, a_n)$ y su valor $||I(a_1, \ldots, a_n)||$ en tiempo $O(n)$.

Ejercicio. *Corra el algoritmo en la secuencia dada en el Ejemplo 2.4.1.*

Máxima común subsecuencia Sea w una palabra. Una *subsecuencia* de w se obtiene desde w al borrar cero o más de sus letras. Por ejemplo, si $w = abcbdab$, entonces algunas de sus subsecuencias son:

$$\epsilon,\ a,\ b,\ ab,\ cd,\ ada,\ adab,\ abcdab.$$

Si w, w' son palabras, denotamos mediante $\mathrm{MCS}(w, w')$ el conjunto de todas las subsecuencias comunes a w, w' que son de largo máximo. El problema de la *máxima común subsecuencia* se define como sigue: dadas palabras w, w' encontrar al menos una subsecuencia común en $\mathrm{MCS}(w, w')$ junto con su largo $||\mathrm{MCS}(w, w')||$.

Ejemplo 2.4.2. Sean $w = abcbdab$ y $w' = bdcaba$. Entonces:

$$\mathrm{MCS}(w, w') = \{bcab,\ bdab,\ bcba\}.$$

Por tanto, $||\mathrm{MCS}(w, w')|| = 4$. □

El algoritmo *naïve* que resuelve el problema de la máxima común subsecuencia es el siguiente:

1. Itere sobre todas las subsecuencias u de w. Por cada una de ellas verifique si es también subsecuencia de w'. De ser así, agregue u al conjunto M.
2. Elija un elemento de largo máximo en M, y compute su largo.

Lamentablemente este algoritmo es exponencial en el largo de w, ya que el número de subsecuencias de una palabra de largo m es 2^m.

A continuación diseñaremos un algoritmo recursivo que resuelve el problema en tiempo polinomial $O(m \cdot n)$, donde m y n representan el largo de w y w', respectivamente. Tal algoritmo hace uso del siguiente resultado que expresa como una solución al problema puede ser expresada recursivamente en términos de la solución a problemas de menor tamaño.

Proposición 2.4.2. *Sean $a_1 \dots a_m$ y $b_1 \dots b_n$ palabras. Se cumple que:*

- *Si $a_m = b_n$, entonces $\mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_n)$ es el conjunto de palabras de la forma $w'' a_m$, para $w'' \in \mathrm{MCS}(a_1 \dots a_{m-1}, b_1 \dots b_{n-1})$. Por tanto:*

$$||\mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_n)|| = 1 + ||\mathrm{MCS}(a_1 \dots a_{m-1}, b_1 \dots b_{n-1})||.$$

- *Si $a_m \neq b_n$, entonces $\mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_n)$ corresponde a:*
  - $\mathrm{MCS}(a_1 \dots a_{m-1}, b_1 \dots b_n) \cup \mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_{n-1})$, *si:*

$$||\mathrm{MCS}(a_1 \dots a_{m-1}, b_1 \dots b_n)|| = ||\mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_{n-1})||.$$

  - $\mathrm{MCS}(a_1 \dots a_{m-1}, b_1 \dots b_n)$, *si:*

$$||\mathrm{MCS}(a_1 \dots a_{m-1}, b_1 \dots b_n)|| > ||\mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_{n-1})||.$$

  - $\mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_{n-1})$, *si:*

$$||\mathrm{MCS}(a_1 \dots a_{m-1}, b_1 \dots b_n)|| < ||\mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_{n-1})||.$$

  Por tanto, $||\mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_n)||$ corresponde a:

$$\max\{||\mathrm{MCS}(a_1 \dots a_{m-1}, b_1 \dots b_n)||, ||\mathrm{MCS}(a_1 \dots a_m, b_1 \dots b_{n-1})||\}.$$

Ejercicio. *Demuestre la Proposición 2.4.2.*

i \ j	0(ϵ)	1(b)	2(d)	3(c)	4(a)	5(b)	6(a)
0(ϵ)	0	0	0	0	0	0	0
1(a)	0	0	0	0	1	1	1
2(b)	0	1	1	1	1	2	2
3(c)	0	1	1	2	2	2	2
4(b)	0	1	1	2	2	3	3
5(d)	0	1	2	2	2	3	3
6(a)	0	1	2	2	3	3	4
7(b)	0	1	2	2	3	4	4

Cuadro 2.1: Tabla que muestra los valores $c(i, j)$, para $0 \leq i \leq 7$ y $0 \leq j \leq 6$, con respecto al Ejemplo 2.4.3. Los valores entre paréntesis representan la letra de w y w' que aparece en la posición respectiva.

Utilizando la Proposición 2.4.2 podemos definir recursivamente el valor $c(i, j)$, para $0 \leq i \leq m$ y $0 \leq j \leq n$, que corresponde a $||\text{MCS}(a_1 \dots a_i, b_1 \dots b_j)||$. (Cuando $i = 0$ entonces asumimos que $a_1 \dots a_i = \epsilon$, y análogamente cuando $j = 0$ asumimos que $b_1 \dots b_j = \epsilon$). En efecto:

- **Caso base:** Si $i = 0$ o $j = 0$, entonces $c(i, j) = 0$.
- **Caso inductivo:** Si $1 \leq i \leq m$ y $1 \leq j \leq n$, entonces:

$$c(i,j) = \begin{cases} 1 + c(i-1, j-1), & \text{si } a_i = b_j, \\ \max\{c(i-1,j), c(i,j-1)\}, & \text{si } a_i \neq b_j. \end{cases}$$

Esto nos permite ir recursivamente computando los valores $c(i, j)$, para $0 \leq i \leq m$ y $0 \leq j \leq n$, hasta obtener $c(m, n) = ||\text{MCS}(a_1 \dots a_m, b_1 \dots b_n)||$. El cómputo de $c(i, j)$ puede realizarse en tiempo constante utilizando los valores $c(i-1, j-1)$, $c(i, j-1)$ y $c(i-1, j)$. Por tanto, calcular $c(m, n)$ toma tiempo $O(m \cdot n)$. Interesantemente, no solo podemos calcular $c(m, n) = ||\text{MCS}(a_1 \dots a_m, b_1 \dots b_n)||$, sino que en el proceso podemos ir obteniendo información importante que nos permita computar al menos una palabra en el conjunto $\text{MCS}(a_1 \dots a_m, b_1 \dots b_n)$. Es más fácil explicar este proceso mediante un ejemplo.

Ejemplo 2.4.3. Suponga que queremos encontrar una máxima común subsecuencia de $w = abcbdab$ y $w' = bdcaba$. Entonces podemos computar recursivamente los valores $c(i, j)$, para $0 \leq i \leq 7$ y $0 \leq j \leq 6$, siguiendo el procedimiento anterior. Los valores obtenidos se muestran en el Cuadro 2.1.

Aún cuando este cuadro no nos aporta información con respecto a las palabras en $\text{MCS}(w, w')$, podemos agregarle un mínimo de información que nos permita extraer tales palabras en caso de que queramos. Para ello le pondremos una etiqueta a cada casillero, la que puede ser S, A, I, o C. La etiqueta S se coloca justamente en las posiciones (i, j) tal que $a_i = b_j$. Esto representa que debemos agregar ese símbolo a nuestra palabra en $\text{MCS}(w, w')$ si es que pasamos por ese casillero, y luego seguir hacia el casillero $(i-1, j-1)$. Si $a_i \neq b_j$ utilizamos la etiqueta A si $c(i-1, j) > c(i, j-1)$ (es decir, el valor de arriba es mayor al de la izquierda), la etiqueta I si $c(i-1, j) < c(i, j-1)$ (es decir, el valor de la izquierda es mayor al de arriba), y C si $c(i-1, j) = c(i, j-1)$. Si leemos la etiqueta A debemos avanzar hacia arriba para encontrar una palabra en $\text{MCS}(w, w')$, si leemos I debemos ir a la izquierda, y si leemos C hacia cualquiera de los dos lados. Esto se muestra en el Cuadro 2.2. Para encontrar una palabra en $\text{MCS}(w, w')$ seguimos ahora el sentido de las anotaciones, agregando una letra de derecha a izquierda cada vez que vemos la anotación S. Por ejemplo, siguiendo el camino:

$$(7,6) \to (7,5) \to (6,4) \to (5,3) \to (4,3) \to (3,3) \to (2,2) \to (2,1) \to (1,0)$$

obtenemos la máxima común subsecuencia *bcab*. □

i \ j	0(ϵ)	1(b)	2(d)	3(c)	4(a)	5(b)	6(a)
0(ϵ)	0	0	0	0	0	0	0
1(a)	0	0(C)	0(C)	0(C)	1(S)	1(I)	1(I)
2(b)	0	1(S)	1(I)	1(I)	1(C)	2(S)	2(I)
3(c)	0	1(A)	1(C)	2(S)	2(I)	2(C)	2(C)
4(b)	0	1(A)	1(C)	2(A)	2(C)	3(S)	3(I)
5(d)	0	1(A)	2(S)	2(C)	2(C)	3(A)	3(I)
6(a)	0	1(A)	2(A)	2(C)	3(S)	3(C)	4(S)
7(b)	0	1(S)	2(A)	2(C)	3(A)	4(S)	4(C)

Cuadro 2.2: Tabla que muestra los valores $c(i, j)$ junto con sus etiquetas, para $0 \leq i \leq 7$ y $0 \leq j \leq 6$, con respecto al Ejemplo 2.4.3.

Ejercicios Finales

1. Sea n un número natural cualquiera. Definimos $d_i(n)$ como el dígito que aparece en la posición i en n con $d_1(n)$ el dígito más significativo, por ejemplo $d_3(1850) = 5$. Demuestre que para todo n con k dígitos se cumple que
$$n - \sum_{i=1}^{k} d_i(n) = 9r \quad \text{para algún } r \text{ natural},$$
o sea que n menos la suma de sus dígitos es siempre un múltipo de 9.

2. Sea $f : N^+ \to N$ una funcion definida como sigue
$$f(n) = \begin{cases} 1 & \text{if } n = 1 \\ 2 \cdot f(\lfloor \frac{n}{2} \rfloor) + n & \text{if } n \geq 2 \end{cases}$$
Demuestre utilizando induccion fuerte que $f(n)$ es $O(n \log n)$.

3. Considere una función $f : \mathbb{N} \to \mathbb{N}$ que satisface la siguiente definición recursiva:
   - $f(0) = f(1) = 1$.
   - $f(n) \leq f(\lfloor \frac{7n}{10} \rfloor) + f(\lfloor \frac{n}{5} \rfloor) + n$, para $n \geq 2$.

   Demuestre que existe $c \geq 1$ tal que $f(n) \leq cn$, para todo $n \geq 0$.

4. Hay un plato de galletas en una mesa. Tu y un amigo deciden jugar un juego con ellas. En este juego el que saque la última galleta del plato pierde, los jugadores, partiendo contigo, se van alternando para retirar al menos una galleta del plato y máximo 3. Determina qué tiene que pasar para que tengas una estrategia ganadora (Es decir, ganes independiente de lo que haga tu amigo) y luego demuestrelo usando inducción.
   Hint: Analice que ocurre cuando hay menos de 5 galletas y luego vea como realizar la inducción.

5. Demuestre por inducción que $7^n - 2^n$ es multiplo de 5.

6. Considere n lineas en el plano de forma que no existen lineas paralelas y nunca pasa que 3 lineas pasen por el mismo punto. ¿En cuantas regiones divide al plano?

7. Imaginemos un círculo. Sobre el perímetro de este pintamos n puntitos blancos y n puntitos negros. Queremos pasar por todo el contorno de este llevando cuenta de los puntitos que pasamos. Para hacerlo

partimos con nuestra cuenta en 0, luego: cada vez que pasamos un puntito blanco sumamos uno a la cuenta, cada vez que pasamos uno negro, restamos uno.

Demuestre que independiente de cómo estén repartidos los colores de los puntitos, hay un lugar del que podemos partir tal que al recorrer todos los puntitos en orden la cuenta nunca sea negativa.

8. Recuerde que el dominio constructible de las expresiones aritméticas $\mathcal{E}_{\mathbb{N}}$ se define por:

   - Si k es natural, entonces k es una expresión.
   - Si E_1 y E_2 son expresiones, entonces $E_1 + E_2$ es una expresión.
   - Si E_1 y E_2 son expresiones, entonces $E_1 * E_2$ es una expresión.
   - Si E es una expresión, entonces (E) es una expresión.

   a) Defina inductivamente en la construcción de $\mathcal{E}_{\mathbb{N}}$ el operador $\#_{op} : \mathcal{E}_{\mathbb{N}} \to \mathbb{N}$ que cuenta la cantidad de operadores aritméticos (+ y *) de una expresión.

   b) Defina inductivamente en la construcción de $\mathcal{E}_{\mathbb{N}}$ el operador $\#_{num} : \mathcal{E}_{\mathbb{N}} \to \mathbb{N}$ que cuenta la cantidad de naturales que aparecen en una expresión.

9. Demuestre que para todo entero positivo n,

$$\frac{1}{2} \cdot \frac{3}{4} \cdots \frac{2n-1}{2n} < \frac{1}{\sqrt{3n}}$$

10. Demuestre utilizando inducción fuerte que la siguiente propiedad es cierta para cada entero positivo n: $\sqrt{2} \neq n/b$ para todo entero positivo b.

11. El conjunto B de los strings de paréntesis *balanceados* se define recursivamente como sigue: (1) El string vacío ϵ está en B; y (2) si $x, y \in B$ entonces (x) y xy pertenecen a B.

    Definimos la función N en el conjunto de strings de paréntesis de la siguiente forma:

$$N(\epsilon) = 0 \ ; \ N(() = 1 \ ; \ N()) = -1$$
$$N(uv) = N(u) + N(v)$$

    a) Demuestre usando inducción estructural que si un string de paréntesis w es balanceado, entonces $N(w) = 0$ y $N(u) \geq 0$ para todo *prefijo* u de w, i.e. para todo u tal que $w = uu'$.

    b) Demuestre usando inducción estructural que si un string de paréntesis w satisface que $N(w) = 0$ y $N(u) \geq 0$ para todo *prefijo* u de w, entonces w es balanceado.

12. Sean a, b números reales tales que $0 < b < a$. Demuestre por inducción que si n es un entero positivo, entonces

$$a^n - b^n \ \leq \ na^{n-1}(a-b).$$

13. Demuestre por inducción que $n \cdot (n+1) \cdot (n+2)$ es divisible por 6, para todo entero positivo n.

14. Demuestre por inducción que

$$\sum_{j=1}^{n} j(j+1)(j+2)\cdots(j+k-1) \ = \ \frac{n(n+1)(n+2)\cdots(n+k)}{k+1},$$

    para todo entero positivo k y n.

15. Utilice inducción fuerte para demostrar que los números $1, 2, \ldots, n$ siempre pueden ser ordenados de tal forma que el promedio de cualquiera dos de estos números nunca aparezca entre ellos.

16. Considere la suma de términos $\mathcal{A}_n$ $(n \geq 2)$ definida como:

$$\frac{1}{1 \times 2} + \frac{1}{2 \times 3} + \frac{1}{3 \times 4} + \cdots + \frac{1}{(n-1)n}.$$

Demuestre que existen polinomios $f(n)$ y $g(n)$ de grado 1 tal que $\mathcal{A}_n = \frac{f(n)}{g(n)}$, para todo $n \geq 2$.

(*Hint:* Calcule los valores iniciales de la serie $(\mathcal{A}_n)_{n \geq 2}$, y con ellos elucubre cuáles son los polinomios $f(n)$ y $g(n)$. Luego demuestre que su elucubración es correcta mediante inducción).

17. Sea n un entero y considere un tablero de $2^n \times 2^n$ casilleros. Demuestre usando inducción que si al tablero se le saca un casillero entonces puede ser *embaldosado* por *triominos*, donde:

- Un *triomino* es una figura en forma de L que cubre 3 unidades cuadradas.
- Un conjunto de triominos *embaldosa* un tablero si cada casillero se halla cubiero por algun triomino, los triominos no se superponen y ningún triomino queda colgando fuera del tablero.

18. Demuestre usando inducción que si $I_1, I_2, \ldots, I_n$ es una colección de intervalos abiertos sobre los números reales, $n \geq 2$, y para cada $1 \leq i, j \leq n$ se tiene que $I_i \cap I_j \neq \emptyset$, entonces $I_1 \cap I_2 \cap \cdots \cap I_n \neq \emptyset$. (Recuerde que un intervalo abierto de números reales es un conjunto de números reales x tal que $a < x < b$, donde a y b son números reales con $a < b$).

19. Considere el conjunto B de todas las palabras sobre alfabeto $\{0, 1\}$ que tienen tantos 0s como 1s. En este ejercicio demostraremos que este conjunto coincide con el conjunto S de palabras sobre alfabeto $\{0, 1\}$ definido recursivamente como sigue:

a) $\epsilon \in S$.

b) Si $u, v \in S$ entonces las palabras $0u1$, $1u0$ y uv también están en S.

Se pide lo siguiente:

- Demuestre por inducción estructural que $S \subseteq B$.
- Demostraremos ahora lo contrario, es decir, $B \subseteq S$. Para esto demostraremos por inducción que $P(n)$ es cierto para todo $n \geq 0$, donde $P(n)$ es la propiedad que expresa que toda palabra en B de largo n está en S.

  Note que $P(0)$ es trivialmente cierto ya que la única palabra de largo 0 es ϵ, la cual pertenece a S. Considere ahora $P(n+1)$, para $n \geq 0$, y asuma por hipótesis inductiva que $P(j)$ es cierto para todo $0 \leq j \leq n$. Sea w una palabra en B de largo $n+1$.

  - Suponga primero que w comienza y termina con letras distintas. Explique por qué $w \in S$.
  - Suponga ahora que w comienza y termina con la misma letra. Explique por qué $w \in S$.

    Ayuda: Demuestre que $w = uv$, donde u, v son palabras no vacías en B. Para esto considere el *desbalance* de cada prefijo u de w, definido como el número de 1s menos el número de 0s en u. ¿Cuál es el desbalance del prefijo que consiste del primer símbolo de w solamente? ¿Cuál es el desbalance del prefijo que contiene todos los símbolos de w salvo el último?

20. Asuma que tenemos n tareas $t_1, \ldots, t_n$, y que con cada tarea t_i $(1 \leq i \leq n)$ asociamos la siguiente información:

- El momento $c_i \geq 0$ en que la tarea comienza y otro $f_i > c_i$ en que finaliza.
- El beneficio $b_i > 0$ de que la tarea se ejecute.

Decimos que las tareas t_i y t_j son *compatibles* si sus intervalos de ejecución no se intersectan; es decir, si $f_i < c_j$ o $f_j < c_i$.

Asuma ahora que las tareas $t_1, \ldots, t_n$ se hallan ordenadas no decrecientemente con respecto a los términos de finalización de tareas; es decir, para todo $1 \leq i \leq j \leq n$ se cumple que $f_i \leq f_j$. Defina $B : \{0, \ldots, n\} \to \mathbb{N}$ de tal forma que $B(i)$ corresponde al mayor beneficio que se puede obtener al ejecutar algunas de las i primeras tareas de forma compatible. Formalmente:

$$B(i) \;=\; \max\{\sum_{t_j \in A} b_j \mid A \subseteq \{t_1, \ldots, t_i\} \text{ y no hay dos tareas distintas en } A \text{ que sean incompatibles}\}.$$

a) Defina recursivamente a $B(i)$ en términos de $B(i-1)$ y $B(\text{pred}(i))$, donde $\text{pred}(i)$ es el mayor $j < i$ tal que t_j y t_i son compatibles.

b) De lo anterior, ¿qué podemos concluir con respecto al costo del problema de computar un conjunto de tareas mutuamente compatibles que maximicen el beneficio?

21. Cuando un corrector ortográfico encuentra un error en una palabra u utiliza una medida que le permite determinar cuáles son las palabras más "parecidas" a u. Para esto utiliza una noción de distancia entre palabras u y v que corresponde al menor número de *operaciones de edición* que son necesarias para convertir a u en v (estas operaciones se aplican simultáneamente en u). Las operaciones de edición permitidas son las siguientes: (a) insertar una letra, (b) borrar una letra, y (c) cambiar a una letra por otra. Utilizamos la notación $d(u, v)$ para referirnos a la distancia entre u y v. Por ejemplo, si $u =$ horda y $v =$ ondas, entonces $d(u, v) = 3$ (borramos la h, reemplazamos r por n, y agregamos s al final).

a) ¿Cuál es el valor de $d(u, v)$ cuando u o v corresponden a la palabra vacía?

b) Asuma que ni u ni v corresponden a la palabra vacía, es decir, $u = a_1 \ldots a_m$ y $v = b_1 \ldots b_n$, donde $m, n > 0$ y los a_i's y b_j's son letras. Defina $u' = a_1 \ldots a_{m-1}$ y $v' = b_1 \ldots b_{n-1}$. Demuestre que:

$$d(u, v) \;=\; \text{mín}\,\{1 + d(u', v),\, 1 + d(u, v'),\, \text{diff}(a_m, b_n) + d(u', v')\},$$

donde $\text{diff}(a_m, b_n) = 1$ si $a_m \neq b_n$ y $\text{diff}(a_m, b_n) = 0$ en caso contrario.

Ayuda: Represente la transformación de u a v de la siguiente forma: Si la letra a en la posición i no fue tocada, entonces ponemos el símbolo (a, a) en la posición i; si la letra a en la posición i fue reemplazada por b, entonces ponemos el símbolo (a, b) en la posición i; si la letra a en la posición i fue eliminada, entonces ponemos el símbolo $(a, \_)$ en la posición i; finalmente, si el símbolo a fue insertado en la posición j, entonces ponemos el símbolo $(\_, a)$ en la posición j. Por ejemplo, la transformación de $u =$ horda a $v =$ ondas arriba descrita es representada por la palabra $(\text{h}, \_)(\text{o}, \text{o})(\text{r}, \text{n})(\text{d}, \text{d})(\text{a}, \text{a})(\_, \text{s})$. Analice los diferentes casos que pueden ocurrir con respecto al símbolo que aparece en la última posición de la palabra que representa la transformación de u a v.

c) Utilizando lo anterior, diseñe un algoritmo recursivo eficiente que calcule $d(u, v)$ y encuentre una secuencia mínima de operaciones necesarias para convertir a u en v.

22. Sea R_n el número de regiones que se generan en la superficie de una esfera al ser dividida por n grandes círculos (es decir por n planos que pasan por el centro de la esfera), asumiendo que no hay tres de estos grandes círculos que pasen por el mismo punto.

- Explique por qué $R_{n+1} = R_n + 2n$, para $n \geq 1$.
- Resuelva iterativamente la expresión anterior.

23. Sea π una permutación de $\{1, \ldots, n\}$. Decimos que π es un *desordenamiento* si $\pi(i) \neq i$, para todo $1 \leq i \leq n$.

a) Demuestre que si D_n denota el número de desordenamientos sobre $\{1, \ldots, n\}$, entonces $D_n = (n-1)(D_{n-1} + D_{n-2})$. (*Hint:* El factor $n-1$ aparece debido a que hay $n-1$ posibilidades de elegir $\pi(1)$ en un desordenamiento π de $\{1, \ldots, n\}$).

b) Utilizando lo anterior, demuestre que $D_n = nD_{n-1} + (-1)^n$, para todo $n \geq 2$.

24. Conjeture una fórmula que determine

$$\sum_{k=1}^{n} \frac{1}{k(k+1)} \;=\; \frac{1}{1 \cdot 2} + \frac{1}{2 \cdot 3} + \cdots + \frac{1}{n \cdot (n+1)} \qquad (n \geq 1).$$

Demuestre por inducción que esta fórmula es correcta.

25. Sea $f : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+$ una función tal que $f(1,1) = 2$ y para todo $m, n \geq 1$ se tiene que

$$\begin{cases} f(m+1, n) &= f(m,n) + 2(m+n) \\ f(m, n+1) &= f(m,n) + 2(m+n-1) \end{cases}$$

Demuestre por inducción que $f(m,n) = (m+n)^2 - (m+n) - 2n + 2$, para todo $m, n \geq 1$.

26. Sea $p \geq 2$ primo. Asuma que $\sqrt{p}$ es racional, es decir que $\sqrt{p} = \frac{a}{b}$, para $a, b \in \mathbb{N}^+$.

a) Demuestre que existen $a', b' \in \mathbb{N}^+$ tales que $b' < b$ y $\sqrt{p} = \frac{a'}{b'}$.

b) Utilizando lo anterior y el principio del buen orden concluya que $\sqrt{p}$ no es racional.

Capítulo 3

Conteo, Probabilidades, y Relaciones de Recurrencia

La combinatoria es la rama de la matemática que estudia los arreglos de los objetos. Un área fundamental de la combinatoria es la enumeración o conteo, que nos permite determinar cuántos arreglos de un cierto tipo existen. En este capítulo estudiaremos el problema de conteo en detalle. Comenzaremos con sus principios básicos, incluyendo las reglas de suma y producto, y el principio del palomar. Luego veremos aplicaciones de tales herramientas en probabilidades y en el análisis del costo promedio de algoritmos. Finalmente estudiaremos el uso de relaciones de recurrencia para resolver problemas sofisticados de conteo, y presentaremos métodos para su resolución en casos particulares.

3.1. Técnicas básicas de conteo

3.1.1. Reglas de suma y producto

La reglas esenciales del conteo son las de la suma y el producto, detalladas a continuación:

- *Regla de la suma:* Si una tarea T puede ser llevada a cabo en paralelo por dos procesos P_1 y P_2, tal que hay m maneras de llevar a cabo P_1, hay n maneras de llevar a cabo P_2, y hay p maneras de llevar a cabo tanto P_1 como P_2, entonces T puede ser llevado a cabo de $n + m - p$ maneras.
- *Regla del producto:* Si T puede ser llevada a cabo en serie por dos procesos P_1 y P_2, tal que hay m maneras de llevar a cabo P_1 y n maneras de llevar a cabo P_2, entonces hay mn maneras de llevar a cabo T.

Ejemplo 3.1.1. Las *palabras binarias* son aquellas que se definen sobre el alfabeto $\{0, 1\}$. El número de palabras binarias de largo 7 es 2^7 por regla del producto. Combinando esta regla con la de la suma obtenemos que el número de palabras binarias de largo 7 que comienzan con 0 o terminan con 11 es $2^6 + 2^5 - 2^4$. □

Ejercicio. *¿Cuántas palabras binarias de largo 7 contienen 3 0's consecutivos o 4 1's consecutivos?*

3.1.2. Principio del palomar

El principio del palomar establece que si $(k + 1)$ objetos se insertan en k cajas, entonces al menos una caja debe contener al menos dos objetos.

Ejemplo 3.1.2. Sean $(x_1, y_1, z_1), \ldots, (x_9, y_9, z_9)$ elementos en $\mathbb{Z}^3$. Entonces el *punto medio* $(\frac{x_i+x_j}{2}, \frac{y_i+y_j}{2}, \frac{z_i+z_j}{2})$ de al menos un par de estos puntos debe tener todas sus coordenadas en $\mathbb{Z}$. En efecto, para que esto suceda se debe cumplir que x_i tiene la misma paridad que x_j, y similarmente y_i e y_j, y z_i y z_j. Note que hay

$2^3 = 8$ posibilidades de paridad para las coordenadas de los elementos (x, y, z) en $\mathbb{Z}^3$. Dado que nuestra entrada consiste de 9 puntos en $\mathbb{Z}^3$, por el principio del palomar obtenemos que dos de ellos deben tener la misma paridad coordenada a coordenada. El punto medio de tales puntos debe, por tanto, tener todas sus coordenadas en $\mathbb{Z}$. □

El principio del palomar puede naturalmente extenderse de la siguiente forma:

Proposición 3.1.1. *Si n objetos se colocan en k cajas, entonces al menos una caja debe contener al menos* $\lceil \frac{n}{k} \rceil$ *objetos.*

Demostración. En efecto, asuma por contradicción que toda caja contiene a lo más $\lceil \frac{n}{k} \rceil - 1$ objetos. Luego, el número de objetos en las cajas es a lo más:

$$k \cdot (\lceil \frac{n}{k} \rceil - 1) \quad < \quad k \cdot \frac{n}{k} = n.$$

Esto es una contradicción. □

Ejemplo 3.1.3. Entre 100 personas hay al menos $\lceil \frac{100}{12} \rceil = 9$ personas que nacieron el mismo mes. □

Ejercicio. *¿Cuál es el menor número de cartas que es necesario tomar desde un mazo para asegurarnos de que al menos 3 tienen la misma pinta?*

Ejemplo 3.1.4. Sea $A = a_1, \ldots, a_{n^2+1}$ una secuencia de números enteros distintos. Por ejemplo:

$$-3, \quad 9, \quad 11, \quad 4, \quad 6, \quad 7, \quad -2, \quad -8, \quad 0, \quad -6$$

es una de tales secuencias para $n = 3$. Una *subsecuencia estrictamente creciente de A* es una secuencia estrictamente creciente que se obtiene al eliminar de A cero o más de sus elementos. Por ejemplo, 4, 6, 7 es una subsecuencia estrictamente creciente de la secuencia que mostramos arriba. Análogamente podemos definir la noción de subsecuencia estrictamente decreciente de A.

Demostraremos a continuación que toda secuencia $A = a_1, \ldots, a_{n^2+1}$ tiene una subsecuencia estrictamente creciente o una subsecuencia estrictamente decreciente de largo $n + 1$. Para ello, con cada elemento a_i de la secuencia, donde $1 \leq i \leq n^2 + 1$, asociamos un par (c_i, d_i) tal que c_i es el largo de la subsecuencia estrictamente creciente de A más larga que comienza en a_i, y análogamente d_i es el largo de la subsecuencia estrictamente decreciente de A más larga que comienza en a_i. Por ejemplo, para la secuencia anterior tenemos que $(c_2, d_2) = (2, 4)$ y $(c_4, d_4) = (3, 3)$. Note que por definición, $1 \leq c_i, d_i \leq n^2 + 1$, para cada $1 \leq i \leq n^2 + 1$.

Asumamos, por contradicción, que no existe subsecuencia estrictamente creciente o subsecuencia estrictamente decreciente de largo $n + 1$ en A. En particular, se debe cumplir entonces que $1 \leq c_i, d_i \leq n$ para cada $1 \leq i \leq n^2 + 1$. Por esto, el número de pares distintos de la forma (c_i, d_i) que pueden aparecer en A es a lo más n^2. Por tanto, dado que A tiene $n^2 + 1$ elementos, deben existir posiciones $1 \leq i < j \leq n^2 + 1$ tal que $(c_i, d_i) = (c_j, d_j)$. Supongamos que $a_i < a_j$. Entonces $c_i > c_j + 1$, lo que es una contradicción. En caso contrario, es decir si $a_i > a_j$, se tiene que $d_i > d_j + 1$, lo que nuevamente es una contradicción. □

Ejercicio. *En una fiesta con 6 personas, todo par de personas es mutuo amigo o enemigo. Demuestre que existen al menos tres personas que son todas amigas entre sí, o todas enemigas entre sí.*

3.1.3. Permutaciones y combinaciones

¿De cuántas formas podemos alinear a 3 personas entre 10 posibles? Es posible resolver este problema mediante la regla del producto, Dado que tenemos 10 posibilidades para la primera persona en la alineación, 9 para la segunda, y 8 para la tercera, tenemos $10 \cdot 9 \cdot 8$ posibilidades en total. En general los ordenamientos (o alineaciones) de r objetos entre n posibles, para $1 \leq r \leq n$, se llaman *r-permutaciones de un conjunto de n objetos*. Abstrayendo de nuestro ejemplo anterior podemos fácilmente demostrar lo siguiente:

Proposición 3.1.2. *El número $P(n,r)$ de r-permutaciones de un conjunto de n elementos corresponde a:*

$$n \cdot (n-1) \cdot \cdots \cdot (n-r+1) \quad = \quad \frac{n!}{(n-r)!}.$$

En las permutaciones nos importa el orden en que los r objetos seleccionados son alineados. Existen otro tipo de problemas en los cuales tal orden es irrelevante. Por ejemplo, suponga que queremos seleccionar un comité de 3 personas entre 10 miembros. ¿De cuántas formas podemos hacer esto? La respuesta es $\frac{P(10,3)}{3!} = \frac{10 \cdot 9 \cdot 8}{3 \cdot 2 \cdot 1}$. En efecto, hay una relación uno-a-uno entre las 3-permutaciones de nuestro conjunto de 10 miembros y los ordenamientos de los posibles comités de 3 personas que podemos formar en él. El resultado se sigue ahora del hecho de que cada comité de tres personas puede ser ordenado de 3! maneras y el número de 3-permutaciones de nuestro conjunto de 10 elementos es $P(10,3) = 10 \cdot 9 \cdot 8$.

De forma más general, si $0 \leq r \leq n$ entonces una *r-combinación de un conjunto de n elementos* es una elección de r elementos distintos desde el conjunto. Abstrayendo de nuestro ejemplo anterior podemos fácilmente demostrar lo siguiente:

Proposición 3.1.3. *El número $C(n,r)$ de r-combinaciones de un conjunto de n elementos corresponde a:*

$$\frac{P(n,r)}{r!} \quad = \quad \frac{n!}{r! \cdot (n-r)!}.$$

Note, por tanto, que $C(n,r)$ corresponde precisamente con el coeficiente binomial $\binom{n}{r}$. Ocuparemos indistintamente ambas notaciones de ahora en adelante. Recuerde que $\binom{n}{r}$ denota el número de subconjuntos de tamaño r de un conjunto de tamaño n. Es fácil entonces demostrar lo siguiente:

Proposición 3.1.4. *Se cumple que:*

- *Si* $0 \leq r \leq n$, *entonces* $\binom{n}{r} = \binom{n}{n-r}$.
- $\sum_{k=0}^{n} \binom{n}{k} = 2^n$.

Ejemplo 3.1.5. En este ejemplo utilizaremos lo que hemos visto hasta ahora para demostrar que el número de palabras binarias de largo n que contienen exactamente 2 ocurrencias de la subpalabra 01 es $\binom{n+1}{5}$.

Primero, note que tales palabras son de la forma:

$$1 \ldots 10 \ldots \underline{01} \ldots 10 \ldots \underline{01} \ldots 10 \ldots 0,$$

donde la primera corrida de 1's y la última corrida de 0's podrían no existir. El subrayado denota las dos ocurrencias de la palabra 01. Observe que para "codificar" esta palabra nos basta especificar las posiciones $i_1 < i_2 < i_3 < i_4 < i_5$ tal que i_1 es la posición donde termina la primera fila de 1's; i_2 es la posición donde termina la siguiente corrida de 0's; i_3 es la posición donde termina la siguiente corrida de 1's; i_4 es la posición donde termina la siguiente corrida de 0's; e i_5 es la posición donde termina la última corrida de 1's. Dado que la primera corrida de 1's podría no existir, es posible que $i_1 = 0$. De igual forma, dado que la última corrida de 0's podría no existir, es posible que $i_5 = n$. Por ejemplo, si la palabra es 1100101 entonces tales posiciones son $2, 4, 5, 6, 7$, y si la palabra es 00001000111110000 entonces tales posiciones son $0, 4, 5, 8, 13$.

Cuando nos referimos a "codificar" tal palabra por medio de las posiciones $0 \leq i_1 < i_2 < i_3 < i_4 < i_5 \leq n$ descritas arriba, lo que queremos decir es que hay una relación uno-a-uno entre las palabras binarias de largo n que contienen exactamente 2 ocurrencias de la subpalabra 01 y las tuplas de la forma $(i_1, i_2, i_3, i_4, i_5)$ tal que $0 \leq i_1 < i_2 < i_3 < i_4 < i_5 \leq n$. Es decir, para cada palabra binaria de largo n que contiene exactamente 2 ocurrencias de la subpalabra 01 existe una tupla $(i_1, i_2, i_3, i_4, i_5)$, con $0 \leq i_1 < i_2 < i_3 < i_4 < i_5 \leq n$, que describe las posiciones arriba descritas. Por otro lado, para cada tupla $(i_1, i_2, i_3, i_4, i_5)$, con $0 \leq i_1 < i_2 < i_3 < i_4 < i_5 \leq n$, existe una, y solo una, palabra w de largo n que contiene exactamente 2 ocurrencias de la subpalabra 01 tal que $(i_1, i_2, i_3, i_4, i_5)$ describe las posiciones arriba descritas para w.

Por tanto, el número de palabras que queremos contar es precisamente el número de tuplas $(i_1, i_2, i_3, i_4, i_5)$, con $0 \leq i_1 < i_2 < i_3 < i_4 < i_5 \leq n$. Pero esto corresponde precisamente al número de formas de elegir 5 elementos distintos entre $\{0, \ldots, n\}$; es decir, el número de formas de elegir 5 elementos distintos entre $n+1$ posibles. Sabemos que esto corresponde a $\binom{n+1}{5}$. □

Ejemplo 3.1.6. Recuerde que el Teorema del Binomio establece que:

$$(x+y)^n = \sum_{0\leq k\leq n} \binom{n}{k} x^k y^{n-k}.$$

No es difícil, pero sí engorroso, demostrar algebraicamente este resultado (se puede hacer por inducción en $n \geq 1$). Acá mostraremos una simple "demostración combinatorial" del Teorema.

Expresemos $(x+y)^n$ como:

$$\underbrace{(x+y)(x+y)\cdots(x+y)}_{n \text{ veces}}. \tag{3.1}$$

Luego, las expresiones que aparecen en este producto son de la forma $x^k y^{n-k}$, para $0 \leq k \leq n$. Cada uno de esas expresiones se obtiene de haber elegido a x en k de los factores de la expresión (3.1) y a y en los $n-k$ restantes. Los k factores donde seleccionamos x pueden ser elegidos de $\binom{n}{k}$ maneras. Al tomar la suma de estas posibilidades para todo $0 \leq k \leq n$ obtenemos precisamente el Teorema del Binomio. □

Ejercicio. *Demuestre combinatorialmente los siguientes resultados*

- $\binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1}$.
- $\sum_{k=1}^{n} k \cdot \binom{n}{k} = n \cdot 2^{n-1}$.

Ejercicios Finales

1. Demuestre que $\sum_{k=1}^{n} k\binom{n}{k}^2 = n\binom{2n-1}{n-1}$ (*Hint:* Utilice una demostración combinatorial).
2. Demuestre que en toda fiesta con n personas debe haber dos personas que conozcan a la misma cantidad de gente.
3. Tome 11 enteros distintos $\{a_1, \ldots, a_{11}\}$ entre 1 y 100. Demuestre que existen dos subconjuntos A y B de $\{a_1, \ldots, a_{11}\}$ tales que (1) A y B son no vacíos, (2) $A \cap B = \emptyset$, y (3) $\sum_{a\in A} a = \sum_{b\in B} b$.
4. La idea de este ejercicio es contar el número de *caminos* en el plano desde el origen $(0,0)$ hasta el punto (m,n), donde m, n son naturales positivos. Cada camino consta de una serie de *pasos*, donde cada paso corresponde a una movida de una unidad hacia arriba o hacia la derecha. Por ejemplo, la secuencia

   $$(0,0),(0,1),(1,1),(1,2),(2,2),(3,2)(4,2),(4,3),(5,3)$$

   es un camino desde el origen hasta el punto $(5,3)$.

   ¿Cuántos distintos caminos existen en el plano desde el origen $(0,0)$ hasta el punto (m,n)?
5. Demuestre que en toda secuencia de m enteros existe uno o más términos consecutivos cuya suma es divisible por m.
6. ¿Cuántas formas hay de elegir r billetes desde un banco que posee n tipos distintos de billetes? Fundamente su respuesta. (Asuma que el orden en que se eligen los billetes no importa, que los billetes de cada tipo son indistinguibles, y que existen al menos r billetes de cada tipo).
7. ¿De cuántas formas se pueden repartir n monedas de un peso entre k niños, si

   a) Cada niño debe recibir al menos una moneda?

   b) Permitimos que algunos niños no reciban monedas?
8. ¿Cuántas soluciones tiene la ecuación $x_1 + x_2 + x_3 = 25$, si x_1, x_2 y x_3 son enteros no negativos?

9. Utilice el principio del palomar para demostrar lo siguiente: Si se eligen 5 puntos cualesquiera en el interior de un cuadrado de lado 2, entonces existen dos de ellos cuya distancia es a lo más $\sqrt{2}$.

10. Asuma que los números del 1 al 36 han sido distribuidos en una ruleta (es decir, circularmente). Demuestre que deben existir tres números consecutivos en esta distribución cuya suma es al menos 55.

11. Un *desorden* es una permutación de un conjunto de objetos que no deja a ningún objeto en su lugar original. ¿Cuál es el número de desordenes de un conjunto con n elementos?

12. Sean $a_1, \ldots, a_{n+1}$ enteros positivos cualesquiera en el intervalo $\{1, \ldots, 2n\}$ $(n \geq 1)$. Demuestre que existen $1 \leq i, j \leq n+1$ tal que $i \neq j$ y a_i divide a a_j.

13. Sean n y k enteros positivos tal que $1 \leq k \leq n$. Demuestre lo siguiente:

$$\sum_{k=1}^{n} \binom{n}{k}\binom{n}{k-1} \;=\; \binom{2n+2}{n+1}/2 - \binom{2n}{n}.$$

14. ¿Cuántas palabras de largo n sobre el alfabeto $\Sigma = \{a_1, \ldots, a_p\}$ tienen exactamente $0 \leq n_i \leq n$ ocurrencias de la letra a_i $(1 \leq i \leq p)$?

15. Demuestre que si n es un entero positivo, entonces

$$(x_1 + x_2 + \cdots + x_p)^n \;=\; \sum_{n_1+n_2+\cdots+n_p=n} T(n;(n_1,\ldots,n_p))x_1^{n_1}x_2^{n_2}\cdots x_p^{n_p},$$

asumiendo que $T(n;(n_1,n_2,\ldots,n_p)) = \frac{n!}{n_1!n_2!\cdots n_p!}$.

16.

17. Sea $x \in \mathbb{R}$ y n un entero positivo mayor que 1. Demuestre que existe entero j tal que $1 \leq j \leq n$ y el valor absoluto de la distancia de jx al entero más cercano a jx es menor que $1/n$.

18. Lanzamos 50 tiros a un objetivo cuadrado de 70 cms por lado. Todos nuestros tiros dan en el objetivo. Demuestre que existen dos tiros que están a una distancia menor a 15cms.

19. Demuestre combinatorialmente la siguiente igualdad:

$$\sum_{k=0}^{n} \binom{n}{k} \cdot \binom{k}{m} \;=\; \binom{n}{m} \cdot 2^{n-m}.$$

20. Suponga que j_n $(n \geq 1)$ cuenta el número de formas en que se puede subir una escalera de n peldaños si se puede avanzar de uno o dos peldaños a la vez. Defina la sucesión $j_1, j_2 \ldots$ en términos de la sucesión de Fibonacci $f_1, f_2, \ldots$.

21. Sean $a_1, a_2, \ldots, a_n$ números naturales. Demuestre que existe un subconjunto no vacío de estos números cuya suma es divisible por n.

3.2. Probabilidades discretas

Son bien conocidas las aplicaciones de las probabilidades para analizar el comportamiento de los juegos de azar o los resultados de ciertos experimentos. Las probabilidades se ocupan, además, extensamente en áreas como la física o la biología. En el área que nos atañe más directamente, la ciencia de la computación, las probabilidades tienen diversas e importantes aplicaciones. Estas incluyen, por ejemplo, el análisis del costo promedio de ciertos algoritmos y la obtención de cotas para estimar la confianza de algunos métodos en el área del aprendizaje computacional (*machine learning*).

3.2.1. Conceptos básicos

Sea S un conjunto de posibles resultados de un experimento. Asumiremos siempre que S es finito. Un *evento* E es un subconjunto de S. Si todos los resultados del experimento en S son igualmente probables, entonces la *probabilidad de* E, denotada $Pr(E)$, se define como $\frac{|E|}{|S|}$. Es fácil demostrar entonces lo siguiente:

Proposición 3.2.1. *1. La probabilidad de la unión de dos eventos E_1 y E_2 corresponde a $Pr(E_1) + Pr(E_2) - Pr(E_1 \cap E_2)$.*

2. La probabilidad del complemento $\bar{E} = E \setminus S$ del evento E corresponde a $1 - Pr(E)$

Ejemplo 3.2.1. A continuación analizaremos la probabilidad de varios eventos referidos al juego del póker. Asumimos que una mano de póker corresponde a la elección de 5 cartas entre las 52 del mazo. Por tanto, existen $\binom{52}{5}$ manos de póker posibles. El mazo tiene 4 cartas de cada tipo, una por cada pinta:

1. La probabilidad de que una mano de póker contenga 4 cartas del mismo tipo es:

$$\frac{13 \cdot 48}{\binom{52}{5}}.$$

   En efecto, cada una de esas manos se determina exclusivamente por el tipo que aparece 4 veces, para el cual tenemos 13 posibilidades, y luego por la carta restante, para la cual tenemos 48 posibilidades.

2. La probabilidad de que una mano de póker contenga dos cartas de un tipo y tres de otro es:

$$\frac{P(13,2) \cdot \binom{4}{2} \cdot \binom{4}{3}}{\binom{52}{5}}.$$

   La razón es que cada una de esas manos se determina únivocamente por los tipos que aparecen con 2 y 3 cartas en la mano, respectivamente, los que pueden ser elegidos de $P(13, 2)$ maneras, además de las cartas que elegimos para representar a cada uno de esos tipos. Para el primer tipo podemos elegir las dos cartas de $\binom{4}{2}$ maneras, mientras que para el segundo podemos hacerlo de $\binom{4}{3}$ maneras.

3. Siguiendo con el mismo tipo de razonamiento podemos determinar que la probabilidad de que la mano de póker sea un *color*, es decir, que todas las cartas sean de la misma pinta, es

$$\frac{\binom{4}{1} \cdot \binom{13}{5}}{\binom{52}{5}}.$$

   La probabilidad de que sea una *escalera*, es decir, que la mano consista de tipos consecutivos asumiendo que A es el primer tipo y 10 es el último, corresponde a

$$\frac{\binom{10}{1} \cdot 4^5}{\binom{52}{5}}.$$

   Finalmente, la probabilidad de que la mano sea una escalera de color es

$$\frac{10 \cdot 4}{\binom{52}{5}}.$$

4. La probabilidad de que la mano consista solo de tipos distintos, pero no sea ni escalera ni color, puede entonces determinarse como sigue. Considere los siguientes eventos:

   - A son las manos que solo consisten de tipos distintos.
   - B son las manos que son escaleras.

- C son las que son color.

Por tanto, la probabilidad que buscamos corresponde a:

$$Pr(A) - Pr(B \cup C) \;=\; Pr(A) - Pr(B) - Pr(C) + Pr(B \cap C).$$

Note que $Pr(B)$, $Pr(C)$, y $Pr(B \cap C)$ fueron calculadas en el punto anterior, por lo que solo falta calcular $Pr(A)$. No es difícil observar que esta corresponde a $\frac{\binom{13}{5}\cdot 4^5}{\binom{52}{5}}$.

□

Ejemplo 3.2.2. El problema de *Monty Hall* corresponde a una famosa "paradoja" de las probabilidades básicas. Se dice que es una paradoja ya que su solución parece poco intuitiva a mucha gente.

En el problema un anfitrión ofrece a una persona abrir una de tres puertas y llevarse como premio lo que hay detrás de tal puerta. Se sabe que una de las puertas esconde un auto, mientras que las otras dos esconden una cabra. El objetivo de la persona es llevarse el automóvil. Inicialmente esta elige una puerta, digamos A. Antes de mostrar que hay detrás de A el anfitrión abre una de las otras puertas, B o C, donde sabe que está a la cabra. A partir de esto le pregunta a la persona que elige la puerta si quiere cambiar su elección. El problema es si cambiar la elección inicial a esta altura aumenta o simplemente mantiene las probabilidades de que la persona gane el auto.

Uno podría razonablemente suponer que la probabilidad no aumenta. De hecho, sabemos que el auto está en una de las otras dos puertas con igual probabilidad, por lo que cambiar de elección no tendría por qué aumentar nuestra probabilidad de elegir el auto. Sin embargo, un análisis matemático más detallado muestra por el contrario que esto sí ocurre (esto es lo que sorprende a mucha gente). En efecto, ¿cuál es la probabilidad de haber elegido la puerta correcta de entrada? Tan solo $1/3$. Por otro lado, la probabilidad de elegir correctamente al cambiar de puerta luego de que el anfitrión abre una de las puertas en las que hay una cabra es igual a la probabilidad de haber elegido incorrectamente de entrada; es decir, $2/3$. □

Ejemplo 3.2.3. A continuación presentamos otra paradoja interesante de las probabilidades, llamada *la paradoja del cumpleaños*. Esta muestra que no hace falta tener un grupo muy grande de gente para que la probabilidad de que dos personas del grupo tengan el mismo cumpleaños sea muy alta.

Suponga que tenemos un grupo de n personas. ¿Cuál es la probabilidad de que dos tengan el mismo cumpleaños? Para analizar esto es más fácil analizar la probabilidad del complemento de este evento, es decir, que no hayan dos personas con el mismo cumpleaños. Note que el número de combinaciones de cumpleaños posibles para estas personas es 366^n. Por otro lado, de estas combinaciones exactamente $P(366, n)$ asignan cumpleaños distintos a todas estas personas. Por tanto, la probabilidad de que no hayan dos personas con el mismo cumpleaños es $\frac{P(366,n)}{366^n}$, y de que sí las hayan es $X(n) = 1 - \frac{P(366,n)}{366^n}$. Es posible obtener el valor de $X(n)$ mediante métodos numéricos. Por ejemplo, el menor número n de personas que hacen a $X(n) \geq 0,5$ es 23. Es decir, con solo 23 personas ya hay un 50 % de probabilidades de que dos tengan el mismo cumpleaños. Con $n = 40$ tal probabilidad llega a 89 %, y con $n = 50$ a 97 %. □

Distribuciones de probabilidad En general, por supuesto, no todo resultado $s \in S$ del experimento tiene la misma probabilidad de ocurrir. Por ejemplo, la moneda o el dado pueden estar cargados, o las personas pueden nacer con mayor probabilidad en algunos meses que en otros. Esto lo representamos mediante una *distribución de probabilidad* $p : S \to [0, 1]$, que asigna una probabilidad de ocurrir $p(s)$ a cada resultado $s \in S$ del experimento, de tal forma que $\sum_{s\in S} p(s) = 1$. El caso particular visto antes en que todos los eventos son igualmente probables corresponde entonces a la distribución de probabilidad que cumple $p(s) = \frac{1}{|S|}$ para cada $s \in S$.

La probabilidad de un evento $E \subseteq S$ se define en este caso como $Pr(E) := \sum_{s\in E} p(s)$. Es fácil demostrar que, al igual que antes, se cumple que:

$$Pr(E_1 \cup E_2) = Pr(E_1) + Pr(E_2) - Pr(E_1 \cap E_2) \quad \text{y} \quad Pr(\bar{E}) = 1 - Pr(E).$$

Ejercicio. *En un dado cargado el número 3 sale dos veces más seguido que cualquier otro número. ¿Cuál es la probabilidad de que al lanzar el dado salga un número impar?*

El método probabilista Mediante las probabilidades podemos demostrar resultados de existencia de un objeto x con cierta propiedad P. Para eso demostramos que la probabilidad de que un elemento no tenga la propiedad P es estrictamente menor que 1, y por tanto que debe existir un x tal que $P(x)$ es cierto. La demostración es *no constructiva* en muchos casos, puesto que solo nos dice que tal x existe, pero no nos dice qué objeto es ni cómo construirlo.

Ejemplo 3.2.4. Sea $k \geq 3$ un entero, y defina $R(k)$ como el menor número necesario de personas que debe haber en una reunión para asegurarnos que haya k personas que sean mutuamente amigas o mutuamente enemigas (asumiendo que cualesquiera dos personas deben ser amigas o enemigas). Demostraremos que $R(k) \geq 2^{\frac{k}{2}}$ usando el método probabilista.

Para $k = 3$ esto se puede demostrar por simple inspección. Asuma entonces que $k \geq 4$. Consideremos una fiesta con n personas. Luego, el número de grupos de k personas es $\binom{n}{k}$. Sean $S_1, \ldots, S_{\binom{n}{k}}$ esos grupos. Denotamos por E_i el evento de que el grupo S_i consiste solo de mutuos amigos o solo de mutuos enemigos. La probabilidad de que en la fiesta exista tal grupo es $Pr(E_1 \cup \cdots \cup E_{\binom{n}{k}})$. Por definición tenemos que:

$$Pr(E_1 \cup \cdots \cup E_{\binom{n}{k}}) \ \leq \ \sum_{i=1}^{\binom{n}{k}} Pr(E_i).$$

Por otro lado, $Pr(E_i) = Pr(F_1) + Pr(F_2)$, donde F_1 corresponde al evento en que todos los miembros del grupo S_i son mutuos amigos, y F_2 al evento en que son todos mutuos enemigos. Ahora,

$$Pr(F_1) \ = \ Pr(F_2) \ = \ \frac{1}{2}^{\binom{k}{2}},$$

ya que hay $\binom{k}{2}$ pares de personas distintas en S_i. En conclusión:

$$Pr(E_1 \cup \cdots \cup E_{\binom{n}{k}}) \ \leq \ \sum_{i=1}^{\binom{n}{k}} Pr(E_i) \ \leq \ 2\sum_{i=1}^{\binom{n}{k}} \frac{1}{2}^{\binom{k}{2}} \ = \ 2\binom{n}{k}\frac{1}{2}^{\binom{k}{2}}.$$

Ahora, es fácil comprobar que $\binom{n}{k} \leq \frac{n^k}{2^{k-1}}$, por lo que:

$$2\binom{n}{k}\frac{1}{2}^{\binom{k}{2}} \ \leq \ 2 \cdot \frac{n^k}{2^{k-1}} \cdot \frac{1}{2}^{\binom{k}{2}}.$$

Asumiendo entonces que $n < 2^{\frac{k}{2}}$, obtenemos que la probabilidad de que una reunión con n personas tenga un grupo de k personas que son mutuamente amigas o mutuamente enemigas es estrictamente menor a:

$$2 \cdot \frac{2^{\frac{k^2}{2}}}{2^{k-1}} \cdot \frac{1}{2}^{\frac{k(k-1)}{2}} \ = \ 2^{2-\frac{k}{2}} \ \leq \ 1.$$

La última desigualdad se cumple ya que $k \geq 4$. Esto quiere decir que la probabilidad de que no exista tal grupo es estrictamente mayor a 0, es decir, que existe una reunión de n personas en la que no hay grupo de k personas que son mutuamente amigas o mutuamente enemigas (¡aún cuando no sabemos cuál es tal reunión!). □

3.2.2. Variables aleatorias

Una *variable aleatoria* no es una variable ni tampoco es aleatoria, sino que es un función X que mapea el conjunto S de resultados del experimento en los reales $\mathbb{R}$; es decir, $X : S \to \mathbb{R}$. Por ejemplo, si nuestro experimento consiste en lanzar tres monedas, entonces X podría representar el número de caras obtenidas.

Si, por otro lado, el experimento consiste en lanzar dos dados, entonces X podría representar el valor de la suma de los resultados.

Note que cada variable aleatoria $X : S \to \mathbb{R}$ representa una distribución de probabilidad sobre los valores r de la forma $X(s)$, para $s \in S$. Tal distribución asigna a r el valor $Pr(X = r)$, el cual representa la probabilidad del evento referido a que X tome el valor r (bajo la distribución de probabilidad $p : S \to [0, 1]$ subyacente).

Ejemplo 3.2.5. Asuma que se lanzan tres monedas y X_c es la variable aleatoria que cuenta el número de caras obtenidas. Luego:

$$Pr(X_c = 3) = \frac{1}{8}; \quad Pr(X_c = 2) = \frac{3}{8}; \quad Pr(X_c = 1) = \frac{3}{8}; \quad Pr(X_c = 0) = \frac{1}{8}.$$

□

Sea $p : S \to [0, 1]$ una distribución de probabilidad sobre S. Usualmente estamos interesados en el *valor esperado* de una variable aleatoria $X : S \to \mathbb{R}$, denotado $E(X)$, el que representa el promedio ponderado del valor de X. Formalmente definimos:

$$E(X) := \sum_{s \in S} p(s) \cdot X(s).$$

La siguiente proposición establece una útil reformulación del valor esperado que nos permite calcular su valor cuando el número de salidas del experimento es muy grande:

Proposición 3.2.2. *Se cumple que:*

$$E(X) := \sum_{r = X(s),\ para\ s \in S} r \cdot Pr(X = r).$$

Demostración. Por definición, $E(X) = \sum_{s \in S} p(s) \cdot X(s)$. Pero para cada r de la forma $X(s)$, donde $s \in S$, se cumple que $Pr(X = r) = \sum_{s \in S, X(s) = r} p(s)$. Se sigue entonces que $E(X) = \sum_{\{r = X(s) | s \in S\}} r \cdot Pr(X = r)$. □

Ejemplo 3.2.6. Siguiendo con el Ejemplo 3.2.5 tenemos que:

$$E(X_c) = 0 \cdot \frac{1}{8} + 1 \cdot \frac{3}{8} + 2 \cdot \frac{3}{8} + 3 \cdot \frac{1}{8} = \frac{3}{2}.$$

□

La noción de valor esperado cumple algunas importantes propiedades de *linearidad* expresadas en la siguiente proposición:

Proposición 3.2.3. *Se cumple que:*

1. *Si $X_1, \ldots, X_n$ son variables aleatorias en S, entonces:*

$$E(X_1 + \cdots + X_n) = \sum_{i=1}^{n} E(X_i).$$

2. *Si X es una variable aleatoria en S y $a, b \in \mathbb{R}$, entonces:*

$$E(aX + b) = aE(X) + b.$$

Ejercicio. *Demuestre la Proposición 3.2.3.*

Las propiedades de linearidad antes expresadas son extremadamente importantes para poder resolver problemas más complejos relacionados con el valor esperado. A continuación vemos un ejemplo:

Ejemplo 3.2.7. A un restaurante entran n personas que dejan su sombrero en guardarropía. La persona a cargo del guardarropas olvida, sin embargo, marcar quién es el dueño de cada sombrero. Creyendo que es una buena solución, al llegar los clientes a reclamar sus sombreros simplemente los devuelve al azar. ¿Cuál es el valor esperado de sombreros que fueron bien devueltos?

Sea X una variable aleatoria que cuenta el número de sombreros entregados a la persona correcta. Por tanto, X es una variable aleatoria sobre el conjunto de permutaciones de los sombreros. Matemáticamente podemos representar tales permutaciones como aquellas funciones $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$, donde $\pi(i) = j$ representa el hecho de que la persona i recibió el sombrero de la persona j. Note que $X(\pi)$ corresponde entonces al número de elementos en $1 \leq i \leq n$ tal que $\pi(i) = i$. Estamos interesados en calcular $E(X)$. Sin embargo, no es fácil calcular tal valor directamente, por lo que ocupamos una técnica muy usual basada en descomponer nuestra variable X como la suma de variables para las cuales es más sencillo calcular su valor esperado, y luego aplicar las propiedades de linealidad.

En particular, en este caso descomponemos a X como $X_1 + \cdots + X_n$, donde X_i es la variable aleatoria definida como sigue:

$$X_i(\pi) := \begin{cases} 1, & \text{si el } i\text{-ésimo sombrero fue bien entregado, i.e., } \pi(i) = i, \\ 0, & \text{en caso contrario.} \end{cases}$$

Las variables aleatorias del tipo de X_i, que toman valor 0 o 1 solamente, se llaman *variables indicatorias*. Son particularmente útiles en el análisis ya que su valor esperado $E(X_i)$ corresponde precisamente a $Pr(X_i = 1)$ (debido a que $E(X_i) = \sum_{j \in \{0,1\}} j \cdot Pr(X_i = j) = Pr(X_i = 1)$), y por tanto es en algunos casos más fácil de calcular.

Por linearidad del valor esperado obtenemos entonces que:

$$E(X) \;=\; \sum_{1 \leq i \leq n} E(X_i) \;=\; \sum_{1 \leq i \leq n} Pr(X_i = 1).$$

A continuación analizamos cuál es el valor de $Pr(X_i = 1)$. Por definición, $X_i = 1$ solo si la permutación de los sombreros entrega el sombrero i-esimo a la persona correcta. Esto ocurre, precisamente, en aquellas permutaciones $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ donde $\pi(i) = i$. Es fácil ver que hay $(n-1)!$ permutaciones que satisfacen esto. Por otro lado, el número total de permutaciones es $n!$, por lo que $Pr(X_i = 1) = \frac{1}{n}$. Concluímos entonces que

$$E(X) \;=\; \sum_{1 \leq i \leq n} Pr(X_i = 1) \;=\; \sum_{1 \leq i \leq n} \frac{1}{n} \;=\; 1.$$

Es decir, el número esperado de personas que recibe bien su sombrero es tan solo una (¡independientemente de cuán grande sea n!). □

Ejemplo 3.2.8. Considere el conjunto de las permutaciones $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$. Sea X una variable aleatoria definida sobre tal conjunto, de tal forma que $X(\pi)$ corresponde al número de índices $1 \leq i \leq n$ tal que:

$$\pi(i) \;=\; \text{mín}\,\{\pi(1), \ldots, \pi(i)\}.$$

Calcularemos $E(X)$. Para eso descomponemos a X como $X_1 + \cdots + X_n$, donde X_i es la variable indicatoria que vale 1 si y solo si $\pi(i) = \text{mín}\,\{\pi(1), \ldots, \pi(i)\}$. Por linearidad sabemos que $E(X) = \sum_{1 \leq i \leq n} E(X_i)$, y por propiedades de las variables indicatorias concluimos entonces que $E(X) = \sum_{1 \leq i \leq n} Pr(X_i = 1)$. A continuación analizamos cuál es el valor de $Pr(X_i = 1)$.

¿Para cuántas permutaciones $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ se cumple que $X_i(\pi) = 1$? Un análisis basado en conteo permite demostrar que tal número es igual a $\binom{n}{i} \cdot (i-1)! \cdot (n-i)!$. Por otro lado, el número total de permutaciones es $n!$, por lo que:

$$Pr(X_i = 1) \;=\; \frac{\binom{n}{i} \cdot (i-1)! \cdot (n-i)!}{n!} \;=\; \frac{1}{i}.$$

Concluímos entonces que

$$E(X) \;=\; \sum_{1\leq i\leq n} Pr(X_i = 1) \;=\; \sum_{1\leq i\leq n} \frac{1}{i}.$$

Note que $\sum_{1\leq i\leq n} \frac{1}{i} \leq \int_1^n \frac{1}{x}dx + 1$, es decir, $E(X)$ es $\log n + 1$, que es $O(\log n)$. □

Independencia de variables aleatorias Dos variables aleatorias sobre S se dicen *independientes*, si para todo $r, s \in \mathbb{R}$ se cumple que:

$$Pr(X = r \text{ e } Y = s) \;=\; Pr(X = r) \cdot Pr(Y = s).$$

Ejemplo 3.2.9. Considere el lanzamiento de dos dados. Sean X_1 y X_2 las variables aleatorias que nos entregan el valor del primer y segundo dado, respectivamente. Entonces estas variables son independientes, ya que para cada $1 \leq i, j \leq 6$ se tiene que:

$$Pr(X_1 = i \text{ y } X_2 = j) \;=\; \frac{1}{36} \;=\; Pr(X_1 = i) \cdot Pr(X_2 = j).$$

Por otro lado, si definimos $X = X_1 + X_2$, entonces X_1 y X no son independientes ya que:

$$Pr(X_1 = 1 \text{ y } X = 12) \;=\; 0 \;\neq\; Pr(X_1 = 1) \cdot Pr(X = 12),$$

pues $Pr(X_1 = 1), Pr(X = 12) > 0$. □

Una importante propiedad es que el valor esperado del producto de variables aleatorias independientes corresponde al producto de sus valores esperados:

Proposición 3.2.4. *Si X e Y son variables aleatorias independientes sobre S, entonces:*

$$E(XY) \;=\; E(X) \cdot E(Y).$$

Ejercicio. *Demuestre la proposición.*

Análisis de costo promedio de algoritmos El valor esperado de una variable puede ser utilizada para entender cuál es el costo de un algoritmo en promedio. En algunos casos es posible demostrar que tal costo promedio es mejor que el costo de peor caso, lo que es una evidencia de que el algoritmo podría comportarse mal solo en algunas pocas entradas.

Ejemplo 3.2.10. El algoritmo de *bucketsort* ordena una lista $a_1, \ldots, a_n$ de elementos en el intervalo $[0, 1)$ colocando a tales elementos en *buckets* $B[0], \ldots, B[n-1]$, de tal forma que si $1 \leq i < j \leq n$ entonces los elementos colocados en el bucket $B[i]$ deben ser menores a los colocados en el bucket $B[j]$, y luego ordenando cada bucket con un algoritmo tradicional como *insertsort*. Formalmente, esto se realiza de la siguiente forma:

1. Para cada $1 \leq i \leq n$ se coloca al elemento a_i en el bucket $B(\lfloor na_i \rfloor)$. En particular, $B(0)$ contiene todos los elementos en $[0, \frac{1}{n})$, $B(1)$ a todos los elementos en $[\frac{1}{n}, \frac{2}{n})$, y así sucesivamente.

2. Luego se ordena cada bucket mediante insertionsort, lo que toma tiempo cuadrático en el número de elementos en el bucket.

3. La concatenación de los elementos ordenados en los buckets

   $$B[0], B[1], \ldots, B[n-1]$$

   se entrega como resultado.

Dado que en el peor caso todos los elementos de la entrada caerán en el mismo bucket, este algoritmo toma tiempo $\Theta(n^2)$ (incluso en el caso promedio). Sin embargo, en promedio la complejidad de Bucketsort es mucho menor, como demostramos a continuación.

Note que el costo del algoritmo en una entrada $a_1, \ldots, a_n$ es:

$$O(n) + \sum_{0 \leq i \leq n-1} O(Y_i^2),$$

donde Y_i es la variable aleatoria que representa el número de elementos en bucket i. En efecto, el término $O(n)$ representa el costo de colocar a cada elemento en su bucket, mientras que el término $O(Y_i^2)$ representa el costo de ordenar el bucket $B[i]$ utilizando insertionsort. Por ende, el valor esperado del costo del algoritmo es $E(O(n) + \sum_{0 \leq i \leq n-1} O(Y_i^2))$, el que corresponde a $O(n) + \sum_{i=0}^{n-1} O(E(Y_i^2))$ por linearidad. Para resolver esto analizamos a continuación el valor de $E(Y_i^2)$.

Primero descomponemos a Y_i como $\sum_{j=1}^{n} X_{ij}$, donde X_{ij} es la variable indicatoria que vale 1 si y solo si a_j cae en bucket i. Luego:

$$E(Y_i^2) \;=\; E((\sum_{j=1}^{n} X_{ij})^2) \;=\; E(\sum_{j=1}^{n} \sum_{k=1}^{n} X_{ij} \cdot X_{ik}).$$

Esto último puede a su vez expresarse por linearidad como:

$$\sum_{j=1}^{n} \sum_{k=1}^{n} E(X_{ij} X_{ik}),$$

lo que puede ser descompuesto de la siguiente forma:

$$\sum_{j=1}^{n} E(X_{ij}^2) \;+ \sum_{j=1}^{n} \sum_{\substack{k=1 \\ k \neq j}}^{n} E(X_{ij} X_{ik}).$$

Ahora calculamos cada uno de estos valores por separado:

- Dado que X_{ij} es una variable indicatoria, se tiene que:

  $$E(X_{ij}^2) = \sum_{i \in \{0,1\}} i \cdot Pr(X_{ij}^2 = i) = Pr(X_{ij}^2 = 1) = Pr(X_{ij} = 1).$$

  Pero cada elemento tiene la misma probabilidad de pertenecer a cada uno de los buckets, por lo que $E(X_{ij}^2) = Pr(X_{ij} = 1) = \frac{1}{n}$.

- Por otro lado, X_{ij} y X_{ik} son variables independientes cuando $k \neq j$, por lo que obtenemos lo siguiente a partir de la Proposición 3.2.4:

  $$E(X_{ij} X_{ik}) \;=\; E(X_{ij}) \cdot E(X_{ik}) \;=\; \frac{1}{n^2}.$$

Concluimos entonces que:

$$E(Y_i^2) \;=\; \sum_{j=1}^{n} \frac{1}{n} + \sum_{j=1}^{n} \sum_{\substack{k=1 \\ k \neq j}}^{n} \frac{1}{n^2} \;=\; n \cdot \frac{1}{n} + \frac{n(n-1)}{n^2} \;=\; 2 - \frac{1}{n}.$$

El costo esperado de nuestro algoritmo es por tanto:

$$O(n) + \sum_{i=0}^{n-1} O(E(Y_i^2)) \;=\; O(n) + \sum_{i=0}^{n-1} O(2 - \frac{1}{n}) \;=\; O(n).$$

Es decir, el costo promedio de bucketsort es lineal en el tamaño de la entrada. □

Ejercicios Finales

1. Tres parejas se sientan en una fila. Suponga que todos los ordenamientos son igualmente probables:

   a) Calcule la cantidad de ordenamientos distintos tales que todos los esposos queden sentados junto a sus respectivas esposas.

   b) Demuestre que la probabilidad que algún esposo se siente junto a su respectiva esposa es $2/3$.

2. Un ropero contiene $n > 0$ pares de zapatos. Si se escogen al azar $2r$ zapatos (con $2r < n$) ¿Cuál es la probabilidad de que:

   a) no haya ningún par completo;

   b) haya exactamente un par completo?

3. Considere la siguiente estrategia para buscar un elemento x en una lista A no ordenada: Elija un índice i al azar. Si el i-ésimo elemento de A es x, entonces terminamos. De otra forma, seguimos la búsqueda eligiendo un nuevo índice j al azar hasta que encontramos x. Note que en cada paso elegimos sobre el conjunto completo de índices, por lo que podríamos examinar un elemento más de una vez.

   a) Asuma que existe exactamente un índice i tal que la i-esima posición de A contiene a x. ¿Cuál es el número esperado de pasos que la estrategia realizará hasta encontrar x?

   b) ¿Qué sucede si hay $k \geq 1$ índices i que contienen a x?

   Hint: Recuerde que $\sum_{k=1}^{\infty} kx^k = \frac{x}{(1-x)^2}$ para $|x| < 1$.

4. Considere el problema de ordenar una lista de n elementos distintos $a_1, \ldots, a_n$. Utilizaremos un modelo de algoritmos basado en *árboles de decisión*. Un árbol de decisión es un árbol binario con raíz en la que cada nodo representa una comparación ¿$a_i < a_j$? entre dos elementos distintos de la lista (es decir, $i \neq j$). El resultado de tal comparación nos dice si debemos ir a la izquierda o a la derecha en el árbol. Cada uno de los dos subárboles de tal nodo representa, por tanto, una posible salida de la comparación ¿$a_i < a_j$? Además, cada uno de las hojas del árbol de decisión debe representar un posible ordenamiento de la lista. Por uniformidad además requerimos que para cada $n \geq 1$ sea el mismo árbol de decisión el que ordene cada una de las listas con n elementos.

   Por ejemplo, un posible árbol de decisión para ordenar listas a_1, a_2, a_3 con tres elementos es el siguiente. La raíz es la comparación ¿$a_1 < a_2$? Si la respuesta es SI se avanza al hijo izquierdo de la raíz que corresponde a la comparación ¿$a_2 < a_3$?, y si la respuesta es NO (es decir, $a_2 < a_1$) se avanza al hijo derecho de la raíz que corresponde a la comparación ¿$a_1 < a_3$? Si la respuesta a la comparación ¿$a_2 < a_3$? es SI entonces se avanza al hijo izquierdo de tal nodo que es una hoja que corresponde al ordenamiento $a_1 < a_2 < a_3$. Si la respuesta es NO entonces se avanza al hijo derecho que es una hoja que corresponde al ordenamiento $a_1 < a_3 < a_2$. Simétricamente se define el subárbol que cuelga del nodo correspondiente a la comparación ¿$a_1 < a_3$?

   Demuestre que cualquier algoritmo de ordenamiento de listas basado en árboles de decisión realiza $\Omega(n \log n)$ comparaciones.

   a) Demuestre que la altura promedio de una hoja en un árbol binario con n hojas es $\Omega(\log n)$.

   b) ¿Qué es posible entonces deducir con respecto al número promedio de comparaciones que un algoritmo de ordenamiento basado en árboles de decisión debe realizar?

5. Considere una *caminata aleatoria* en $\mathbb{Z}$ que comienza en el origen 0. En cada momento, el caminante tiene una probabilidad $0 \leq p \leq 1$ de moverse una unidad a la derecha y $1 - p$ de moverse una unidad a la izquierda.

a) Encuentre una expresión para la probabilidad $P(x, n)$ de que el caminante se encuentre en el punto x, con $x \geq 0$, luego de n pasos.

b) Utilice la *aproximación de Stirling* $y! \approx y^{y+\frac{1}{2}} e^{-y}\sqrt{2\pi}$ para demostrar que:

$$P(0, 2m) \approx (4p(1-p))^m/\sqrt{m\pi}.$$

c) ¿Qué sucede con $P(0, 2m)$ cuando $m \to \infty$? (Piense cuál es el máximo valor que puede tomar $p(1-p)$).

6. Considere un grupo de n personas en que cada par de personas es mutuamente amigo o enemigo. Sea E el evento de que exista un grupo de k mutuos amigos o k mutuos enemigos, para un cierto $1 \leq k \leq n$. Demuestre que:

$$Pr(E) \leq \binom{n}{k} \cdot 2(\frac{1}{2})^{\binom{k}{2}}.$$

7. Sea $a_1, \ldots, a_n$ una permutación generada uniformemente al azar de los números $\{1, \ldots, n\}$. Una *inversión* en $a_1, \ldots, a_n$ es un par (i, j) tal que $1 \leq i < j \leq n$ y $a_i > a_j$. ¿Cuál es el número esperado de inversiones en tales permutaciones?

3.3. Resolución de problemas de conteo mediante relaciones de recurrencia

La solución de algunos problemas complejos de conteo no puede ser encontrada fácilmente mediante las técnicas básicas que hemos visto antes, pero tal solución sí puede ser expresada mediante una *relación de recurrencia*. Esta define la solución al problema en entradas de tamaño n en términos de la solución sobre entradas de menor tamaño. Aunque resolver relaciones de recurrencia es, en general, un problema difícil, en algunos casos particulares que estudiaremos en este capítulo podemos desarrollar métodos que nos permitan encontrar fórmulas explícitas que representan la solución de la ecuación.

3.3.1. Contando con relaciones de recurrencia

Podemos usar las relaciones de recurrencia para resolver una amplia variedad de problemas de conteo. En esta sección veremos algunos ejemplos.

Ejemplo 3.3.1. Consideremos el siguiente problema que representa el crecimiento de la población de conejos en una isla bajo condiciones extremadamente simplificadas. Un par de conejos se deja en el mes 1 en una isla desierta. Un par de conejos no se reproduce hasta que ambos tienen 2 meses de edad. Cada par con al menos dos meses de edad se reproduce una vez al mes, generando un nuevo par de conejos. ¿Cuántos pares de conejos hay al finalizar el mes n, para $n \geq 1$?

Podemos encontrar una solución implícita a este problema mediante una relación de recurrencia. Sea f_n el número de pares de conejos al finalizar el mes n. Luego, $f_1 = 1$ y $f_2 = 1$, ya que el par de conejos dejado en la isla inicialmente no se reproduce hasta tener 2 meses de edad. A partir del mes 3, este par de conejos comienza a reproducirse, generando un nuevo par de conejos cada mes. Correspondientemente, cada nuevo par de conejos comienza a reproducirse luego de que ambos cumplen 2 meses. Esto genera aún más pares. En particular, el número de pares de conejos al finalizar el mes n corresponde al número de pares de conejos al finalizar el mes $n-1$ más el número de pares de conejos que nacieron durante el mes n. Es fácil observar que tal número es igual al número de pares de conejos al finalizar el mes $(n-2)$, ya que solo los pares de conejos con 2 meses de edad pueden reproducirse. Concluimos que:

$$f_n = f_{n-1} + f_{n-2}, \qquad n \geq 2.$$

La relación de recurrencia definida por esta ecuación junto con $f_1 = f_2 = 1$ corresponde a la famosa secuencia de los *números de Fibonacci*. Tal secuencia proviene, precisamente, del estudio del problema antes descrito realizada por el matemático italiano Fibonacci. □

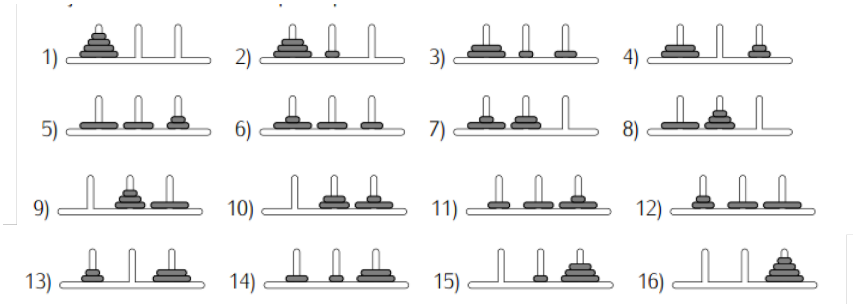


Figura 3.1: Solución al juego de las Torres de Hanoi con 4 discos.

Ejemplo 3.3.2. Un problema muy popular en el área de la computación es el de las *Torres de Hanoi*: Este consiste de tres clavijas montadas en un tablero junto con discos de diferente tamaño. Inicialmente, los discos están localizados en la primera clavija, en orden decreciente de tamaño, con el disco más grande abajo. Las reglas del problema permiten mover los discos uno a la vez desde una clavija a la otra, bajo la condición de que nunca un disco se ponga sobre otro de menor tamaño. El objetivo del problema es pasar todos los discos desde la primera a la tercera clavija. En la Figura 3.1 observamos una solución al juego de las Torres de Hanoi con 4 discos.

Es posible encontrar una relación de recurrencia que exprese el número h_n de movimientos necesarios para resolver el juego de las Torres de Hanoi con n discos. En efecto, si $n > 1$, entonces podemos comenzar por jugar Hanoi con los $n-1$ discos más pequeños y moverlos todos a la segunda clavija utilizando h_{n-1} movimientos (durante esta parte del juego el disco más grande se mantiene fijo en la primera clavija). Luego pasamos el disco más grande a la tercera clavija con un movimiento, y finalizamos moviendo nuevamente los $(n-1)$ discos más pequeños desde la segunda a la tercera clavija – quedando todos estos sobre el disco más grande – lo que requiere h_{n-1} movimientos más. Es decir:

$$h_n \;=\; 2h_{n-1} + 1, \qquad n > 1.$$

Por otro lado, el caso base corresponde a $h_1 = 1$, ya que un movimiento basta para mover un disco de una clavija a otra. □

Ejemplo 3.3.3. El número a_n de palabras binarias de largo n que no tienen dos 0s consecutivos puede definirse recursivamente como sigue:

$$a_n \;=\; a_{n-1} + a_{n-2}, \qquad n > 1.$$

En efecto, las palabras binarias que no tienen dos 0s consecutivos solo pueden ser de dos formas:

1. $w1$, donde w es una palabra binaria de largo $(n-1)$ que no tiene dos 0s consecutivos;
2. $w10$, donde w es una palabra binaria de largo $(n-2)$ que no tiene dos 0s consecutivos.

El número de palabras del primer y segundo tipo es precisamente a_{n-1} y a_{n-2}, respectivamente. El resultado ahora se sigue del hecho de que una palabra no puede ser de ambos tipos, ya que las del primer tipo terminan en 1 y las del segundo en 0. Los casos base corresponden a $a_0 = 1$ y $a_1 = 2$. □

Ejercicio. *Defina recursivamente el número de palabras binarias que sí tienen dos 0s consecutivos.*

Todas las relaciones de recurrencia vistas en esta sección son de la forma:

$$a_n \;=\; c_1 a_{n-1} + \cdots + c_k a_{n-k} + f(n),$$

donde los c_is son enteros y $f : \mathbb{N} \to \mathbb{N}$. Tales relaciones de recurrencia son *lineales* y con *coeficientes enteros*, ya que $c_1 a_{n-1} + \cdots + c_k a_{n-k}$ corresponde a una combinación lineal con coeficientes enteros de los términos previos de la relación. Cuando $f(n)$ no está presente se dice además que la relación es *homogénea*. Por ejemplo, la relación de recurrencia que define los números de Fibonacci es homogénea, pero la que define el número de jugadas de las Torres de Hanoi no lo es.

Las relaciones de recurrencia de este tipo presentan buena propiedades que en algunos casos importantes nos permiten encontrar una forma explícita para representar su solución.

3.3.2. Solución a relaciones de recurrencia lineales y con coeficientes enteros

El caso homogéneo

Comencemos por analizar el caso más simple en el que el término $f(n)$ no está presente; es decir, la relación de recurrencia es homogénea. Considere entonces una relación de recurrencia lineal, con coeficientes enteros, y homogénea de la forma:

$$a_n = c_1 a_{n-1} + \cdots + c_k a_{n-k}, \tag{3.2}$$

donde $c_k \neq 0$. Nos concentramos en buscar soluciones de la forma r^n, donde r es una constante. Para que $\{r^n \mid n \geq 0\}$ sea solución a la relación de recurrencia expresada en (3.2) debe suceder que:

$$r^n = c_1 r^{n-1} + \cdots + c_k r^{n-k},$$

o equivalentemente que:

$$r^n - c_1 r^{n-1} - \cdots - c_k r^{n-k} = 0.$$

Dividiendo ambos lados por r^{n-k}, obtenemos que esto es equivalente a:

$$r^k - c_1 r^{k-1} - \cdots - c_k = 0.$$

En suma, $\{r^n \mid n \geq 0\}$ es solución a la relación de recurrencia expresada en (3.2) si, y solo si, r es solución de la ecuación de grado k:

$$x^k - c_1 x^{k-1} - \cdots - c_k = 0.$$

Llamamos a esta última la *ecuación característica* de la relación de recurrencia.

Ecuaciones características de grado 2 Como en general es difícil encontrar soluciones a ecuaciones de grado k, nos concentraremos para comenzar en las ecuaciones de recurrencia de la forma:

$$a_n = c_1 a_{n-1} + c_2 a_{n-2},$$

la que tiene asociada la ecuación característica $x^2 - c_1 x - c_2 = 0$, que es de grado 2 (y para la cual sabemos encontrar sus soluciones). Note, en particular, que todas las relaciones de recurrencia homogéneas vistas en los ejemplos de la sección anterior son de esta forma.

En general, asumimos que las raíces r_1, r_2 de la ecuación característica $x^2 - c_1 x - c_2 = 0$ son reales. Comencemos con el caso en que estas raíces son distintas. El siguiente resultado expresa que las soluciones de la relación de recurrencia $a_n = c_1 a_{n-1} + c_2 a_{n-2}$ pueden ser expresadas como una combinación lineal de r_1^n y r_2^n:

Teorema 3.3.1. *Sean r_1, r_2 raíces reales de la ecuación característica $x^2 - c_1 x - c_2 = 0$ tal que $r_1 \neq r_2$. Entonces toda solución a la relación de recurrencia $a_n = c_1 a_{n-1} + c_2 a_{n-2}$ es de la forma:*

$$a_n = k_1 r_1^n + k_2 r_2^n,$$

donde k_1, k_2 son constantes.

Además, siempre es posible encontrar valores de k_1 y k_2 que hacen que la solución $k_1 r_1^n + k_2 r_2^n$ satisfaga las condiciones iniciales $a_0 = c$ y $a_1 = d$.

Demostración. Primero demostramos que $a_n = k_1 r_1^n + k_2 r_2^n$ es solución de la relación de recurrencia $a_n = c_1 a_{n-1} + c_2 a_{n-2}$. Para ello debemos demostrar que:

$$k_1 r_1^n + k_2 r_2^n \;=\; c_1(k_1 r_1^{n-1} + k_2 r_2^{n-1}) + c_2(k_1 r_1^{n-2} + k_2 r_2^{n-2}).$$

Agrupando términos podemos reescribir el lado derecho como:

$$k_1 r_1^{n-2}(c_1 r_1 + c_2) + k_2 r_2^{n-2}(c_1 r_2 + c_2).$$

Ocupando el hecho de que r_1 y r_2 son raíces de $x^2 - c_1 x - c_2 = 0$ obtenemos que $c_1 r_1 + c_2 = r_1^2$ y $c_1 r_2 + c_2 = r_2^2$. Luego:

$$k_1 r_1^{n-2}(c_1 r_1 + c_2) + k_2 r_2^{n-2}(c_1 r_2 + c_2) \;=\; k_1 r_1^2 + k_2 r_2^2.$$

Ahora demostramos que existen k_1, k_2 tales que la solución $k_1 r_1^n + k_2 r_2^n$ satisface las condiciones iniciales $a_0 = c$ y $a_1 = d$. Para ello es necesario que:

$$\begin{aligned} k_1 + k_2 &= c \\ k_1 r_1 + k_2 r_2 &= d \end{aligned}$$

Este sistema de ecuaciones tiene una solución dada por:

$$k_1 = \frac{d - c r_2}{r_1 - r_2} \qquad\qquad k_2 = \frac{c r_1 - d}{r_1 - r_2}.$$

Note que para que esta solución esté bien definida es necesaria nuestra hipótesis de que $r_1 \neq r_2$. □

Ejemplo 3.3.4. Utilizando este resultado podemos ahora encontrar una solución a la secuencia de los números de Fibonacci cuya definición recursiva corresponde a $f_n = f_{n-1} + f_{n-2}$. La ecuación característica de esta expresión es $x^2 - x - 1 \;= 0$, la que tiene dos raíces reales distintas: $\frac{1+\sqrt{5}}{2}$ y $\frac{1-\sqrt{5}}{2}$. Por tanto, la solución es de la forma:

$$k_1 \cdot \frac{1+\sqrt{5}}{2} \;+\; k_2 \cdot \frac{1-\sqrt{5}}{2}.$$

Los valores k_1 y k_2 que satisfacen las condiciones iniciales $f_0 = 1$ y $f_1 = 1$ son:

$$k_1 = \frac{1}{\sqrt{5}} \qquad\qquad k_2 = -\frac{1}{\sqrt{5}}.$$

□

¿Qué podemos hacer en el caso de que la ecuación característica $x^2 - c_1 x - c_2 = 0$ tenga una sola raíz real r con multiplicidad 2? Para ser capaces de determinar las constantes que satisfacen las condiciones iniciales necesitamos, en este caso, cambiar un poco la forma de la solución:

Teorema 3.3.2. *Suponga que la ecuación característica $x^2 - c_1 x - c_2 = 0$ tiene una sola raíz real r con multiplicidad 2. Entonces existe solución a la relación de recurrencia $a_n = c_1 a_{n-1} + c_2 a_{n-2}$ es de la forma:*

$$a_n \;=\; k_1 r^n + k_2 n r^n,$$

donde k_1, k_2 son constantes.

Además, siempre es posible encontrar valores de k_1 y k_2 que hacen que la solución $k_1 r^n + k_2 n r^n$ satisfaga las condiciones iniciales $a_0 = c$ y $a_1 = d$.

Demostración. Primero demostramos que $a_n = k_1 r^n + k_2 n r^n$ es solución de la relación de recurrencia $a_n = c_1 a_{n-1} + c_2 a_{n-2}$. Para ello debemos demostrar que:

$$k_1 r^n + k_2 n r^n \;=\; c_1(k_1 r^{n-1} + k_2(n-1) r^{n-1}) + c_2(k_1 r^{n-2} + k_2(n-2) r^{n-2}).$$

Agrupando términos podemos reescribir el lado derecho como:

$$k_1 r^{n-2}(c_1 r + c_2) + k_2 r^{n-2}(n(c_1 r + c_2) - (c_1 r + 2c_2)).$$

Ocupando el hecho de que r es raíz de de $x^2 - c_1 x - c_2 = 0$ obtenemos que $c_1 r + c_2 = r^2$. Luego:

$$k_1 r^{n-2}(c_1 r + c_2) + k_2 r^{n-2}(n(c_1 r + c_2) - (c_1 r + 2c_2)) \;=\; k_1 r^2 + k_2 r^{n-2}(nr^2 - (c_1 r + 2c_2)).$$

Por tanto, para terminar la demostración basta establecer que $c_1 r + 2c_2 = 0$. Pero esto se sigue del hecho de que si $x^2 - c_1 x - c_2 = 0$ tiene como raíz solo a r, entonces $c_1/2 = r$ y $c_1^2 + 2c_2 = 0$. Es decir, por la primera condición $c_1 = 2r$, y por la segunda $c_2 = -2r^2$. Concluimos que $c_1 r - 2c_2 = 0$.

Ahora demostramos que existen k_1, k_2 tales que $k_1 r^n + k_2 n r^n$ satisface las condiciones iniciales $a_0 = c$ y $a_1 = d$. Para ello es necesario que:

$$\begin{aligned} k_1 &= c \\ k_1 r + k_2 r &= d \end{aligned}$$

Este sistema de ecuaciones tiene una solución dada por: $k_1 = c$ y $k_2 = \frac{d - cr}{r}$. □

Ejercicio. *Utilizando el resultado anterior encuentre la solución de la relación de recurrencia* $a_n = 6a_{n-1} - 9a_{n-2}$ *con condiciones iniciales* $a_0 = 1$ *y* $a_1 = 6$.

Ecuaciones características de grado $k \geq 2$ Volvamos ahora el caso general en el que la relación de recurrencia es de la forma:

$$a_n \;=\; c_1 a_{n-1} + \cdots + c_k a_{n-k},$$

asumiendo $c_k \neq 0$, y por tanto que la ecuación característica $x^k - c_1 x^{k-1} - \cdots - c_k = 0$ es de un grado arbitrario $k \geq 1$. El siguiente resultado (cuya demostración omitimos) determina la forma general de las soluciones de esta ecuación. En particular, este resultado extiende los dos resultados antes vistos:

Teorema 3.3.3. *Asuma que la ecuación característica* $x^k - c_1 x^{k-1} - \cdots - c_t = 0$ *tiene tan solo raíces reales* $r_1, \ldots, r_t$*, las que aparecen con multiplicidad* $m_1, \ldots, m_t$ *(*$\sum_{1 \leq i \leq t} m_i = k$*), respectivamente. Entonces existe solución a la relación de recurrencia* $a_n = c_1 a_{n-1} + \cdots + c_k a_{n-k}$ *de la forma:*

$$\begin{aligned} a_n \;=\; & (d_{1,0} + d_{1,1} n + \cdots + d_{1,m_1-1} n^{m_1-1}) r_1^n + \\ & (d_{2,0} + d_{2,1} n + \cdots + d_{2,m_2-1} n^{m_2-1}) r_2^n \\ & + \cdots + (d_{t,0} + k_{t,1} n + \cdots + d_{t,m_t-1} n^{m_t-1}) r_t^n \end{aligned}$$

donde los $d_{i,j}$ *son constantes para cada* $1 \leq i \leq t$ *y* $0 \leq j \leq m_i - 1$.

Además, siempre es posible encontrar valores de las constantes $d_{i,j}$*, para* $1 \leq i \leq t$ *y* $0 \leq j \leq m_i - 1$*, que hacen que la solución arriba expresada satisfaga las condiciones iniciales.*

El caso no homogéneo

El caso no homogéneo, es decir, cuando la relación de recurrencia es de la forma:

$$a_n \;=\; c_1 a_{n-1} + \cdots + c_k a_{n-k} + f(n),$$

es bastante más complicado ya que $f : \mathbb{N} \to \mathbb{N}$ puede ser una función arbitraria. Sin embargo, como veremos en esta sección para algunas funciones de interés podemos encontrar una solución explícita al problema.

Primero, es posible relacionar las soluciones de una relación de recurrencia no homogénea de la forma $a_n = c_1 a_{n-1} + \cdots + c_k a_{n-k} + f(n)$ con las de la *relación de recurrencia homogénea asociada*, la que se obtiene simplemente obviando el término $f(n)$; es decir, esta relación es: $a_n = c_1 a_{n-1} + \cdots + c_k a_{n-k}$. En efecto:

Proposición 3.3.1. *Sean dos soluciones cualesquiera* $\{b_n \mid n \geq 0\}$ *y* $\{e_n \mid n \geq 0\}$ *de la relación de recurrencia no homogénea* $a_n = c_1 a_{n-1} + \cdots + c_k a_{n-k} + f(n)$. *Entonces* $\{b_n - e_n \mid n \geq 0\}$ *es solución de la ecuación homogénea asociada* $a_n = c_1 a_{n-1} + \cdots + c_k a_{n-k}$.

Dado que ya sabemos encontrar soluciones a relaciones homogéneas, el resultado anterior nos dice que para encontrar las soluciones de una relación de recurrencia no homogénea basta encontrar una en particular y sumarla con las de la homogénea asociada. El problema es encontrar tal solución particular. El siguiente resultado (presentado sin demostración) nos entrega tal herramienta para una amplia gama de casos de interés:

Teorema 3.3.4. *Considere la relación de recurrencia no homogénea de la forma* $a_n = c_1 a_{n-1} + \cdots + c_k a_{n-k} + f(n)$, *donde:*

$$f(n) = (b_t n^t + \cdots + b_1 n + b_0) s^n,$$

y los b_i*'s son coeficientes enteros. Considere la ecuación característica asociada* $x^k - c_1 x^{k-1} - \cdots - c_k = 0$. *Entonces:*

1. *Si* s *no es raíz de* $x^k - c_1 x^{k-1} - \cdots - c_k = 0$, *existe solución particular de la relación no homogénea que es de la forma:*
$$(p_t n^t + \cdots + p_1 n + p_0) s^n.$$
2. *Si* s *es raíz de* $x^k - c_1 x^{k-1} - \cdots - c_k = 0$ *con multiplicidad* m, *existe solución particular de la relación no homogénea que es de la forma:*
$$n^m (p_t n^t + \cdots + p_1 n + p_0) s^n.$$

Con este resultado podemos ahora analizar varios casos interesantes. Uno de ellos lo presentamos en el siguiente ejemplo.

Ejemplo 3.3.5. La suma s_n de los primeros n números naturales se define recursivamente como:

$$s_n = s_{n-1} + n, \qquad n \geq 1,$$

con caso inicial $s_0 = 0$. Note que $f(n)$ es en este caso de la forma para el cual el Teorema 3.3.4 nos provee solución; en particular, es de la forma $f(n) = (b_t n^t + \cdots + b_1 n + b_0) s^n$ para $b_t = \cdots = b_1 = 0$, $b_0 = 1$, y $s = 1$.

La ecuación homogénea asociada es $x - 1 = 0$, la que tiene como raíz a 1. Por ende:

- Las soluciones de la relación de recurrencia homogénenea asociada $s_n = s_{n-1}$ son de la forma $k \cdot 1^n = k$.
- Dado que $s = 1$, y 1 es raíz de $x - 1 = 0$ con multiplicidad 1, el Teorema 3.3.4 nos dice que existe una solución particular de $s_n = s_{n-1} + n$ de la forma $n(p_1 n + p_0)$.

Por tanto, las soluciones generales de la relación $s_n = s_{n-1} + n$ son de la forma:

$$k + p_0 n + p_1 n^2.$$

Para determinar las constantes k, p_0, p_1 simplemente utilizamos las condiciones iniciales $s_0 = 0$, $s_1 = 1$, y $s_2 = 3$. A partir de esto obtenemos el siguiente sistema de ecuaciones:

$$\begin{aligned} k &= 0 \\ p_0 + p_1 &= 1 \\ 2p_0 + 4p_1 &= 3 \end{aligned}$$

Resolviendo obtenemos $k = 0$ y $p_1 = p_2 = \frac{1}{2}$. Por tanto, la solución final es $s_n = \frac{n(n+1)}{2}$, la que corresponde a la bien conocida fórmula para determinar el valor de la suma de los primeros n naturales. □

Ejercicio. *Encuentre una solución a la relación de recurrencia que define el número de movimientos necesarios para resolver el problema de las Torres de Hanoi.*

3.3.3. Análisis del costo de algoritmos recursivos mediante el Teorema Maestro

Como vimos antes, los algoritmos recursivos dividen una instancia de un problema en instancias más pequeñas, y luego recursivamente resuelven estas. Por ejemplo, Mergesort divide una lista A de largo n en dos listas B y C de largo $n/2$, luego recursivamente ordena a B y C, y finalmente obtiene un ordenamiento para A basado en los ordenamientos de B y C. Esto quiere decir que el costo $A(n)$ de Mergesort en una lista de tamaño n, asumiendo n par, es:

$$A(n) \;=\; 2A(n/2) + O(n).$$

El término $O(n)$ representa el hecho que se necesitan $O(n)$ comparaciones para mezclar las versiones ordenadas de B y C.

De la misma forma se puede demostrar que el costo del algoritmo de búsqueda binaria en una lista ordenada de tamaño n satisface la siguiente definición recursiva:

$$B(n) \;=\; B(n/2) + O(1).$$

El término $O(1)$ representa el hecho que se necesitan dos comparaciones para determinar en cuál mitad de la lista seguir buscando, y si hay o no elementos en esa mitad.

Considere, por otro lado, un algoritmo que determina el máximo y el mínimo de una lista A de tamaño n de la siguiente forma: (1) Divida la lista A en listas B y C de tamaño $n/2$. (2) Compute el máximo y el mínimo de B y C. (3) Determine el máximo y el mínimo de A comparando los máximos y los mínimos de B y C, respectivamente. El costo $C(n)$ de este algoritmo puede definirse como:

$$C(n) \;=\; 2C(n/2) + O(1).$$

El Teorema Maestro

El tipo de ecuaciones recursivas que definen el costo de estos algoritmos no corresponden a las ecuaciones lineales con coeficientes enteros que vimos antes. Sin embargo, es de todas formas posible encontrar una forma general de solución a través del Teorema Maestro que presentamos en esta sección. En particular, este teorema nos dice cómo determinar una cota para cualquier función $f : \mathbb{N} \to \mathbb{R}$ creciente que satisface lo siguiente para todo n:

$$f(n) \;\leq\; af(\lceil n/b \rceil) + cn^d,$$

donde a y b son enteros positivos, $b > 1$, c es un real positivo, y d es un entero no negativo.

Teorema 3.3.5 (Teorema Maestro)**.** *Sea $f : \mathbb{N} \to \mathbb{R}$ una función creciente que satisface:*

$$f(n) \;\leq\; af(\lceil n/b \rceil) + cn^d,$$

donde a y b son enteros positivos, $b > 1$, c es un real positivo, y d es un entero no negativo. Entonces:

$$f(n) \text{ es } \begin{cases} O(n^d), & \text{si } a < b^d, \\ O(n^d \log n), & \text{si } a = b^d, \\ O(n^{\log_b a}), & \text{si } a > b^d. \end{cases}$$

Demostración. Asuma inicialmente que $n = b^k$, para $k \geq 1$. Entonces

$$\begin{aligned} f(n) &\leq af(n/b) + cn^d \\ &\leq a\big(af(n/b^2) + c(n/b)^d\big) + cn^d \\ &\leq a\Big(a\big(af(n/b^3) + c(n/b^2)\big) + c(n/b)^d\Big) + cn^d \\ &\leq \cdots \\ &\leq a^k f(n/b^k) + \sum_{i=0}^{k-1} a^i c(n/b^i)^d \\ &= a^k f(1) + \sum_{i=0}^{k-1} a^i c(n/b^i)^d. \end{aligned}$$

Por tanto, para todo n de la forma b^k tenemos que $f(n) \leq a^k f(1) + cn^d \cdot \sum_{i=0}^{k-1}(a/b^d)^i$.

Asuma primero que $a < b^d$. Entonces $\sum_{i=0}^{k-1}(a/b^d)^i$ es igual a una constante c'. Por tanto, para todo n de la forma b^k tenemos que $f(n) \leq a^k f(1) + cc'n^d$. Demostraremos entonces que $f(n)$ es $O(n^d)$. Sea n un entero cualquiera. Por tanto, existe $k \geq 1$ tal que $b^k \leq n < b^{k+1}$. Luego, $f(n) < f(b^{k+1}) = a^{k+1}f(1) + cc'b^{(k+1)d}$. Por tanto, $f(n)$ es $O(a^k + n^d)$. Pero $a^k \leq b^{kd} \leq n^d$, por hipótesis, es decir, $f(n)$ es $O(n^d)$.

Asuma ahora que $a = b^d$. Sabemos que para todo n de la forma b^k tenemos que $f(n) \leq a^k f(1) + cn^d \cdot \sum_{i=0}^{k-1}(a/b^d)^i$. Por tanto, en este caso $f(n) \leq a^k f(1) + ckn^d$. Sea n un entero cualquiera. Por tanto, existe $k \geq 1$ tal que $b^k \leq n < b^{k+1}$. Luego, $f(n) < f(b^{k+1}) = a^{k+1}f(1) + c(k+1)b^{(k+1)d}$. Esto quiere decir que $f(n)$ es $O(a^k + kn^d)$, y por un análisis similar al anterior que $f(n)$ es $O(n^d + n^d \log_b n)$. Dado que $n^{\log_b a} = n^d$ concluimos que $f(n)$ es $O(n^d \log_b n)$.

El caso $a > b^d$ es similar y se deja como ejercicio. □

Como corolario obtenemos entonces que Mergesort puede resolverse en tiempo $O(n \log n)$, búsqueda binaria en tiempo $O(\log n)$, y que el algoritmo para el máximo y el mínimo toma tiempo $O(n)$.

3.4. El Método de Sustitución

A veces es posible resolver una ecuación de recurrencia "adivinando"la forma de solución y luego verificándola por inducción. Esto sirve también para determinar las constantes exactas involucradas en el término $O(\dots)$ que define el costo del algoritmo.

Ejemplo 3.4.1. Suponga que:

$$T(n) = 2T(\lfloor n/2 \rfloor) + n.$$

Entonces el Teorema Maestro nos dice que $T(n)$ es $O(n \log n)$. Probaremos entonces con un $T(n)$ tal que $T(n) \leq cn \log n$, y verificaremos si existe una constante c para la cual se satisfaga la definición recursiva. Por HI podemos asumir que:

$$T(\lfloor n/2 \rfloor) \leq c\lfloor n/2 \rfloor \log \lfloor n/2 \rfloor.$$

Por tanto, para que $T(n) \leq cn \log n$ basta que se cumpla que

$$2c\lfloor n/2 \rfloor \log \lfloor n/2 \rfloor + n \leq cn \log n.$$

Claremente:

$$2c\lfloor n/2 \rfloor \log \lfloor n/2 \rfloor + n \leq cn \log n/2 + n = cn \log n - cn + n.$$

Para que este último término sea menor que $cn \log n$ basta que $c \geq 1$. Por otro lado, se debe cumplir también que $T(2) \leq 2c$ para cumplir las condiciones iniciales. □

Ejemplo 3.4.2. Considere ahora:

$$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 1.$$

El Teorema Mestro nos dice que $T(n)$ es $O(n)$, pero ¿podemos demostrar que $T(n) \leq cn$? Por HI sabemos que:

$$T(n) \leq c\lfloor n/2 \rfloor + c\lceil n/2 \rceil + 1 = cn + 1,$$

pero esto no demuestra que $T(n) \leq cn$. Debemos, por tanto, asumir algo más fuerte: $T(n) = cn - b$, para $b > 0$. Por HI sabemos entonces que:

$$T(n) \leq c\lfloor n/2 \rfloor + c\lceil n/2 \rceil + 1 - 2b \leq cn + 1.$$

Las constantes c y b debe además satisfacer la condición inicial $T(1) \leq c - b$. □

Ejercicios Finales

1. Encuentre una relación de recurrencia que defina el número de strings sobre el alfabeto $\{a, b, c\}$ que tengan dos símbolos iguales consecutivos. ¿Es posible expresar esta relación de recurrencia como $a_n = ca_{n-1} + f(n)$, donde c es un entero positivo y f es una función?

2. Un modelo permite estimar la caza de huemules en un año mediante el promedio de caza de los dos años anteriores.

   - Encuentre una relación de recurrencia que defina a $\{H_n\}$, el número de huemules cazados en un año, para todo $n > 3$.
   - Resuelva esta relación de recurrencia con condiciones iniciales $H_1 = 1$ y $H_2 = 3$.

3. Asuma que existen n distintas sillas. Encuentre una relación de recurrencia que determine el número de maneras en que un subconjunto de estas sillas puede ser elegido, de modo que dos sillas contiguas nunca pertenecen al subconjunto elegido, si:

   a) Las sillas están arregladas en una línea.

   b) Las sillas están arregladas en un círculo.

   Determine las condiciones iniciales en ambos casos. Fundamente su respuesta.

4. Sea $a_n = \sum_{k=1}^{n} k^2$. Defina una relación de recurrencia que represente a esta secuencia. Encuentre la forma general de las soluciones de esta relación de recurrencia. Resuelva las constantes con respecto a las condiciones iniciales $a_1 = 1$, $a_2 = 5$ y $a_3 = 14$.

5. Defina recursivamente una función p_n, $n \geq 1$, que cuente el número de particiones del conjunto $\{1, 2, \ldots, n\}$.

6. Sea $S(m, n)$ el número de funciones sobreyectivas que van desde un conjunto con m elementos hasta un conjunto con n elementos, $n \leq m$. Defina recursivamente a $S(m, n)$ en función de m, n, $\binom{n}{k}$, para $k < n$, y $S(m, k)$, para $k < n$. Justifique su respuesta.

7. Encuentre las soluciones al sistema de ecuaciones recursivas

$$\begin{aligned} a_n &= a_{n-1} + b_{n-1} \\ b_n &= a_{n-1} - b_{n-1} \end{aligned}$$

   con $a_0 = 1$ y $b_0 = 2$.

8. Demuestre que $f(n)$ es $O(\log n \cdot \log(\log n))$, si $f(n) = 2f(\sqrt{n}) + \log n$, para todo cuadrado perfecto $n > 1$. (*Hint:* Utilice la transformación $m = \log n$).

9. Considere el siguiente algoritmo para encontrar el k-ésimo elemento de un arreglo desordenado $A[1, n]$:

   a) Cortar el arreglo en grupitos de 5 elementos y calcular la mediana de cada grupito.

   b) Calcular la mediana m de las $n/5$ medianas, usando recursión.

   c) Usar esa mediana para separar el arreglo entre los menores y mayores que m.

   d) Sea t la cantidad de elementos menores que m. Si $k = t + 1$, entonces responder m y terminar. Si $k \leq t$ continuar recursivamente en el conjunto de los menores. Sino, continuar recursivamente en el grupo de los mayores, ahora con $k - t - 1$ en vez de k.

   Se pide:

   a) Demostrar que m debe ser mayor que 3/10 de los elementos de A, y menor que otros 3/10.

   b) Establecer una recurrencia para el costo $T(n)$ del algoritmo, y probar que $T(n) = O(n)$.

10. Encuentre una relación de recurrencia que defina la cantidad a_n de regiones que son formadas por n líneas rectas en el plano, asumiendo que no hay líneas paralelas ni tres líneas distintas que se corten en el mismo punto. Resuelva la relación de recurrencia y encuentre una solución explícita para a_n.

11. Sea R_n el número de regiones que se generan en la superficie de una esfera al ser dividida por n grandes círculos (es decir por n planos que pasan por el centro de la esfera), asumiendo que no hay tres de estos grandes círculos que pasen por el mismo punto.

    - Explique por qué $R_{n+1} = R_n + 2n$, para $n \geq 1$.
    - Resuelva iterativamente la expresión anterior.

12. Sea $s(n)$ el número de secuencias $(x_1, \ldots, x_k)$ de enteros, tal que $1 \leq x_i \leq n$ para cada $1 \leq i \leq k$, y $x_{i+1} \geq 2x_i$ para cada $1 \leq i < k$. (Note que el largo de la secuencia no está especificado, y, en particular, la secuencia vacía se haya incluída).

    a) Asumiendo que $s(0) = 1$, demuestre que para todo $n \geq 1$ se cumple que:

    $$s(n) \;=\; s(n-1) + s(\lfloor n/2 \rfloor).$$

    Ayuda: Considere por separado las secuencias que contienen a n y aquellas que no.

    b) Utilizando lo anterior, demuestre que $s(n) \leq n \cdot s(\lfloor n/2 \rfloor)$, para todo $n \geq 2$.

    c) Concluya que $s(n)$ es $O(n^{\log n})$.

13. Considere una función $f : \mathbb{N} \to \mathbb{N}$ que satisface la siguiente definición recursiva:

    - $f(0) = f(1) = 1$.
    - $f(n) \leq f(\lfloor \frac{7n}{10} \rfloor) + f(\lfloor \frac{n}{5} \rfloor) + n$, para $n \geq 2$.

    Demuestre que existe $c \geq 1$ tal que $f(n) \;\leq cn$, para todo $n \geq 0$.

Capítulo 4

Teoría de Grafos

Los grafos son estructuras discretas que consisten de *vértices* (también conocidos como *nodos*), y de *arcos* que conectan a tales vértices. Visto de otra forma, un grafo no es más que una forma visual de representar una relación binaria. Los grafos se utilizan para modelar y resolver problemas en casi cualquier disciplina científica que uno pueda pensar. También se pueden utilizar para representar y analizar todo tipo de interacciones entre individuos, incluyendo por ejemplo las redes sociales o las redes de transporte, lo que los hace particularmente importante en la actualidad.

4.1. Conceptos básicos

4.1.1. Modelos de grafos

En su forma más general, un grafo se define como sigue:

Definición 4.1.1. *Un* grafo $G = (V, E)$ *consiste de un conjunto no vacío* V *de* vértices *o* nodos*, y un conjunto* E *de* arcos*, de tal forma que cada arco* $e \in E$ *se halla asociado con un par* (u, v) *de vértices en* V*. Se dice entonces que el arco* e conecta a u con v*.*

Ejemplo 4.1.1. Un ejemplo interesante de grafo es la Web: los vértices son las páginas, y dos páginas están conectadas si existe un link de la primera a la segunda. Este grafo permite *loops*, i.e., arcos que conectan un nodo consigo mismo, *arcos paralelos*, i.e., arcos distintos que conectan el mismo par de nodos, y para su representación son importantes las *direcciones en los arcos*. □

Dependiendo del tipo de aplicación que queremos modelar con nuestros grafos, podemos dejar de lado uno o más elementos de estos grafos:

- *Grafos simples:* No permiten loops, ni arcos paralelos, ni importan las *direcciones en los arcos*, i.e., si el par (u, v) está conectado también está (v, u).

  A pesar de su simplicidad el estudio de los grafos simples presenta grandes desafíos teóricos. En efecto, el objeto básico de la teoría de grafos es precisamente esta clase de grafos. Por otro lado, por medio de los grafos simples también podemos modelar situaciones prácticas interesantes. Una de ellas es el *grafo de colaboraciones de Erdös*: los nodos de este grafo son los científicos, de tal forma que dos nodos están conectados si y solo si han publicado un artículo en conjunto. La idea es conocer la distancia que un científico tiene en este grafo al nodo que representa al matemático húngaro Paul Erdös.

  Note que un grafo simple siempre se puede representar como $G = (V, E)$, donde E es simplemente un subconjunto de $V \times V$ tal que si $(u, v) \in E$ entonces $(v, u) \in E$. Este subconjunto contiene a los pares (u, v) de vértices que están conectados por un arco en G. En tal caso decimos que u y v son *adyacentes*.

- *Multigrafos:* Extienden los grafos simples permitiendo más de una conexión entre nodos; es decir, en este caso E es un multiconjunto de pares en $V \times V$. Un multigrafo podría ayudarnos, por ejemplo, a representar el número de colaboraciones existentes entre dos científicos en el grafo de colaboraciones visto antes.

- *Seudografo:* Extienden los multigrafos con loops.

- *Grafo dirigido:* Es un grafo dirigido que no cumple la condición de que E es simétrico; es decir, los arcos ahora sí pueden tener dirección. Si no permitimos multiarcos ni loops hablamos de *grafo dirigido simple*. Si permitimos multiarcos, entonces decimos que es un *multigrafo dirigido*. Si además permitimos loops, entonces volvemos simplemente a la definición de grafo que dimos en la Definición 4.1.1.

De ahora en adelante, suponemos que nuestros grafos son finitos. Además, vale la pena recalcar que en la mayoría de los casos nos concentramos en el estudio de los grafos simples.

4.1.2. Un resultado básico

Sea $G = (V, E)$ un grafo simple. El *grado* de un nodo $v \in V$ se define como el números de nodos $u \in V$ que son adyacentes a v. Denotados esto como $\mathsf{grado}(v)$. El siguiente resultado establece una relación básica entre la suma de los grados de los nodos en un grafo simple G y el número de arcos en G:

Teorema 4.1.1 (Teorema de los saludos)**.** *Sea $G = (V, E)$ un grafo simple. Se cumple que:*

$$\sum_{v \in V} \mathsf{grado}(v) \;=\; 2 \cdot |E|.$$

Demostración. Al sumar los grados de los nodos estamos contando cada arco dos veces, ya que cada arco en un grafo simple toca exactamente a dos vértices. □

No es difícil extender este resultado a multigrafos y seudografos; simplemente consideramos que cada arco que conecta a un vértice con otro vértice distinto aumenta en 1 su grado, mientras que los loops aumentan el grado en 2.

Por medio del Teorema de los Saludos podemos demostrar algunas propiedades interesantes de los grafos simples:

Corolario 4.1.1. *Todo grafo simple $G = (V, E)$ tiene un número par de nodos de grado impar.*

Demostración. Sean U y W los vértices en V de grado par y grado impar, respectivamente. Entonces:

$$\sum_{u \in U} \mathsf{grado}(u) + \sum_{w \in W} \mathsf{grado}(w) \;=\; 2 \cdot |E|. \tag{4.1}$$

Note que $\sum_{u \in U} \mathsf{grado}(u)$ es par, ya que cada $u \in U$ tiene grado par por definición. Es decir, tanto el lado derecho como el primer término del lado izquierdo de la Ecuación (4.1) son valores pares. Esto solo puede ocurrir si $\sum_{w \in W} \mathsf{grado}(w)$ es par; es decir, la suma de los grados de los nodos de grado impar en G es par. Esto quiere decir que W tiene un número par de elementos. □

4.1.3. Clases particulares de grafos simples

A continuación presentamos algunas importantes clases de grafos simples:

- *Grafos completos:* El grafo completo K_n, para $n \geq 1$, contiene n vértices y todos los posibles arcos entre esos vértices. En particular, entonces, K_n tiene $\binom{n}{2}$ arcos, uno por cada par de nodos distintos.

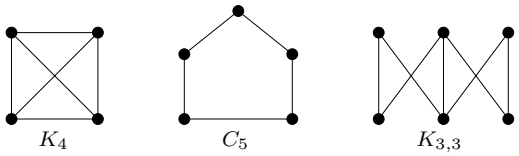


Figura 4.1: Los grafos K_4, C_5, y $K_{3,3}$.

- *Ciclos:* El ciclo C_n con n vértices, para $n \geq 3$, consiste de vértices $v_1, \ldots, v_n$ y arcos:

  $$(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n), (v_n, v_1).$$

  En particular, C_n tiene n arcos.

- *Grafos bipartitos:* Los grafos bipartitos son aquellos cuyo conjunto V de vértices puede ser particionado en dos conjuntos dijuntos V_1 y V_2, de tal forma que todo arco en E conecta un nodo de V_1 con un nodo de V_2.

  El grafo bipartito se dice que es *completo* si $E = V_1 \times V_2$; es decir, cada vértice en V_1 es adyacente a todo vértice en V_2. En particular, denotamos como $K_{m,n}$, para $m, n \geq 1$, el grafo bipartito completo en el que $|V_1| = m$ y $|V_2| = n$. Este grafo tiene mn arcos.

Algunos casos particulares de estos grafos se muestran en la Figura 4.1.

Ejercicio. *Demuestre que K_n no es bipartito para ningún $n \geq 3$.*

Ejercicio. *Demuestre que C_n es bipartito sii n es par.*

Algunos resultados básicos La siguiente proposición establece una caracterización útil de cuándo un grafo es bipartito:

Proposición 4.1.1. *Un grafo $G = (V, E)$ es bipartito sii es posible asignarle uno de dos colores a cada vértice de G de tal forma que si dos nodos son adyacentes entonces reciben colores distintos.*

Demostración. Asuma primero que $G = (V, E)$ es bipartito y que la partición de V que atestigua esto está dada por V_1 y V_2. Pinte los vértices en V_1 de azul y los vértices en V_2 de rojo. Entonces todo arco en E conecta un nodo azul con otro rojo. Por otro lado, asuma que es posible asignarle uno de dos colores a cada vértice de G de tal forma que si dos nodos son adyacentes entonces reciben colores distintos. Defina V_1 como el conjunto de nodos coloreados de azul y V_2 como el conjunto de nodos coloreados de rojo. Por definición, cada arco en E conecta un nodo en V_1 con uno en V_2. Es decir, V_1 y V_2 representan una partición de V que atestigua que G es bipartito. □

Pregunta. *¿Cómo podría utilizar el resultado anterior para diseñar un procedimiento eficiente que determine si un grafo es bipartito?*

Ejemplo 4.1.2. Sea $G = (V, E)$ un grafo bipartito con n vértices y m arcos. Estableceremos aquí que el máximo número de arcos que puede haber en E es $\frac{n^2}{4}$ (recuerde que en el caso general, es decir, cuando G no es bipartito, este número puede llegar a ser $\binom{n}{2} = \frac{n(n-1)}{2}$). En efecto, asuma que el hecho que G es bipartito es atestiguado por la partición de V dada por V_1 y V_2. Por tanto, $m \leq n_1 n_2$ asumiendo que $|V_1| = n_1$ y $|V_2| = n_2$. Demostraremos a continuación que $n_1 n_2 \leq \frac{n^2}{4}$, lo que demuestra el resultado. Sabemos por definición que $n = n_1 + n_2$. Luego:

$$\frac{n^2}{4} \geq \frac{(n_1 + n_2)^2}{4} \geq n_1 n_2,$$

donde la última desigualdad se tiene por el hecho de que $(n_1 - n_2)^2 \geq 0$. □

Sea G un grafo simple. Un *subgrafo* de G es cualquier grafo $G' = (V', E')$ tal que $V' \subseteq V$ y $E' \subseteq E$. A continuación presentamos un ejemplo de cómo es posible obtener un subgrafo que satisface ciertas propiedades:

Ejemplo 4.1.3. Sea $G = (V, E)$ un grafo simple con al menos un arco. Demostraremos que existe un subgrafo no vacío $G' = (V', E')$ de G tal que:

$$\delta(G') > \epsilon(G') \geq \epsilon(G),$$

asumiendo que:

- $\delta(G')$ es el grado mínimo de un vértice en G';
- $\epsilon(G) = \frac{m}{n}$ y $\epsilon(G') = \frac{m'}{n'}$, donde n, n' corresponden al número de vértices en V y V', respectivamente, y m, m' representan el número de arcos en E y E', respectivamente.

Es decir, queremos demostrar que el grado mínimo de un vértice en G' es estrictamente mayor al radio entre el número de arcos y vértices en G', que es a la vez al menos tan grande como tal radio en G.

Comencemos por pensar qué tiene que pasar para que podamos eliminar un vértice v de G sin disminuir el radio entre el número de arcos y el número de vértices en el grafo. Note que al eliminar v desde G disminuimos el número de vértices en 1 pero el número de arcos en $\mathsf{grado}(v)$. Por tanto, lo que buscamos en este caso es un vértice v que satisfaga lo siguiente:

$$\frac{m - \mathsf{grado}(v)}{n - 1} \geq \frac{m}{n}.$$

Al despejar $\mathsf{grado}(v)$ en esta desigualdad obtenemos $\mathsf{grado}(v) \leq \frac{m}{n} = \epsilon(G)$. Es decir, podemos eliminar un nodo v de G sin diminuir el radio $\epsilon(G)$ entre el número de arcos y nodos en G si es que el grado de v es a lo más tal radio.

Lo que podemos hacer entonces es ir eliminando iterativamente tales nodos de forma de obtener grafos $G, G_1, \ldots, G_{m-1}, G_m$ hasta que ya no se pueda aplicar más tal eliminación. Por definición tenemos entonces que:

$$\epsilon(G_m) \geq \epsilon(G_{m-1}) \geq \ldots \geq \epsilon(G_1) \geq \epsilon(G).$$

Además, G_m no tiene ningún vértice v tal que $\mathsf{grado}(v) \geq \epsilon(G_m)$, ya que de otra forma podríamos haber aplicado una vez más el procedimiento de eliminación de nodos. Definamos entonces G' como G_m. Se cumple lo que buscamos; es decir:

$$\delta(G') > \epsilon(G') \geq \epsilon(G).$$

Solo falta demostrar entonces que G' no es vacío. En efecto, este procedimiento siempre deja al menos un arco en el grafo. Suponga, por contradicción, que esto no es así; es decir, en algún momento el procedimiento borra todos los arcos del grafo (recuerde que inicialmente G tiene al menos un arco). Esto quiere decir que hemos borrado un nodo v tales que todos los arcos que quedan en el grafo son adyacentes a v. Pero entonces claramente el grado de v es estrictamente mayor que el radio entre el número de arcos y el número de nodos al momento de eliminar v. Esto es una contradicción. □

Ejercicio. *La* secuencia de grados *de un grafo simple G es la secuencia de los grados de sus nodos en orden no creciente.*

Asuma que $d_1, d_2, \ldots, d_n$ es la secuencia de los grados de un grafo simple. Demuestre que existe un grafo simple G con vértices $v_1, v_2, \ldots, v_n$ que satisface lo siguiente:

- *Para cada vértice v_i, con $1 \leq i \leq n$, el grado de v_i en G es d_i; y*
- *v_1 es adyacente a $v_2, v_3, \ldots, v_{d_1+1}$.*

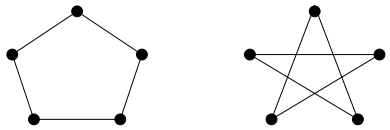


Figura 4.2: Los grafos isomorfos del Ejemplo 4.1.4.

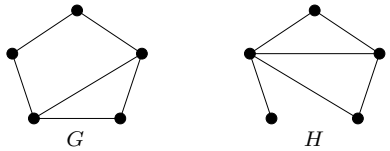


Figura 4.3: Los grafos del Ejemplo 4.1.5.

4.1.4. Isomorfismo de grafos

¿Cómo se puede representar computacionalmente un grafo simple $G = (V, E)$? Una forma común de hacerlo es mediante la *matriz de adyacencia*. Esta lista los vértices en V en las filas y columnas de la matriz, y agrega un 1 precisamente en aquellas celdas que representan a los pares (u, v) tal que $(u, v) \in E$. Por tanto, un grafo simple $G = (V, E)$ con n vértices puede ser representado mediante matrices de adyacencia de $n!$ formas diferentes, una por cada posible ordenamiento de los vértices en V.

Asuma que $G = (V, E)$ y $G' = (V', E')$ son grafos simples con la misma cantidad de vértices $n \geq 1$. ¿Qué significa entonces que $G = (V, E)$ y $G' = (V', E')$ tengan la misma "forma", o equivalentemente, que puedan ser dibujados de la misma manera? Para que esto ocurra debemos ser capaces de demostrar que G puede ser representado por medio de la misma matriz de adyacencia que G'. En particular, esto quiere decir que si M es una representación cualquiera de G como matriz de adyacencia, entonces existe un ordenamiento de los vértices en V' que genera una representación de G' como matriz de adyacencia dada por M. Más formalmente, definimos la noción de *isomorfismo de grafos* como sigue:

Definición 4.1.2 (Isomorfismo de grafos simples)**.** *Sean $G = (V, E)$ y $G' = (V', E')$ grafos simples. Un* isomorfismo *entre G en G' es una biyección $f : V \to V'$ tal que para todo $(u, v) \in V$ se cumple que:*

$$(u, v) \in E \quad \iff \quad (f(u), f(v)) \in E'.$$

Decimos que G y G' son isomorfos*, denotado $G \cong G'$, si existe isomorfismo entre G y G'.*

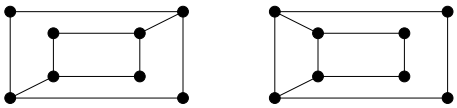
Ejemplo 4.1.4. Los grafos simples que se muestran en la Figura 4.2 son isomorfos. (Encuentre un isomorfismo entre ellos). □

Pregunta. *¿Se le ocurre un algoritmo que determine si dos grafos son isomorfos? ¿Es este algoritmo eficiente? ¿Cree que existan algoritmos eficientes?*

A veces es sencillo demostrar que dos grafos no son isomorfos. Para ello utilizamos un *invariante*; es decir, cualquier propiedad de los grafos que se preserva bajo isomorfismo. La idea es demostrar que uno de los grafos tiene esta propiedad pero el otro no, y que por tanto los grafos no pueden ser isomorfos. Un invariante muy simple es el número de vértices: dos grafos isomorfos deben tener igual número de vértices. Sin embargo, es posible encontrar invariantes más interesantes que nos permiten demostrar que incluso ciertos grafos con el mismo número de vértices no son isomorfos.

Ejemplo 4.1.5. Los grafos de la Figura 4.3 no son isomorfos. Para demostrar esto utilizamos el invariante dado por la existencia de un nodo de grado 1; en efecto, esta propiedad debe ser respetada por un isomorfismo. Claramente, el grafo G tiene un nodo de grado 1 mientras que el grafo H no. Por tanto, $G \ncong H$. □

Ejercicio. *Demuestre que los siguientes grafos no son isomorfos:*



Para ello utilice un invariante.

Ejercicio. *Decimos que un grafo simple $G = (V, E)$ es* autocomplementario *si G es isomorfo a su* complemento *$\bar{G} = (V, \bar{E})$, donde $\bar{E} = (V \times V) \setminus E$. Por ejemlo, el ciclo C_5 es autocomplementario; esto se sigue directamente de observar la Figura 4.2.*

Demuestre que si un grafo simple $G = (V, E)$ con $n > 1$ vértices es autocomplementario, entonces n ó $n - 1$ es divisible por 4.

Ejercicios Finales

1. Demuestre que en todo grafo simple existen al menos dos nodos con el mismo grado.

2. Para este ejercicio necesitamos la siguiente notación:

   - Sean $G = (V, E)$ y $G' = (V', E')$ grafos simples. Un *homomorfismo* de G en G' es una función $h : V \to V'$ tal que para cada par $v_1, v_2 \in V \times V$,

   $$(v_1, v_2) \in E \implies (h(v_1), h(v_2)) \in E'.$$

   - Un grafo simple $G = (V, E)$ es *k-partito* ($k \geq 1$) si su conjunto de vértices V puede ser particionado en a lo más k distintos bloques tal que, para cada arco $e = (v, v') \in E$, se tiene que v y v' pertenecen a diferentes bloques de la partición.
   - El grafo completo en k vértices, denotado K_k, es el grafo simple con k vértices que contiene un arco entre cada par de vértices distintos.

   Demuestre que un grafo simple G es k-partito si y solo si existe un homomorfismo de G en K_k ($k \geq 1$).

3. Asuma que n es un entero par mayor mayor o igual a 4. Sea G un grafo simple con n vértices y estrictamente más de $\frac{n^2}{4}$ arcos. Demuestre que G contiene un triángulo; esto es, existen 3 vértices a, b, c en G tal que (a, b), (b, c) y (a, c) son arcos de G.

4. Sea $G = (V, E)$ un grafo simple bipartito. Decimos que $E' \subseteq E$ es un *matrimonio* en G, si no existen dos arcos en E' que sean incidentes al mismo nodo, y para todo vértice $v \in V$ existe un arco en E' que es incidente a él.

   Para un nodo v en G, defina $N(v)$ como el conjunto de nodos que son adyacentes a v. Si X es un conjunto de vértices en G, entonces definimos $N(X)$ como $\bigcup_{v \in X} N(v)$.

   Sea $G = (V, E)$ un grafo bipartito simple y conexo cuyo conjunto de nodos puede ser particionado en V_1 y V_2 (es decir, todos los arcos en E van desde V_1 a V_2). Demuestre que G contiene un matrimonio si y solo si $|V_1| = |V_2|$ y para cada $X \subseteq V_1$ se tiene que $|X| \leq |N(X)|$.

5. Recuerde que un *clique* es un grafo simple $G = (V, E)$ tal que para cada par de nodos $u, v \in V$ se tiene que $(u, v) \in E$. Dado un grafo simple G decimos que el subgrafo G' de G es un clique *maximal* de G, si G' es un clique y no existe clique G'' tal que G' es un subgrafo propio de G'' y G'' es subgrafo de G.

   Por ejemplo, considere el grafo simple $G = (V, E)$, donde $V = \{v_1, v_2, v_3, v_4\}$ y

   $$E = \{(v_1, v_2), (v_1, v_3), (v_2, v_3), (v_1, v_4)\}.$$

   Entonces el subgrafo G' inducido por el conjunto $\{v_1, v_2\}$ es un clique en G pero no es un clique máximal. Esto porque G' es un subgrafo propio del grafo inducido por vértices $\{v_1, v_2, v_3\}$, el cual también es un clique.

   Sea $\mathcal{I} = \{I_1, \ldots, I_n\}$ un conjunto finito de intervalos de números reales. Definimos el grafo simple *asociado* con $\mathcal{I}$ como $G_{\mathcal{I}} = (V, E)$, donde $V = \{v_1, \ldots, v_n\}$ y para todo $1 \leq i < j \leq n$ se tiene que $(v_i, v_j) \in E$ si y solo si $I_i \cap I_j \neq \emptyset$. Un grafo simple G es de *intervalos* si existe un conjunto finito $\mathcal{I}$ de intervalos de números reales tal que $G = G_{\mathcal{I}}$.

   Sea G un grafo simple de intervalos. Demuestre que existe una enumeración $M_1, M_2, \ldots, M_p$ del conjunto de los cliques máximos de G, de tal forma que si M_i y M_k tienen algún vértice v en común, para algún $1 \leq i < k \leq p$, entonces v pertenece también a M_j, para cada $i \leq j \leq k$.

6. Sea $G = (V, E)$ un grafo simple y asuma que cada par de arcos en G tiene un nodo en común, i.e. para cada par de arcos $e = (u, v), e' = (u', v')$ en E se tiene que $u = u'$ o $u = v'$ o $v = u'$ o $v = v'$.

   Demuestre que G es K_3 o una *estrella*. Esto último quiere decir que E es de la forma

   $$\{(v_1, v), (v_2, v), \ldots, (v_n, v)\}, \qquad \text{para } n \geq 1.$$

7. Un *hipergrafo* $\mathcal{H}$ es un par (V, E), donde V es un conjunto finito de nodos y E es un conjunto de subconjuntos de V. Los elementos de E se denominan *hiperarcos* de $\mathcal{H}$ (es decir, cada hiperarco es un subconjunto de V). Un hipergrafo está *normalizado* si no existen hiperarcos $e_1, e_2 \in E$ tal que e_1 es subconjunto propio de e_2.

   Un ejemplo de un hipergrafo normalizado es $\mathcal{H}_a = (V_a, E_a)$, donde $V_a = \{a, b, c, d, e\}$ y $E_a = \{\{a, b, c\}, \{c, d\}, \{d, e, a\}\}$.

   Decimos que $V' \subseteq V$ *satura* a $\mathcal{H}$ si (i) $V' \cap e \neq \emptyset$ para todo $e \in E$, y (ii) no existe subconjunto propio V'' de V' tal que V'' que cumple la condición (i). Por ejemplo, al hipergrafo $\mathcal{H}_a$ lo saturan los siguientes conjuntos:

   $$\{a, d\} \quad \{a, c\} \quad \{c, e\} \quad \{c, d\} \quad \{d, b\}.$$

   Ningún otro conjunto satura a $\mathcal{H}_a$.

   Para $\mathcal{H} = (V, E)$ definimos $T(\mathcal{H})$ como el hipergrafo cuyo conjunto de nodos es V, y cuyos hiperarcos son exactamente los $V' \subseteq V$ que saturan a V. Por ejemplo, $T(\mathcal{H}_a)$ es el hipergrafo (V_t, E_t), donde $V_t = \{a, b, c, d, e\}$ y $E_t = \{\{a, d\}, \{a, c\}, \{c, e\}, \{c, d\}, \{d, b\}\}$.

   Demuestre que si $\mathcal{H}$ es un hipergrafo normalizado, entonces $\mathcal{H} \subseteq T(T(\mathcal{H}))$.

8. En este ejercicio consideramos grafos no dirigidos que permiten arcos paralelos (es decir, un par de nodos puede estar unido por más de un arco) y *loops* (es decir, arcos que unen a un nodo consigo mismo). Note que un loop en un nodo v incrementa su grado en 2.

   Recuerde que la *secuencia de grados* de un grafo no dirigido con arcos paralelos y loops es la secuencia de los grados de sus nodos escrita en orden no creciente. Sea

   $$d_1, d_2, \ldots, d_n$$

   una secuencia no creciente de enteros positivos cuya suma es par. Demuestre que existe un grafo no dirigido con arcos paralelos y loops cuya secuencia de grados es precisamente $d_1, d_2, \ldots, d_n$.

9. Sea $G = (V, E)$ un grafo simple con $(m-1)n+1$ vértices. Demuestre que este grafo debe contener un subconjunto de m vértices distintos tal que no hay arcos entre ellos, o un vértice que está conectado con al menos n otros vértices.

   Hint: Considere un subconjunto de vértices sin arcos entre ellos que es de tamaño maximal. Analice cómo están conectados estos vértices con el resto del grafo.

4.2. Caminos y conectividad

Los grafos son particularmente importantes por las relaciones que establecen entre distintos objetos. En muchos casos es importante *navegar* tales relaciones, es decir, avanzar recursivamente a través de los *caminos* definidos por los arcos de un grafo, de forma de poder identificar la información implícita que está contenida en él. Por ejemplo, en una red social nos gustaría identificar de qué forma están relacionadas dos personas que no son necesariamente amigas; en una red de transporte nos gustaría determinar cuál es la forma más barata de viajar de un punto a otro; y en una ciudad nos gustaría saber cuál es la forma óptima de repartir el correo. A continuación presentamos formalmente la noción de camino y algunas de sus propiedades matemáticas.

4.2.1. Nociones básicas

Caminos

Definición 4.2.1 (Caminos)**.** *Sea $G = (V, E)$ un grafo (no necesariamente simple). Un* camino *en G es una secuencia:*

$$v_0 e_1 v_1 \ \ldots \ v_{k-1} e_k v_k \qquad (k \geq 0),$$

donde cada v_i es un vértice en V, para $0 \leq i \leq k$, y cada e_i es un arco en E que conecta a v_{i-1} con v_i, para $1 \leq i \leq k$. Decimos que este es un camino de v_0 a v_k.

Un camino es simple *si no repite arcos. Un* circuito *es un camino de v a v, para algún $v \in V$. Un circuito simple se conoce como ciclo.*

Note que nuestra definición permite caminos que consisten de un solo vértice v, para $v \in V$. Este define un camino de v hasta v in G.

Proposición 4.2.1. *Si existe camino de u a v en $G = (V, E)$, entonces existe camino simple de u a v en G.*

Ejercicio. *Demuestre la proposición anterior.*

Note que si $G = (V, E)$ es un grafo simple, entonces todo camino en G puede ser unívocamente asociado con la secuencia de vértices que lo componen (ya que en este caso cada para de vértices está conectado por a lo más un arco).

Invariantes basados en caminos Interesantemente, también es posible utilizar invariantes basados en caminos para demostrar que dos grafos no son isomorfos.

Ejemplo 4.2.1. Los grafos de la Figura 4.4 no son isomorfos. Para demostrar esto utilizamos el invariante dado por la existencia de un ciclo de largo 3; en efecto, esta propiedad debe ser respetada por un isomorfismo. Claramente, el grafo G tiene tal ciclo mientras que el grafo H no. Por tanto, $G \ncong H$. □

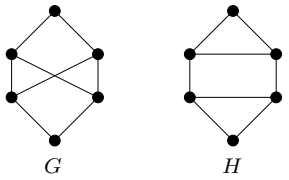


Figura 4.4: Los grafos del Ejemplo 4.2.1.

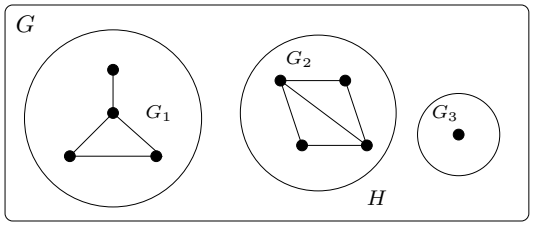


Figura 4.5: El grafo G del Ejemplo 4.2.2.

Conectividad

A continuación definimos la noción de *conectividad* de grafos:

Definición 4.2.2 (Grafos conexos)**.** *Un grafo $G = (V, E)$ es conexo si existe camino de u a v en G, para todo par de vértices $u, v \in V$.*

Si un grafo $G = (V, E)$ no dirigido no es conexo, entonces está formado por la unión disjunta de *componentes conexas*. Estos son los subgrafos G' de G que son *maximalmente conexos*; es decir, aquellos que satisfacen lo siguiente:

- G' es conexo;
- no existe G'' que sea conexo y extienda a G' con más arcos.

Ejemplo 4.2.2. El grafo G que se muestra en la Figura 4.5 consta de tres componentes conexas: G_1, G_2, y G_3. □

Ejercicio. *Demuestre que si $G = (V, E)$ es un grafo simple, entonces todo vértice en V de grado está unido mediante un camino en G a otro vértice distinto de grado impar.*

Caracterización de grafos bipartitos basada en ciclos

Es posible caracterizar los grafos bipartitos como aquellos que no contienen circuitos de largo impar. El *largo* de un circuito $v_0, \ldots, v_k, v_0$, para $k \geq 1$, es $(k + 1)$ (es decir, es el número de arcos que participan del circuito). En particular, este circuito es de largo impar sii k es par.

Proposición 4.2.2. *Un grafo simple $G = (V, E)$ es bipartito sii no contiene circuitos de largo impar.*

Demostración. Si G contiene un circuito $v_0, v_1, \ldots, v_k, v_0$ de largo impar, entonces $\{v_1, v_3, \ldots, v_{k-1}\}$ deben estar a un lado de la partición de los vértices, mientras que $\{v_0, v_2, v_4, \ldots, v_k\}$ deberán estar al otro. Pero por definición $(v_k, v_0) \in E$, lo que implica que existe un arco entre dos vértices del mismo lado de la partición. Concluimos que G no es bipartito.

Asuma, por el contrario, que todos los circuitos de G son de largo par. Suponemos, sin pérdida de generalidad, que G es conexo. La *distancia* entre dos elementos $u, v \in V$ es el largo del camino más corto que conecta a u con v en G. Sea v un nodo arbitrario en V. Construimos entonces dos conjuntos V_1 y V_2 de la siguiente forma:

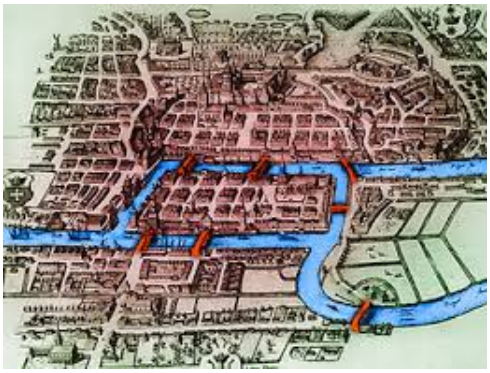


Figura 4.6: La distribución del pueblo de Königsberg.

- En V_1 ponemos todos los elementos que están a distancia par de v (lo que incluye a v por definición).
- En V_2 ponemos los que están a distancia impar de v.

Dado que asumimos que G es conexo, V_1 y V_2 definen una partición de V. Para demostrar que G es bipartito necesitamos demostrar entonces que no hay arcos entre nodos en V_1 ni arcos entre nodos en V_2. Consideremos los dos casos por separado:

- Suponga primero que hay dos nodos u, w en V_1 que son adyacentes. Por definición, existen caminos π_1 y π_2 de largo par de v a u y de v a w en G. Por tanto, el circuito que se obtiene al seguir π_1, luego pasar de v a w mediante el arco que existe entre ellos, y luego seguir π_2 de w a v, es de largo impar. Esto es una contradicción.
- Suponga por otro lado que hay dos nodos u, w en V_2 que son adyacentes. Por definición, existen caminos π_1 y π_2 de largo impar de v a u y de v a w en G. Por tanto, el circuito que se obtiene al seguir π_1, luego pasar de v a w mediante el arco que existe entre ellos, y luego seguir π_2 de w a v, es de largo impar. Esto es una contradicción.

Esto finaliza la demostración. □

4.2.2. Caminos eulerianos y hamitonianos

Circuitos y caminos eulerianos

El pueblo de Königsberg, hoy en Rusia, está dividido en 4 regiones. Tales regiones están conectadas por puentes como se muestra en la Figura 4.6. Durante el Siglo XVIII, los habitantes de Königsberg discutían a menudo si era posible comenzar un paseo en una de estas regiones, cruzar cada puente exactamente una vez, y volver finalmente a la misma región donde se comenzó.

El famoso matemático Euler solucionó este problema. Para ello decidió modelar el problema como un problema de grafos. En efecto, si modelamos el pueblo de Königsberg como un multigrafo $G = (V, E)$, donde los nodos son las regiones, y los arcos son los puentes (ver la Figura 4.7), entonces el problema se reduce a verificar si existe un circuito en G que pase una, y exactamente una vez, por cada arco en E. Esto es lo que conocemos como un *circuito euleriano* de G.

Lo realmente interesante de la solución desarrollada por Euler, es que esta no se concentra en el caso particular del grafo que representa a Königsberg. En vez de eso, Euler desarrolló condiciones generales que permiten identificar cuándo un multigrafo tiene un circuito euleriano. A partir de esto es muy fácil demostrar que el grafo de Königsberg no tiene circuito euleriano, ya que no satisface tales condiciones.

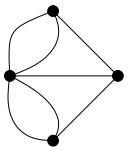


Figura 4.7: El grafo $G = (V, E)$ que representa la distribución de Königsberg.

El grado de un nodo en un multigrafo es el número de arcos que salen de él. El resultado general obtenido por Euler es el siguiente:

Teorema 4.2.1. *Sea $G = (V, E)$ un multigrafo conexo con al menos dos vértices. Entonces G tiene circuito euleriano sii todo nodo en G tiene grado par.*

Dado que en el grafo que representa a Königsberg hay un nodo de grado 3, concluimos que no tiene circuito euleriano. Ahora demostramos el Teorema 4.2.1.

Demostración. Asuma primero que $G = (V, E)$ tiene circuito euleriano. Tal circuito comienza en un nodo v, y al salir de v contribuye en 1 a su grado. De ahí en adelante, cada vez que el circuito pasa a través de un nodo contribuye en dos a su grado (ya que entra por un arco y sale por otro distinto). Finalmente, el circuito vuelve a v a través de un arco distinto a cualquiera que ha usado antes, por lo que vuelve a contribuir en 1 a su grado. En resumen, el circuito contribuye con un valor par al grado de cada nodo en G. Dado que el circuito es euleriano, pasa por todos los arcos, por lo que podemos concluir que cada nodo en G tiene grado par.

Asumamos, en la otra dirección, que cada nodo en V tiene grado par. Construiremos entonces un circuito euleriano de G. El circuito comienza en un nodo cualquiera $v_0 \in V$. A partir de ahí comenzamos a construir un camino simple arbitrario:

$$v_0 e_1 v_1 \dots v_{k-1} e_k v_k$$

tan largo como se pueda. (Recuerde que un camino simple es aquel que no pasa dos veces por el mismo arco). Tal proceso debe detenerse después de un número finito de pasos ya que el grafo es finito. Dado que el camino es maximal en largo, todos los arcos que salen de v_k están entre $\{e_1, \dots, e_k\}$. Interesantemente, es posible demostrar entonces la siguiente propiedad:

Lema 4.3. *Se debe cumplir que $v_0 = v_k$.*

Demostración. Asuma por contradicción que $v_0 \neq v_k$. Luego, este camino ha aportado un valor impar al grado de v_k. Dado que v_k tiene grado par, podemos extender este camino con un arco más sin repetir arcos antes usados. Esto es una contradicción. □

Luego, si el circuito $v_0 e_1 v_1 \dots v_{k-1} e_k v_0$ pasa por todos los arcos de G tenemos listo nuestro circuito euleriano. De otra forma, borramos de G todos los arcos en $\{e_1, \dots, e_k\}$. Al hacer esto obtenemos un nuevo multigrafo G', no necesariamente conexo, donde los vértices siguen teniendo grado par. Esto último se cumple ya que el circuito $v_0 e_1 v_1 \dots v_{k-1} e_k v_0$ contribuye con un valor par al grado de cada arco.

Note que G' y $\{v_0, \dots, v_{k-1}\}$ deben tener al menos un nodo v' en común, ya que G es conexo. Podemos entonces nuevamente comenzar a construir un camino simple tan largo como sea posible a partir de v'. Sabemos por nuestra explicación anterior que tal camino debe terminar en v', y por tanto que es un circuito de G'. Esto nos permite construir un circuito aún más largo de G al "componer" nuestro circuito inicial $v_0 e_1 v_1 \dots v_{k-1} e_k v_0$ con aquel que se ha obtenido desde v' en G'. Esto se muestra gráficamente en la Figura 4.8.

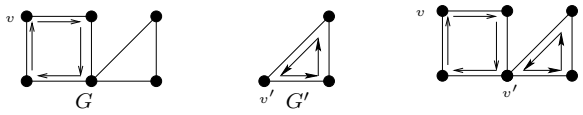


Figura 4.8: El multigrafo G y el circuito inicial; el multgrafo G' y el segundo circuito; el grafo G y el circuito euleriano final.

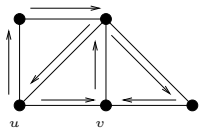


Figura 4.9: Este grafo simple time un camino euleriano desde u hasta v, pero no tiene circuito euleriano.

Debería ser claro para terminar, que al seguir recursivamente este proceso hasta que todos los arcos hayan sido usados obtenemos un circuito euleriano de G. Por tanto, si todos los vértices en G tienen grado par, entonces G tiene circuito euleriano. □

Análogamente, es posible definir el concepto de *camino euleriano*. Este es un camino en un multigrafo que pasan por cada arco una, y exactamente una, vez (aunque podría terminar en un nodo distinto al nodo donde comienza). Por ejemplo, el grafo que se muestra en la Figura 4.8 tiene un camino euleriano. Por otro lado, este grafo no tiene circuito euleriano ya que dos de sus nodos tienen grado impar.

Note que si un grafo tiene un camino, pero no un ciruito euleriano, entonces tiene exactamente dos vértices de grado impar (en particular, estos corresponden al nodo inicial y final de tal camino). Una simple modificación de la demostración anterior nos permite establecer que esta condición no solo es necesaria, sino también suficiente para caracterizar la existencia de caminos eulerianos:

Teorema 4.3.1. *Sea $G = (V, E)$ un multigrafo conexo con al menos dos vértices. Entonces G tiene camino, pero no un circuito euleriano sii tiene exactamente dos nodos de grado impar.*

En consecuencia, el grafo de Königsberg tampoco tiene camino euleriano.

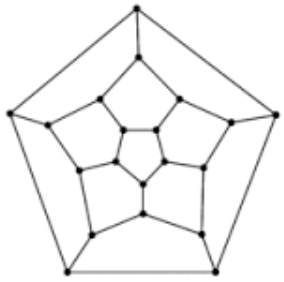
Una importante consecuencia de los Teoremas 4.2.1 y 4.3.1 es que existe un algoritmo eficiente que permite verificar si un multigrafo $G = (V, E)$ tiene un circuito (resp., camino) euleriano. Basta computar los grados de los nodos en V, y luego verificar que el valor de cada uno de estos grados es par (resp., solamente dos de ellos tienen grado impar).

Circuitos y caminos hamiltonianos

Los caminos eulerianos son caminos simples que pasan por cada arco de un multigrafo $G = (V, E)$ exactamente una vez. ¿Qué pasa si ahora queremos determinar si algún camino simple pasa por cada vertice en V exactamente una vez? (Note que en el caso particular en que $G = (V, E)$ es un grafo simple esta condición es equivalente a que un camino cualquiera pase exactamente una vez por cada vértice). Este tipo de caminos se conoce como *hamiltoniano*, y constituyen un objeto de estudio recurrente en teoría de grafos y ciencia de la computación. Un circuito hamiltoniano es un camino hamiltoniano que termina en el mismo vértice en el que comienza.

Ejercicio. *Demuestre que el grafo simple que se muestra a continuación:*

tiene un circuito hamiltoniano.



Al contrario del caso de los caminos y circuitos eulerianos, no se conoce una caracterización útil de cuando un grafo tiene un circuito o camino hamiltoniano. Por útil nos referimos a una que permita detectar eficientemente la existencia o no existencia de tal camino. Basándonos en resultados fundamentales de la teoría de complejidad computacional, podemos decir que la gran mayoría de los científicos de la computación cree que tal caracterización no existe.

Ejercicios Finales

1. Demuestre que un grafo $G = (V, E)$ es conexo si y solo si para cada par de subconjuntos no vacíos de vértices U y W tales que $U \cup W = V$ y $U \cap W = \emptyset$, existe una arista en E que une a un vértice de U con un vértice de W.

2. Un *torneo* es un grafo simple y dirigido que cumple que para cada par u, v de vértices distintos en el grafo se tiene que exactamente uno de los pares (u, v) y (v, u) es un arco en el grafo.

   Demuestre usando inducción que todo torneo contiene un camino hamiltoniano. Recuerde que un camino hamiltoniano en este caso es un camino simple y dirigido que visita a cada vértice del grafo.

3. Demuestre que todo grafo simple con n vértices y estrictamente más de $\binom{n-1}{2}$ arcos tiene que ser conexo.

   Hint: Demuestre que si G tiene al menos dos componentes conexas, y una de sus componentes conexas tiene $k < n$ vértices, entonces G tiene a lo más $\binom{n-1}{2}$ arcos.

4. Sea G un grafo simple. Demuestre que G o su complemento, $\bar{G}$, es conexo.

5. Asuma que $G = (V, E)$ es un grafo simple no dirigido. Demuestre que si $|E| > \binom{|V|-1}{2}$ entonces G es conexo.

6. La *distancia* entre dos nodos distintos v y v' de un grafo simple $G = (V, E)$ es el largo (número de arcos) del camino más corto en G entre v y v'. El diámetro de G es la máxima distancia existente entre dos nodos distintos.

   Demuestre que si el diámetro de un grafo simple G es ≥ 4, entonces el diámetro de su complemento es ≤ 2.

7. *a*) Sea G un grafo simple y conexo con al menos 2 nodos. Demuestre que existe un nodo v en G tal que si se saca a v de G (junto a todos los arcos incidentes a v) el grafo sigue siendo conexo.

   b) Sea $G = (V, E)$ un grafo simple. Demuestre que G contiene un camino cuyo largo es al menos $\min\{\deg(v) \mid v \in V\} - 1$.

8. Defina el *grosor* de un grafo simple G como el menor número de subgrafos planares de G tal que su unión es G.

a) Demuestre que si $G = (V, E)$ es simple y conexo entonces su grosor es al menos $\lceil \frac{|E|}{(3 \cdot |V| - 6)} \rceil$.

b) Demuestre que el grosor de K_n es al menos $\lfloor \frac{(n+7)}{6} \rfloor$, para n un entero positivo.

9. Sea $\Sigma = \{1, \ldots, p\}$. Una (p, n) *secuencia de Bruijn* es una palabra $a_1 a_2 \ldots a_L \in \Sigma^*$ tal que $L = p^n$ y todas sus subsecuencias $a_i a_{i+1} \ldots a_{i+n-1}$ de largo n son distintas. En este caso la suma de los índices se realiza módulo L, y por tanto la secuencia es un orden circular. Esto quiere decir, en particular, que para toda palabra w sobre Σ de largo n se satisface que $w = a_i a_{i+1} \ldots a_{i+n-1}$, para algún $1 \leq i \leq L$.

   El *diagrama de Bruijn* $D_{p,n}$ es un grafo dirigido cuyos vértices corresponden a las palabras $a_1 a_2 \ldots a_{n-1}$ de largo $n-1$ sobre Σ. Desde cada vértice $a_1 a_2 \ldots a_{n-1}$ en $D_{p,n}$ salen exactamente p arcos $e_1, \ldots, e_p$, de tal forma que el arco e_i apunta al vértice $a_2 a_3 \ldots a_{n-1} i$, para todo $1 \leq i \leq p$.

   a) ¿Cuántos vértices y arcos tiene $D_{n,p}$?

   b) Demuestre que $D_{n,p}$ es fuertemente conexo; es decir, que todo par de vértices en $D_{n,p}$ está conectado por un camino dirigido.

   c) Un circuito *euleriano* de $D_{n,p}$ es un circuito (dirigido) que pasa por cada arco de $D_{n,p}$ exactamente una vez. Demuestre que es posible construir una (p, n) secuencia de Bruijn desde cada circuito euleriano de $D_{n,p}$.

   d) ¿Es cierto que siempre existe al menos una (p, n) secuencia de Bruijn?

10. Decimos que un vértice v en un grafo G es *crítico*, si sacar a v de G (junto con todos los arcos que llegan a tal nodo) genera más de una componente conexa (es decir, el grafo deja de ser conexo).

    Demuestre que todo grafo simple y conexo contiene al menos dos vértices que no son críticos.

    Hint: Para el caso inductivo, suponga que G tiene un nodo crítico; es decir, al ser sacado del grafo genera al menos dos componentes conexas G_1 y G_2. Por HI ambos G_1 y G_2 tienen dos vértices que no son críticos. Asuma que tales vértices corresponden a u_1, u_2 en G_1 y $w_1, w_2 \in G_2$. Demuestre que existen $i, j \in \{1, 2\}$ tal que ni v_i ni w_j es crítico.

11. Suponga que G no tiene circuito hamiltoniano. Comience a agregar arcos arbitrariamente a G hasta llegar a un grafo H que contiene tal circuito (este H debe existir ya que el grafo completo en n vértices tiene circuito hamiltoniano). Sea (a, b) el último arco que agregamos en este proceso H. Por tanto, si sacamos el arco (a, b) de H obtenemos un grafo H' que no contiene circuito hamiltoniano.

    Sea C un circuito hamiltoniano en H. Entonces C debe pasar por (a, b). Suponga entonces sin pérdida de generalidad que el ciclo es de la forma:

    $$(v_1 = a) \leftrightarrow (v_2 = b) \leftrightarrow v_3 \leftrightarrow \cdots \leftrightarrow v_n \leftrightarrow (v_1 = a),$$

    donde $\leftrightarrow$ denota que vértices consecutivos son adyacentes.

    a) Demuestre que si b es adyacente a algún v_i en H', para $3 \leq i \leq n$, entonces a no puede ser adyacente a v_{i-1} en H'.

    Hint: Demuestre que si esto ocurre entonces H' tiene circuito hamiltoniano.

    b) A partir de lo anterior, demuestre que $\text{grado}_{H'}(a) + \text{grado}_{H'}(b) < n$, donde $\text{grado}_{H'}$ denota el grado de un nodo en H'.

    c) Concluya que si $\text{grado}_G(x) + \text{grado}_G(y) \geq n$ para todo par de vértices no adyacentes en G tal que $x \neq y$, entonces G tiene circuito hamiltoniano. (Como antes, grado_G denota el grado de un nodo en G).

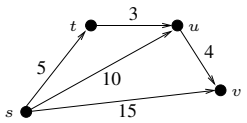


Figura 4.10: El grafo $G = (V, E)$ con pesos utilizado en el Ejemplo 4.4.1.

4.4. Algoritmos inductivos en grafos

La naturaleza recursiva de los caminos en grafos hace particularmente apropiada la utilización de algoritmos inductivos para solucionar problemas relacionados con tales caminos. Uno de los problemas más famosos que aceptan tal solución inductiva es el de los *caminos más cortos*. La entrada de este problema consiste en un grafo simple pero dirigido $G = (V, E)$, un nodo distinguido $s \in V$, y una función $\lambda : E \to (\mathbb{N} \setminus \{0\})$ que asocia un *peso*, i.e., un natural distinto de 0, a cada arco en E. El *peso de un camino en* G corresponde a la suma de los pesos en los arcos de tal camino. El problema consiste en determinar el valor $\mathsf{peso}(v)$, para cada $v \in V$, el que corresponde al mínimo peso de un camino que conecta s con v en G. Si tal camino no existe definimos $\mathsf{peso}(v) := \infty$.

Este tipo de problemas tiene una gran cantidad de aplicaciones prácticas. Típicamente, los pesos en los arcos representan costos de comunicación entre dos nodos (por ejemplo, en una red de computadores o en una red de transporte). Por tanto, el problema de los caminos más cortos busca minimizar el costo de acceso a cada uno de los nodos en el grafo desde un nodo inicial predeterminado.

Aquí presentamos un algoritmo inductivo muy famoso que resuelve el problema de los caminos más cortos. Este se llama el *Algoritmo de Dijkstra*, en honor a su creador, el científico de la computación holandés Edgar Dijkstra. Considere una entrada a nuestro problema dada por el grafo simple y dirigido $G = (V, E)$, el nodo distinguido $s \in V$, y la función de pesos $\lambda : E \to (\mathbb{N} \setminus \{0\})$. El algoritmo opera en pasos $1, \ldots, |V|$. En cada paso i, con $1 \leq i \leq |V|$, el algoritmo almacena un conjunto de nodos S_i de tamaño i. Para cada nodo $v \in S_i$, el algoritmo ya conoce el valor $\mathsf{peso}(v)$. Más aún, para cada $u \in V \setminus S_i$ el algoritmo almacena el valor $\mathsf{temp}_i(u)$, que corresponde al mínimo peso de un camino de s a u en G que solo avanza por nodos en S_i antes de llegar a u. Estos valores se determinan inductivamente de la siguiente manera:

- $S_1 = \{s\}$, $\mathsf{peso}(s) = 0$, y para cada $u \in V \setminus \{s\}$:

$$\mathsf{temp}_1(u) \;=\; \lambda((s, u)),$$

  asumiendo que $\lambda((s, u)) = \infty$ si $(s, u) \notin E$.

- Considere un i con $1 \leq i < |V|$, y asuma que v es un elemento en $V \setminus S_i$ cuyo valor $\mathsf{temp}_i(v)$ es mínimo. Definimos entonces $S_{i+1} = S_i \cup \{v\}$ y $\mathsf{peso}(v) = \mathsf{temp}_i(v)$. Además, para cada $u \in V \setminus S_{i+1}$ definimos:

$$\mathsf{temp}_{i+1}(u) \;=\; \text{mín}\,\{\mathsf{temp}_i(u), \mathsf{peso}(v) + \lambda((v, u))\},$$

  asumiendo que $\lambda((v, u)) = \infty$ si $(v, u) \notin E$.

Antes de demostrar la correctitud del algoritmo, mostramos un ejemplo de cómo este opera en un caso concreto.

Ejemplo 4.4.1. Considere el grafo $G = (V, E)$ con pesos que se muestra en la Figura 4.10. Por definición:

$$S_1 = \{s\}; \quad \mathsf{peso}(s) = 0; \quad \mathsf{temp}_1(t) = 5; \quad \mathsf{temp}_1(u) = 10; \quad \mathsf{temp}_1(v) = 15.$$

Para el paso 2 buscamos el elemento en $V \setminus S_1$ que minimiza el valor de temp_1. Este es t. Luego, computamos recursivamente:

$$S_2 = \{s,t\}; \quad \mathsf{peso}(s) = 0; \quad \mathsf{peso}(t) = 5;$$
$$\mathsf{temp}_2(u) = \text{mín}\{\mathsf{temp}_1(u),\, \mathsf{peso}(t) + \lambda(t,u)\} = \text{mín}\{10, 5+3\} = 8;$$
$$\mathsf{temp}_2(v) = \text{mín}\{\mathsf{temp}_1(v),\, \mathsf{peso}(t) + \lambda(t,v)\} = \text{mín}\{15, 5+\infty\} = 15.$$

Para el paso 3, buscamos el elemento en $V \setminus S_2$ que minimiza el valor de temp_2. Este es u. Luego, computamos recursivamente:

$$S_3 = \{s,t,u\}; \quad \mathsf{peso}(s) = 0; \quad \mathsf{peso}(t) = 5; \quad \mathsf{peso}(u) = 8;$$
$$\mathsf{temp}_3(v) = \text{mín}\{\mathsf{temp}_2(v),\, \mathsf{peso}(u) + \lambda(u,v)\} = \text{mín}\{15, 8+4\} = 12.$$

Finalmente, en el paso 4 buscamos el elemento en $V \setminus S_3$ que minimiza el valor de temp_3. Este es v. Luego, computamos recursivamente:

$$S_4 = \{s,t,u,v\}; \quad \mathsf{peso}(s) = 0; \quad \mathsf{peso}(t) = 5; \quad \mathsf{peso}(u) = 8; \quad \mathsf{peso}(v) = 12.$$

Es posible constatar que $\mathsf{peso}(w)$ corresponde al mínimo peso de un camino des s a w en G, para cada $w \in \{s,t,u,v\}$. □

Ahora demostramos la correctitud del algoritmo. Dado que $V = S_{|V|}$, es decir, en el último paso el algoritmo ya ha computado el valor de $\mathsf{peso}(v)$ para cada $v \in V$, basta demostrar lo siguiente:

Proposición 4.4.1. *Para cada $1 \leq i \leq |V|$ se cumple que:*

- $\mathsf{peso}(v)$ *corresponde al peso mínimo de un camino en G desde s hasta v, para cada $v \in S_i$.*
- $\mathsf{temp}_i(u)$ *corresponde al peso mínimo de un camino en G desde s hasta u que solo pasa por nodos intermedios en S_i, para cada $u \in V \setminus S_i$.*

Demostración. La demostración es por inducción en i. El caso base es $i = 1$. Entonces $S_1 = \{s\}$. Por definición, $\mathsf{peso}(s) = 0$, lo que corresponde al mínimo peso de un camino de s hasta s en G. Además, $\mathsf{temp}_1(u) = \lambda((s,u))$ para cada $u \in V \setminus S_1$. Claramente entonces $\mathsf{temp}_1(u)$ corresponde al valor mínimo del peso de un camino en G desde s hasta u y que solo pasa por nodos intermedios en $S_1 = \{s\}$.

Consideremos ahora el caso inductivo $(i+1)$, para $1 \leq i < |V|$. Por definición, $S_{i+1} = S_i \cup \{v\}$, donde v es un vértice en $V \setminus S_i$ cuyo valor $\mathsf{temp}_i(v)$ es mínimo. Necesitamos demostrar dos cosas:

1. $\mathsf{peso}(v)$ corresponde al peso mínimo de un camino en G desde s hasta v.
2. $\mathsf{temp}_{i+1}(u)$ corresponde al peso mínimo de un camino en G desde s hasta u que solo pasa por nodos intermedios en S_{i+1}, para cada $u \in V \setminus S_{i+1}$.

Comencemos con (1). Por definición, $\mathsf{peso}(v) = \mathsf{temp}_i(v)$. Por HI, por otro lado, $\mathsf{temp}_i(v)$ corresponde al peso mínimo de un camino en G desde s hasta v y que solo pasa por nodos intermedios en S_i. Asuma, por contradicción, que existe camino de s a v en G de peso $p < \mathsf{temp}_i(v)$. Considere entonces un camino π cualquiera de peso p desde s hasta v en G. Por suposición, tal camino debe pasar por algún nodo intermedio que no está en S_i. Sea x el primer tal nodo que aparece en el camino. Definimos π_x como la restricción de π que llega hasta la primera aparición de x. Esta situación puede observarse en la Figura 4.11.

Dado que π_x es un prefijo del camino π, el peso p' de π_x es menor o igual que el peso de π, es decir, p. Pero π_x es un camino que llega a x solo pasando por nodos intermedios en S_i. Esto quiere decir que $\mathsf{temp}_i(x) \leq p' \leq p < \mathsf{temp}_i(v)$, ya que por HI sabemos que $\mathsf{temp}_i(x)$ corresponde al peso mínimo de un

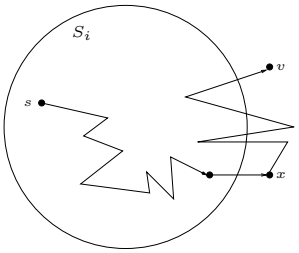


Figura 4.11: Ilustración de la construcción utilizada en la Demostración de la Proposición 4.4.1.

camino de s a x en G que solo pasa por nodos intermedios en S_i. Esto contradice la forma en que v fue elegido.

Ahora seguimos con (2). Por definición:

$$\mathsf{temp}_{i+1}(u) \;=\; \text{mín}\,\{\mathsf{temp}_i(u), \mathsf{peso}(v) + \lambda((v,u))\},$$

para cada $u \in V \setminus S_{i+1}$. Por HI, por otro lado, $\mathsf{temp}_i(u)$ corresponde al peso mínimo de un camino en G desde s hasta v que solo pasa por nodos intermedios en S_i. Además, como ya demostramos, $\mathsf{peso}(v)$ es el peso mínimo de un camino que va de s a v en G.

Considere el conjunto Π de aquellos caminos π de s a u en G que solo pasan por nodos intermedios en S_{i+1} y cuyo peso es mínimo. Sea p el peso de los caminos en Π. Hay dos posibilidades: (a) algún $\pi \in \Pi$ pasa solo por nodos intermedios en S_i, o (b) todos los caminos $\pi \in \Pi$ pasan por v (ya que $S_{i+1} = S_i \cup \{v\}$). En el caso (a) tenemos entonces que algún $\pi \in \Pi$ es también un camino de peso mínimo de s a u en G que pasa solo por nodos intermedios en S_i, y por tanto que el peso de π, es decir, p, es precisamente $\mathsf{temp}_i(u)$ por HI. En el segundo caso podemos demostrar lo siguiente: Sea π cualquiera de los caminos en Π, y asuma que π_v la restricción de π que llega hasta la primera aparición de v. Luego π es la concatenación de π_v con el arco que lleva de v a u. Asumiendo esto tenemos entonces que el peso de π es el peso de π_v más $\lambda((v,u))$. Por HI el peso de π_v es $\mathsf{temp}_i(v)$, ya que este camino solo pasa por nodos intermedios en S_i y es de peso mínimo. Como por definición $\mathsf{temp}_i(v) = \mathsf{peso}(v)$, esto quiere decir que el peso de π (es decir, p) coincide con $\text{mín}\,\{\mathsf{temp}_i(u), \mathsf{peso}(v) + \lambda((v,u))\}$, y que por tanto nuestra definición de $\mathsf{temp}_{i+1}(u)$ es correcta.

Falta entonces demostrar que efectivamente π es la concatenación de π_v con el arco que lleva de v a u. Asuma, por contradicción, que esto no es así, y por ende que luego de pasar por v por primera vez el camino π vuelve a S_i. Note que luego de pasar por v el camino π debe quedarse en S_i antes de llegar a u (en caso contrario pasaría dos veces por v y su peso no sería minimal). Sea $x \in S_i$ el último elemento que visita π antes de llegar a u, y asuma que π_x es la restricción de π hasta llegar a x. Al momento de haber incluido a x en S_j, para $1 \leq j \leq i$, actualizamos el valor de $\mathsf{peso}(x) = \mathsf{temp}_{j-1}(x)$. Sabemos por HI que $\mathsf{peso}(x)$ es el peso mínimo de un camino de s a x en G y que $\mathsf{temp}_{j-1}(x)$ es el peso mínimo de un camino en G de s a x que solo pasa por nodos intermedios en $S_{j-1} \subseteq S_i$. Esto quiere que hay un camino π^* de peso mínimo de s a x en G que solo pasa por nodos intermedios en S_i. En particular, el peso de π^* es menor o igual al peso de π_x. Considere entonces el camino que sigue π^* y luego avanza por el arco (x,u). El peso de tal camino es menor o igual al de π, puesto que π es la concatenación de π_x con el arco (x,u). Esto quiere decir que existe camino de s a u de peso menor o igual a π que solo pasa por nodos intermedios en S_i. Esto es una contradicción con nuestra suposición en (b). □

Ejercicio. *Analice el costo computacional del Algoritmo de Dijkstra.*

Ejercicios Finales

1. El propósito del siguiente algoritmo es computar caminos de menor peso en grafos dirigidos cuyos pesos pueden ser negativos. En particular, el peso de un camino entre dos vértices puede ser negativo

y existen casos en que el peso del camino de menor peso entre dos vértices puede ser $-\infty$ (cuando existen circuitos dirigidos de peso negativo).

Sea $G = (V, E)$ un grafo dirigido (sin loops ni multiarcos) con $n \geq 1$ vértices y s un vértice en V. Asuma que $w : E \to \mathbb{Z}$ una función que asigna un peso $w(u, v)$ a cada arco $(u, v) \in E$. Nuestro algoritmo computa iterativamente un valor $d(v)$ para cada vértice $v \in V$ de la siguiente forma:

a) Inicialmente $d(s) := 0$ y $d(v) := \infty$ para cada $v \in V$ que no es s.

b) Luego se realizan $n - 1$ iteraciones del siguiente proceso: Por cada arco $(u, v) \in E$, si $d(v) > d(u) + w(u, v)$ entonces se cambia el valor de $d(v)$ a $d(u) + w(u, v)$.

c) Finalmente, por cada arco $(u, v) \in E$ se verifica si $d(v) > d(u) + w(u, v)$. De ocurrir esto para algún arco, se retorna el valor `falso` y se detiene. De no ser así, se retorna el valor `verdadero` y se detiene.

Demuestre lo siguiente:

a) Si G no contiene un circuito dirigido de peso negativo que pueda ser alcanzado desde s, entonces al terminar el paso (b) se cumple lo siguiente para cada $v \in V$ tal que existe camino dirigido de s a v en G: El valor de $d(v)$ es el peso del camino de menor peso de s a v en G.

b) Existe camino dirigido de s a v en G si y solo si $d(v) < \infty$ al terminar el algoritmo.

c) Si G no contiene circuitos de peso negativo que puedan ser alcanzados desde s, entonces el valor retornado es `verdadero`. En caso contrario, el algoritmo retorna el valor `falso`.

4.5. Planaridad

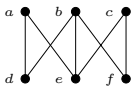
4.5.1. Definición y un resultado básico

Un grafo $G = (V, E)$ es *planar* si puede ser dibujado en el plano sin que se crucen sus arcos. Este concepto juega un rol importante, por ejemplo, en el diseño de circuitos digitales. En tal caso queremos traspasar un circuito dado a una placa plana. Esto solo puede realizarse si el grafo que representa al circuito es planar.

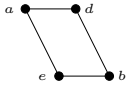
Es posible demostrar que algunos grafos importantes no son planares:

Proposición 4.5.1. *El grafo bipartito $K_{3,3}$ que se muestra en la Figura 4.1 no es planar.*

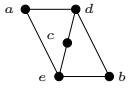
Demostración. Asuma que nombramos los grafos de $K_{3,3}$ como se muestra a continuación:



Suponga, por contradicción, que es posible hacer una representación planar de $K_{3,3}$. De cualquier forma en que dispongamos a los vértices $\{a, b, d, e\}$ en tal representación, estos deben dividir al plano en dos regiones, una *interna* y otra *externa*, de la siguiente forma:



Hay dos posibilidades entonces para colocar al nodo c: en la región interna o en la externa. Suponga primero que c se halla en la región interna, como se muestra a continuación:



Esto genera una nueva región interna. El problema es dónde colocamos ahora al vértice f. Si lo hacemos en la región externa entonces debemos pasar por arriba de algún arco para conectar a f con c. Si lo hacemos en una de las regiones internas entonces debemos cruzar al menos un arco para conectar a f ya sea con a o con b.

El análisis del caso en que c se coloca en la región externa es similar, y se deja como ejercicio. □

Un análisis similar permite demostrar también que K_5 no es planar:

Proposición 4.5.2. *El grafo K_5 no es planar.*

4.5.2. Caracterización de los grafos planares

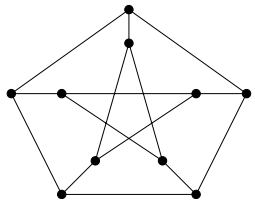
El matemático Kuratowski desarrolló una caracterización de cuando un grafo simple $G = (V, E)$ es planar. Esta caracterización expresa, de una manera precisa, que los grafos que no son planares son precisamente los que extienden algún grafo que puede ser obtenido desde $K_{3,3}$ o K_5 utilizando ciertas operaciones básicas.

La caracterización está basada en la noción de *homeomorfismo* que explicamos a continuación. Una *subdivisión elemental* de un arco $(u, v) \in E$ se obtiene agregando un nuevo vértice w y reemplazando el arco (u, v) por los arcos (u, w) y (w, v). Es fácil observar que esta operación preserva la planaridad de un grafo. Dos grafos simples $G = (V, E)$ y $G' = (V', E')$ son *homeomorfos* si ambos pueden ser obtenidos desde el mismo grafo mediante una secuencia de divisiones elementales. La caracterización expresa entonces lo siguiente:

Teorema 4.5.1. *Un grafo simple $G = (V, E)$ es planar sii no contiene un subgrafo que sea homeomórfico a $K_{3,3}$ o K_5.*

Una de las direcciones de este teorema es fácil de demostrar: Si G contiene un subgrafo homeomórfico a $K_{3,3}$ o K_5 entonces no es planar (porque si lo fuera entonces $K_{3,3}$ o K_5 también lo sería). La otra dirección es bastante complicada y escapa a los contenidos de este curso.

Ejercicio. *Demuestre qiue el grafo que se muestra a continuación no es planar:*



Este se conoce como el grafo de Petersen.

4.5.3. La fórmula de Euler

La representación planar de un grafo simple y conexo divide al plano en regiones: varias internas y una externa. El resultado fundamental de Euler en esta área es que todas las representaciones planares de un grafo simple y conexo dividen al plano en la misma cantidad de regiones. Además, tal número depende linealmente del número de vértices y arcos en el grafo. En particular:

Teorema 4.5.2. *Considere una representación planar de un grafo simple y conexo $G = (V, E)$. Asuma que $|V| = n$, $|E| = m$, y r es el número de regiones generadas por la representación planar. Entonces:*

$$r \;=\; m - n + 2.$$

Demostración. Considere una representación planar cualquiera de G. Asuma que $e_1, \ldots, e_m$ una enumeración de los arcos en E que satisface la siguiente propiedad para cada $1 \leq i \leq m$: El grafo G_i, que se obtiene desde

G al dejar solo los arcos en $\{e_1, \ldots, e_i\}$ y los vértices que aparecen en tales arcos, es conexo. Tal enumeración siempre existe ya que hemos asumido que G es conexo. Note en particular que $G_m = G$.

Para cada $1 \leq i \leq m$ definimos r_i, n_i y m_i como el número de regiones, vértices y arcos en G_i, respectivamente. Demostraremos por inducción que lo siguiente se cumple para cada $1 \leq i \leq m$:

$$r_i \ = \ m_i - n_i + 2.$$

Esto basta para demostrar el resultado ya que sabemos que $G_m = G$.

El caso base $i = 1$ corresponde al grafo G_i formado tan solo por el arco e_1. En tal caso tenemos que $r_1 = 1$, $n_1 = 2$, y $m_1 = 1$. Por tanto, $r_1 = m_1 - n_1 + 2$.

Consideremos ahora el caso inductivo $(i + 1)$, para $1 \leq i < m$. Por HI tenemos que $r_i = m_i - n_i + 2$. Debemos considerar dos casos. El primer caso ocurre cuando al agregar e_{i+1} a G_i para formar G_{i+1} se obtiene una nueva región. Esto solo puede pasar si e_{i+1} conecta a dos vértices que ya estaban en G_i. Es decir, $r_{i+1} = r_i + 1$, $n_{i+1} = n_i$, y $m_{i+1} = m_i + 1$. Utilizando la HI es fácil demostrar entonces que $r_{i+1} = m_{i+1} - n_{i+1} + 2$.

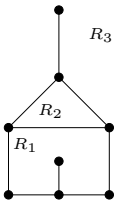
El segundo caso ocurre cuando al agregar e_{i+1} a G_i para formar G_{i+1} no se obtiene una nueva región. Esto solo puede pasar si e_{i+1} conecta un vértice en G_i con un vértice nuevo. Es decir, $r_{i+1} = r_i$, $n_{i+1} = n_i + 1$, y $m_{i+1} = m_i + 1$. Nuevamente, utilizando la HI es fácil demostrar que $r_{i+1} = m_{i+1} - n_{i+1} + 2$. □

Como corolario de Teorema 4.5.2 obtenemos el siguiente resultado de importancia, el que establece una cota para el número de arcos en un grafo planar:

Corolario 4.5.1. *Sea $G = (V, E)$ un grafo simple y conexo que es planar. Suponga que G tiene n vértices y m arcos, y que $n \geq 3$ vértices. Entonces:*

$$m \ \leq \ 3n - 6.$$

Demostración. Consideremos una representación planar cualquiera de G. Definamos el *grado de una región* R en tal representación, denotado por $\mathsf{grado}(R)$, como el número de arcos que aparecen en el borde interno de una región. Cuando un arco aparece dos veces en tal borde contribuye en 2 al grado de la región. Por ejemplo, en el siguiente grafo:



tenemos que $\mathsf{grado}(R_1) = 7$, $\mathsf{grado}(R_2) = 3$, y $\mathsf{grado}(R_3) = 7$.

Debería ser claro entonces que si $\mathcal{R}$ es el conjunto de regiones en la representación planar de G, se tiene que:

$$\sum_{R \in \mathcal{R}} \mathsf{grado}(R) \ = \ 2m,$$

donde m es el número de arcos en E. Esto ya que cada arco contribuye al borde de una región exactamente dos veces (ya sea en dos regiones diferentes, o dos veces en la misma región). Dado que cada región tiene grado al menos 3, se sigue que:

$$\sum_{R \in \mathcal{R}} \mathsf{grado}(R) \ = \ 2m \ \geq \ 3r,$$

donde r es el número de regiones en $\mathcal{R}$. Es decir, $r \leq \frac{2m}{3}$.

Del Teorema 4.5.2 sabemos que $r = m - n + 2$, por lo que:

$$m - n + 2 \ \leq \ \frac{2m}{3}.$$

Despejando m obtenemos que:

$$m \leq 3n - 6.$$

Esto finaliza la demostración. □

Note que una consecuencia inmediata de este corolario es que K_5 no es planar. En efecto, tal grafo tiene 5 vértices y $\binom{5}{2} = 10$ arcos. Por tanto, no cumple la condición expresada en el Corolario 4.5.1.

Otro corolario sumamente importante del Teorema 4.5.2 es el siguiente:

Corolario 4.5.2. *Todo grafo simple y conexo que es planar tiene un vértice de grado a lo más 5.*

Demostración. Si el grafo tiene 1 o 2 vértices entonces esto se cumple trivialmente. Asuma entonces que el grafo tiene $n \geq 3$ vértices, y suponga por contradicción que todos sus nodos tienen grado mayor o igual a 6. Por tanto:

$$\sum_{v \in V} \mathsf{grado}(v) \geq 6n.$$

Por otro lado, por Teorema de los Saludos sabemos que:

$$\sum_{v \in V} \mathsf{grado}(v) = 2m,$$

donde m es el número de arcos en el grafo. Es decir, $2m \geq 6n$, i.e., $m \geq 3n$, lo que contradice al Corolario 4.5.1. □

4.6. Colorabilidad

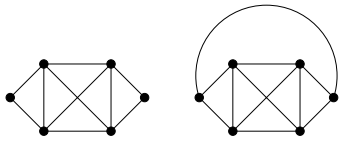
4.6.1. Definición

Sea $k \geq 1$ un entero. Decimos que un grafo simple $G = (V, E)$ es *k-coloreable* si es posible asignarle uno de entre k colores a cada vértice en V, de tal forma que si dos vértices son adyacentes entonces reciben colores distintos. Los grafos pueden representar diferentes aplicaciones en las que el problema de coloración es importante. Por ejemplo, si nuestro grafo representa el mapa de un continente – donde los vértices corresponden a los países y los arcos la existencia de un borde común – entonces una coloración representa una manera de pintar el mapa de forma que países vecinos reciban colores distintos. Por otro lado, si nuestro grafo consiste de vértices que representan cursos y arcos que establecen cuando dos cursos tienen un alumno en común, entonces podemos entender una forma de asignarle horarios a los exámenes de tales cursos, sin que exista tope de horario para los alumnos, como una coloración.

Siempre es posible colorear un grafo asignándole un color distinto a cada nodo. Sin embargo, en algunos casos es posible utilizar menos colores para colorear un grafo. Por ejemplo, en términos de la notación introducida recién, la Proposición 4.1.1 expresa que todos los grafos bipartitos son 2-coloreables (y que, de hecho, esos son los únicos grafos 2-coloreables). Típicamente estamos interesados en colorear un grafo con el menor número de colores posibles. Definimos entonces el siguiente concepto:

Definición 4.6.1 (Número cromático)**.** *Sea $G = (V, E)$ un grafo simple. El* número cromático de G*, denotado $\chi(G)$, corresponde al menor k tal que G es k-coloreable.*

Ejercicio. *Determine el número cromático de los siguiente grafos:*



Ejercicio. *Sea $G = (V, E)$ un grafo con n vértices, donde $n \geq 1$. ¿Es cierto que $\chi(G) = n$ si, y solo si, $G = K_n$?*

4.6.2. Colorabilidad y planaridad

Todo mapa puede ser representado como un grafo planar. Por tanto, el problema de cuántos colores son necesarios para colorear un mapa cualquiera es equivalente a cuántos colores necesitamos para colorear un grafo planar. Interesantemente, los grafos planares tienen un número cromático de a lo más 4, como establece el siguiente famoso resultado de la teoría de grafos:

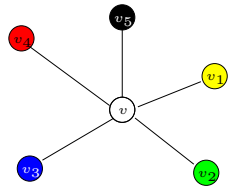
Teorema 4.6.1 (Teorema de los 4 colores)**.** *Todo grafo simple y planar puede ser coloreado con tan solo 4 colores.*

Este teorema es difícil de demostrar. Para los propósitos del curso nos contentaremos en obtener un resultado más humilde, pero casi óptimo:

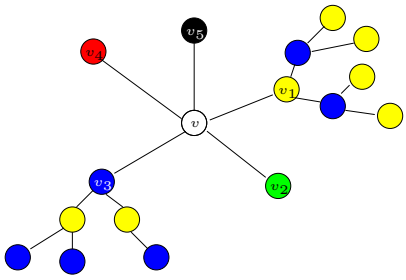
Proposición 4.6.1. *Todo grafo simple y planar puede ser coloreado con tan solo 5 colores.*

Comenzaremos por construir una demostración sencilla de que todo grafo simple y planar puede ser coloreado con tan solo 6 colores. Esta demostración servirá de base a la demostración de la Proposición 4.6.1. Utilizamos una demostración por inducción en el número n de vértices del grafo $G = (V, E)$, con casos base $n = 1, 2$. Tales casos bases cumplen la condición trivialmente. Consideremos ahora el caso inductivo $n+1$, para $n \geq 1$. Podemos asumir sin pérdida de generalidad que G es conexo. Luego, por Corolario 4.5.2, existe un vértice v en G de grado a lo más 5. Si sacamos tal vértice de G obtenemos un grafo G' con $(n-1)$ nodos. Como tal grafo es planar, obtenemos por HI que puede ser coloreado con a lo más 6 colores. Ahora, como G se obtiene de G' al agregar v y los arcos que lo unen a sus 5 vecinos, podemos colorear a G con 6 colores. Para esto, simplemente pinte v de un color que no haya sido utilizado para pintar a ninguno de sus vecinos.

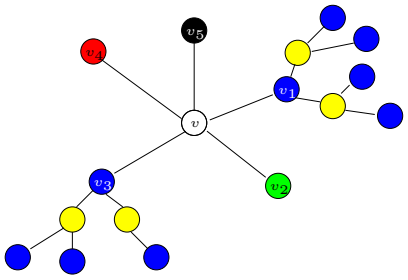
Utilizando una idea similar demostramos ahora la Proposición 4.6.1. Análogamente, en en el caso inductivo consideramos una 5-coloración obtenida por HI del grafo G' obtenido al eliminar de G al nodo v de grado menor o igual a 5. El peor caso corresponde, por supuesto, a cuando v tiene 5 vecinos y todos ellos reciben colores distintos (ya que de otra forma habría color disponible para v). Tal coloración puede observarse a continuación, asumiendo que topológicamente los vecinos de v aparecen de acuerdo a la siguiente disposición en G:



Consideremos entonces qué sucede con el subgrafo G^* de G inducido por los nodos que reciben color azul o amarillo. Suponga primero que en tal grafo no existe un camino que lleve de v_1 a v_3, como se observa a continuación:

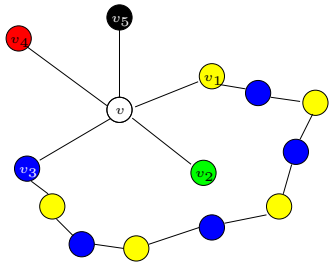


Luego, en la componente conexa de G^* a la que pertenece v_1 podemos intercambiar los colores azul y amarillo, sin afectar a la componente conexa a la que pertenece v_3. Es decir, obtenemos una 5-coloración de G' de la siguiente forma:

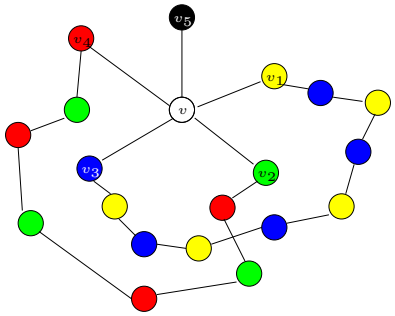


En esta coloración los vecinos de v están pintados con solo 4 colores, y por tanto queda un color disponible para v.

Supongamos entonces que existe camino de v_1 a v_3 en G^*, como se muestra a continuación:



Analizamos entonces qué pasa con el grafo inducido por lo colores rojo y verde. Si en tal grafo no existe camino de v_2 a v_4, entonces podemos liberar un color para v intercambiando colores como hicimos antes. Suponga entonces que en tal grafo los nodos v_2 y v_2 están conectados como se observa:



Necesariamente entonces el camino que lleva de v_1 a v_3 debe cruzarse con el que lleva de v_2 a v_4 (como se observa en la figura). Por tanto, G no es planar, lo que es una contradicción.

Ejercicios Finales

1. Asuma que $G = (V, E)$ es un grafo conexo, simple y planar, con al menos una región interna, tal que V tiene al menos $k \geq 3$ vértices. Asuma además que G no tiene circuitos de largo estrictamente menor a k.

   Demuestre que $|V| \geq (|E|(k-2) + 2k)/k$.

   tem Sea $G = (V, E)$ un grafo simple con al menos 11 vértices. Demuestre que G o su complemento $\bar{G}$ no es planar. (Recuerde que $\bar{G} = (V, \bar{E})$, donde $\bar{E} = (V \times V) - E$).

2. Un conjunto de vértices de un grafo es *independiente* si ningún par de vértices en el conjunto son adyacentes en el grafo. Demuestre que en todo grafo simple $G = (V, E)$ se tiene que $|V| \leq \chi(G) \cdot \mathcal{I}(G)$, donde $\mathcal{I}(G)$ es el tamaño del mayor conjunto independiente de vértices en G.

3. Sea G un grafo simple. Un conjunto de vértices de G es *independiente* si no hay dos vértices en el conjunto que sean adyacentes. Denotamos por $I(G)$ el máximo número de vértices en un conjunto independiente de vértices de G

   Demuestre que para todo grafo de simple $G = (V, E)$ se necesitan a lo más $|V| - I(G) + 1$ colores para colorear G.

4. Sean $G = (V, E)$ y $G' = (V', E')$ dos grafos simples. Denotamos por $G \otimes H$ al grafo simple que tiene como conjunto de vértices a $V \times V'$, y tal que para cada $u, v \in V$ y $u', v' \in V'$ se tiene que existe un arco en $G \otimes H$ entre los vértices (u, u') y (v, v') si y solo si (1) $u = v$ y $(u', v') \in E'$ o (2) $u' = v'$ y $(u, v) \in E$.

   Demuestre que para culaquier grafo simple G y H se cumple que:

$$\chi(G \otimes H) \;=\; \max\{\chi(G), \chi(H)\}.$$

5. Sea G un grafo simple. Una *coloración de los arcos* de G es una función que a cada arco de G le asigna un color de tal forma que no existan dos arcos incidentes al mismo nodo que reciban el mismo color. Denotamos por $\chi'(G)$ el menor número de colores que se necesita para colorear los arcos de G.

   Denotamos por $\delta(G)$ el máximo grado de un nodo en un grafo simple G. Demuestre que si G es un grafo simple bipartito entonces $\chi'(G) = \delta(G)$.

6. Sea G un grafo simple y conexo. Decimos que G es *k-crítico*, para $k \geq 1$, si (a) G es coloreable con k pero no con $k - 1$ colores, y (b) al sacar cualquier arco de G se obtiene un grafo que se puede colorear con $k - 1$ colores.

   Demuestre que si G es k-crítico, entonces cada uno de sus vértices tiene grado al menos $k - 1$.

4.7. Árboles

4.7.1. Caracterización y resultados básicos

En esta sección estudiamos la siguiente importante clase de grafos:

Definición 4.7.1 (Árboles)**.** *Un* árbol *es un grafo simple $G = (V, E)$ que es conexo pero no tiene ciclos (circuitos simples). Si sacamos la restricción de que G sea conexo obtenemos un* bosque.

Los árboles tienen muchas aplicaciones en computación, las que incluyen búsqueda de patrones en texto, códigos de compresión, búsqueda en grafos, representación de estrategias en juegos, representación de información, entre muchas otras.

Como se describe en la siguiente proposición, existen varias formas equivalentes de definir un árbol. Puede ser útil utilizar una o la otra según el resultado que queramos demostrar:

Proposición 4.7.1. *Sea $G = (V, E)$ un grafo simple. Los siguientes enunciados son equivalentes:*

1. *G es un árbol.*
2. *Existe un único camino simple entre cualquier par de vértices en V.*
3. *G es conexo pero sacar cualquiera de sus arcos lo desconecta.*
4. *G es conexo y tiene exactamente $|V| - 1$ arcos.*

Demostración. Primero demostramos que (1) implica (2). Asuma por contradicción que existen dos caminos simples distintos entre los vértices u y v en G. Siga tales caminos desde u hasta v. Debe haber un primer vértice w (que puede ser u mismo) en que tales caminos se separan. Luego, ambos vuelven a unirse por

primera vez en un punto w' (que puede ser v mismo). Siga entonces un camino desde w a w' y luego el otro desde w' a w. Claramente, este un circuito simple en el grafo, lo que es una contradicción.

Ahora demostramos que (2) implica (3). Dado que cada par de vértices está unido por un camino simple, el grafo G es conexo. Asuma por contradicción que podemos sacar un arco (u, v) de E sin perder conectividad. Sea G' el grafo obtenido desde G al sacar el arco (u, v). Entonces existe π camino de u a v en G'. Por tanto, en G existen dos caminos simples distintos desde u a v: el que avanza siguiendo el arco (u, v) y el dado por π. Esto es una contradicción.

Para demostrar que (3) implica (4) saque un arco cualquiera de G. Por (3) esto hace que G pase a tener dos componentes conexas $G_1 = (V_1, E_2)$ y $G_2 = (V_2, E_2)$ no vacías. Por HI tenemos que G_i tiene $|V_i| - 1$ arcos, para $i = 1, 2$. Luego G tiene $|V_1| + |V_2| - 1 = |V| - 1$ arcos.

Finalmente demostramos que (4) implica (1). Suponga por contradicción que G no es árbol. Dado que G es conexo, esto solo puede deberse a que existe circuito simple en G. Iterativamente saque un arco de cada ciclo en G hasta obtener un grafo G' que no tenga ciclos. Note que esta operación preserva la conectividad del grafo por lo que G' es conexo. Concluímos que G' es un árbol. Además, G tiene n vértices pero $t < n - 1$ arcos. Esto es una contradicción. □

4.7.2. Árboles cobertores

Es posible conectar todos los vértices de un grafo conexo G de forma de evitar los ciclos. Esto es importante, por ejemplo, para repartir paquetes de datos en una red de computadores: Queremos que todos los servidores reciban tales paquetes, pero al mismo tiempo necesitamos evitar que el mismo servidor reciba más de una vez el mismo paquete. Formalizamos esta idea a continuación.

Sea $G = (V, E)$ un grafo simple que es conexo. Un *árbol cobertor* de G (o *spanning tree* en inglés) es un subgrafo G' de G que corresponde a un árbol y contiene a cada vértice en V. Podemos demostrar entonces que todo grafo simple que es conexo contiene un árbol cobertor:

Proposición 4.7.2. *Sea $G = (V, E)$ un grafo simple. Entonces:*

$$G \text{ es conexo} \iff G \text{ tiene un árbol cobertor.}$$

Ejercicio. *Demuestre la Proposición 4.7.2.*

Pregunta. *¿Se le ocurre una forma eficiente de construir un árbol cobertor de un grafo conexo?*

En el próximo ejemplo se demuestra una propiedad importante de los árboles cobertores de un grafo conexo:

Ejemplo 4.7.1. Sea $G = (V, E)$ un grafo simple y conexo. Asuma que T_1 y T_2 son dos árboles cobertores distintos cualesquiera de G. Demostraremos que para todo arco e_1 en $T_1 \setminus T_2$ existe arco e_2 en $T_2 \setminus T_1$ tal que ambos:

$$\big((T_1 \cup \{e_2\}) \setminus \{e_1\}\big) \qquad \text{y} \qquad \big((T_2 \cup \{e_1\}) \setminus \{e_2\}\big)$$

son también árboles cobertores de G.

Sea e_1 un arco cualquiera en T_1 pero no en T_2. Asuma que e_1 conecta al nodo u con el nodo v. Al sacar e_1 de T_1 obtenemos dos componentes conexas, digamos U y V (asumiendo que $u \in U$ y $v \in V$). Dado que T_2 es conexo, existe camino π en T_2 desde u hasta v. Por tanto, este camino debe contener un arco (u', v') tal que $u' \in U$ y $v' \in V$. Tal arco no está en T_1 por definición (ya que de otra forma U y V estarían conectados al sacar e_1 desde T_1).

Sea e_2 el arco (u', v'). Primero demostraremos que $(T_1 \cup \{e_2\}) \setminus \{e_1\}$ es un árbol cobertor de G. Claramente, este grafo es conexo y pasa por cada vértice de G. Además, $(T_1 \cup \{e_2\}) \setminus \{e_1\}$ tiene $|V| - 1$ arcos, por lo que no puede tener ciclos (ya que de otra forma podríamos iterativamente ir sacando arcos de los ciclos hasta obtener un árbol cobertor con menos de $|V| - 1$ arcos, lo que es una contradicción).

Ahora demostramos que $(T_2 \cup \{e_1\}) \setminus \{e_2\}$ es un árbol cobertor de G. Es claro que $(T_2 \cup \{e_1\}) \setminus \{e_2\}$ es conexo y pasa por cada vértice de G. En efecto, suponga que existe camino en T_2 desde vértice w a w' que

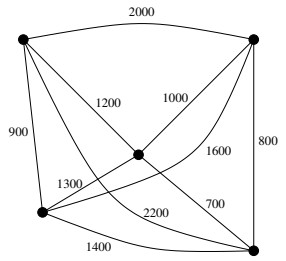


Figura 4.12: El grafo $G = (V, E)$ con pesos utilizado en el Ejemplo 4.7.2.

pasa por arco $e_2 = (u', v')$. Entonces existe camino de w a w' en $(T_2 \cup \{e_1\}) \setminus \{e_2\}$ que en vez de pasar por arco (u', v') sigue camino de u' a u, luego arco $e_1 = (u, v)$, y luego camino de v a v'. Además, $(T_2 \cup \{e_1\}) \setminus \{e_2\}$ tiene $|V| - 1$ arcos, por lo que no tiene ciclos. □

4.7.3. Árboles cobertores mínimos

Suponga ahora que los grafos que consideramos tienen sus aristas etiquetadas por pesos (como en el caso del Algoritmo de Dijkstra que presentamos en la Sección 4.4). Queremos, entonces, no solo encontrar un árbol cobertor sino uno que sea mínimo en términos del peso de los arcos que participan en él. Aún cuando existe un número exponencial de árboles cobertores de un grafo, no necesitamos considerarlos todos ellos para resolver este problema. De hecho, demostraremos a continuación que existe un simple algoritmo *avaro* que resuelve este problema eficientemente.

Formalmente, nuestra entrada consiste de un grafo simple y conexo $G = (V, E)$ y una función $w : E \to \mathbb{N}$ que asocia un peso con cada arco de G. Un *árbol cobertor mínimo* de (G, w) es un árbol cobertor de G que tiene la menor posible suma de pesos de sus arcos.

A continuación presentamos un algoritmo, conocido como *Algoritmo de Prim*, que dado un grafo simple y conexo $G = (V, E)$ y una función $w : E \to \mathbb{N}$, encuentra un árbol cobertor mínimo de (G, w). El algoritmo opera como sigue asumiendo que G tiene n vértices:

1. Sea e_1 un arco cualquiera de peso mínimo en G. Defina $T_1 = \{e_1\}$.
2. Por cada $1 \leq i \leq n - 2$, elija un arco e cualquiera fuera de T_i que sea de peso mínimo entre aquellos que tocan a T_i y tal que $T_i \cup \{e_{i+1}\}$ sea un árbol. Defina:
$$T_{i+1} := T_i \cup \{e\}.$$
3. Entregue como resultado a T_n.

Antes de demostrar la correctitud del algoritmo (es decir, que T_n es efectivamente un un árbol cobertor mínimo de (G, w)) es útil ilustrar cómo opera el algoritmo por medio de un ejemplo:

Ejemplo 4.7.2. Considere el grafo $G = (V, E)$ con pesos que se muestra en la Figura 4.12. El algoritmo de Prim construye una secuencia de árboles $T_1, \ldots, T_4$ tal que:

1. T_1 consiste solo del arco cuyo peso es 700 (ya que este es de peso mínimo entre todos los arcos de G).
2. T_2 extiende T_1 con el arco cuyo peso es 800 (ya que este es de peso mínimo entre los arcos que tocan los nodos en T_1).
3. T_3 extiende T_2 con el arco cuyo peso es 1200 (ya que este es de peso mínimo entre los arcos que tocan los nodos en T_2 y no generan ciclos).

4. T_4 extiende T_3 con el arco cuyo peso es 900 (ya que este es de peso mínimo entre los arcos que tocan los nodos en T_3 y no generan ciclos).

Note que que T_4 tiene costo $700+800+1200+900 = 3600$. Es fácil observar que T_4 es árbol cobertor mínimo de G. □

Ahora demostramos la correctitud del algoritmo:

Proposición 4.7.3. *Sea $G = (V, E)$ un grafo simple y conexo, y $w : E \to \mathbb{N}$ una función que asigna pesos a los arcos de G. Si en la entrada dada por (G, w) el algoritmo de Prim entrega como resultado a T_n, entonces T_n es un árbol cobertor mínimo de (G, w).*

Demostración. Note que T_n es conexo y tiene $n-1$ arcos. Por lo tanto, es un árbol. Sean $e_1, \ldots, e_{n-1}$ los arcos elegidos por el algoritmo. El primer arco e_1 agrega a T_1 dos vértices en G, y con cada arco e_i, para $2 \leq i \leq n-2$, agregamos un nuevo vértice a T_i. Concluimos que T_n tiene n vértices, y por tanto es un árbol cobertor de G. A continuación demostramos que también es mínimo.

Asuma que k es el mayor entero tal que $s_k = \{e_1, \ldots, e_k\}$ está contenido en algún árbol cobertor mínimo de G. Suponga por contradicción que $k < n-1$. Sea T el árbol cobertor mínimo que contiene a s_k. Luego, $T \cup \{e_{k+1}\}$ tiene un circuito simple que pasa por e_{k+1}. Este ciclo debe contener un arco que no está en $s_k \cup \{e_{k+1}\}$, ya que sabemos que este conjunto es un árbol.

Por construcción, sabemos que existe arco en s_k que comparte un vértice v con e_{k+1}. Siga el ciclo antes mencionado desde v hasta encontrar el primer arco e que no pertenece a $s_k \cup \{e_{k+1}\}$ pero comparte un vértice con s_k. Note que, por definición, $e \in T$. Basándonos en las ideas del Ejemplo 4.7.1 podemos demostrar que:

$$\big((T \cup \{e_{k+1}\}) \setminus \{e\}\big)$$

es también un árbol cobertor de G, Además, ambos e como e_{k+1} comparten vértices con s_k. Por construcción, entonces:

$$w(e_{k+1}) \leq w(e).$$

Luego, el costo de $(T \cup \{e_{k+1}\}) \setminus \{e\}$ es igual o menor al de T, por lo que $(T \cup \{e_{k+1}\}) \setminus \{e\}$ es también un árbol cobertor mínimo de G. Esto quiere decir que existe un árbol cobertor mínimo de G que contiene a $\{e_1, \ldots, e_{k+1}\}$, lo que es una contradicción. □

Finalmente, note que el algoritmo es eficiente: de hecho, si G tiene m arcos entonces el algoritmo debe realizar a lo más m pasos, y cada uno de tales pasos debe elegir un arco de peso mínimo entre m posibilidades. Por tanto, el algoritmo puede llevarse a cabo en tiempo $O(m^2)$.

Ejercicios Finales

1. Un spanning tree *dirigido* de un grafo dirigido simple $G = (V, E)$ es un árbol dirigido T (con raíz) que consiste de algunos de los arcos en G, y tal que cada vértice de G (salvo la raíz) es el punto final de algún arco en T.

   Demuestre que si G es un grafo dirigido simple y conexo en que cada vértice tiene el mismo grado de entrada que de salida, entonces G tiene un spanning tree dirigido.

2. Sea $G = (V, E)$ un grafo simple y conexo, y asuma que $c : E \to \mathbb{N}$ una función que asigna a cada arco $e \in E$ un costo $c(e) \geq 0$. El *costo* $c(T)$ de un spanning tree T de G es la suma de los costos de los arcos de G que pertenecen a T. Un spanning tree T de G tiene *costo mínimo* si no existe spanning tree T' de G tal que $c(T') < c(T)$.

   Demuestre que si la función c es inyectiva entonces G tiene un único spanning tree de costo mínimo.

3. Sea T un *spanning tree* (árbol cobertor) de un grafo simple G construído mediante búsqueda en anchura, y asuma que la raíz de T es el nodo u. Demuestre que el largo del camino más corto en G entre u y un nodo cualquiera v corresponde a la distancia entre u y v en T.

4. Sea G un grafo simple *dirigido*. Una búsqueda en profundidad sobre G se define igual que en el caso no dirigido, salvo que ahora podemos agregar un arco solo en caso de que este apunte desde el nodo que estamos visitando a uno que aún no ha sido agregado. En caso de que ningún arco pueda agregarse, elegimos un nodo no visitado al azar y seguimos la búsqueda desde él creando un nuevo árbol.

   Explique cómo una búsqueda en profundidad sobre G puede ser utilizada para determinar si G tiene un ciclo dirigido.

5. Un *grafo etiquetado* es un par (G, l), donde $G = (V, E)$ es un grafo simple y $l : V \to \{0, 1\}$ es una función que etiqueta con valor 0 o 1 a cada vértice de G. Un árbol etiquetado es un grafo etiquetado que satisface que G es un árbol. En tal caso denotamos a G por T.

   Considere dos grafos etiquetados (G_1, l_1) y (G_2, l_2), donde $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$. Un *homomorfismo* de (G_1, l_1) a (G_2, l_2) es una función $h : V_1 \to V_2$ tal que:

   a) Para cada $v \in V_1$ se cumple que $l(v) = 0$ si y solo si $l(h(v)) = 0$.

   b) Para cada $u, v \in V_1$ se cumple que si $(u, v) \in E_1$ entonces $(h(u), h(v)) \in E_2$.

   Construya un procedimiento que dados un grafo etiquetado (G, l) y un árbol eqtiquetado (T, l') determine si existe homomorfismo de (T, l') a (G, l). Su procedimiento debe realizar a lo más $O((nm)^c)$ operaciones, donde n es el número de vértices en G, m es el número de vértices en T, y $c \geq 1$ es la constante que usted desee. (Es importante notar que tal procedimiento existe debido a que T es un árbol. En particular, su procedimiento no puede funcionar cuando T es un grafo cualquiera).

   Hint: Utilice un procedimiento recursivo desde las hojas del árbol T hasta su raíz. Este procedimiento va asignando a cada vértice t de T un subconjunto $S(t)$ de los vértices de G de tal forma que se cumple lo siguiente: Sea H el conjunto de todos los homomorfismos que existen desde el subárbol de T cuya raíz es t al grafo G. Entonces $S(t) = \{h(t) \mid h \in H\}$.

6. Suponga que $G = (V, E)$ es un grafo simple y conexo, y $w : E \to \mathbb{N}$ es una función que asigna pesos a los arcos de G. Asuma que $e \in E$ es un arco que toca un vértice $v \in V$, y que el peso de e no supera el peso de ningún otro arco que toca a v. Demuestre que existe un árbol cobertor mínimo de G que contiene a e.

Capítulo 5

Teoría de Números

La teoría de números se conoce como "la reina de las matemáticas". Esto se debe a qué los problemas que aparecen relacionados con ellas son, en algunos casos, muy fáciles de explicar pero muy difíciles de resolver. Muchos de esos problemas aún están abiertos y corresponden a ciertas conjeturas famosas (veremos algunas en este capítulo). Tales conjeturas han fascinado a varios de los más grandes matemáticos a lo largo de los siglos. La teoría de números tiene, además, importantes aplicaciones en la ciencia de la computación; por ejemplo, como veremos más adelante, las técnicas del área pueden ser utilizadas para diseñar algoritmos eficientes para el problema del máximo común divisor y para desarrollar códigos criptográficos de llave pública computacionalmente seguros.

5.1. Resultados básicos

El teorema fundamental de la teoría de números establece que todo natural $n \geq 2$ admite una única factorización prima:

Teorema 5.1.1 (Teorema Fundamental)**.** *Sea $n \geq 2$ un número natural. Entonces n tiene una única factorización en números primos.*

Demostración. Como vimos en el Ejemplo 2.1.2, todo número natural $n \geq 2$ puede ser factorizado como un producto de primos. Debemos demostrar, por tanto, que tal factorización prima es única. Asuma, por contradicción, que existen números que admiten dos factorizaciones primas distintas. Por el Principio del Buen Orden debe existir un menor tal número; digamos $n \geq 2$. Considere entonces dos factorizaciones primas distintas de n dadas por:

$$n = p_1 p_2 \dots p_k \qquad \text{y} \qquad n = q_1 q_2 \dots q_m,$$

donde $p_1 \leq p_2 \leq \cdots \leq p_k$ y $q_1 \leq q_2 \leq \cdots \leq q_m$. Asumamos, sin pérdida de generalidad, que $p_1 \leq q_1$.

Es posible demostrar entonces que p_1 no divide a ningún q_j, donde $1 \leq j \leq m$. En efecto, dado que los q_j's son primos y están ordenados en orden no decreciente, esto solo podría pasar si $p_1 = q_1$. Sea:

$$n' = \frac{n}{p_1} = \frac{n}{q_1}.$$

Claramente, $n' < n$. Además:

$$n' = p_2 \dots p_k \qquad \text{y} \qquad n' = q_2 \dots q_m.$$

Estas factorizaciones primas deben ser distintas (ya que $p_1 \dots p_k$ y $q_1 \dots q_m$ corresponden a factorizaciones primas distintas de n y $p_1 = q_1$). Concluimos que n no es el menor natural que admite dos factorizaciones primas distintas, lo que es una contradicción.

Sea r_j el resto que se obtiene al dividir q_j por p_1, para cada $1 \leq j \leq m$. Es decir, $q_j = a_j p_1 + r_j$, donde $0 \leq r_j < p_1$ para cada $1 \leq j \leq m$. Pero por nuestra observación anterior, $r_j \neq 0$ para cada $1 \leq j \leq m$. Definamos entonces:

$$n^* = r_1 r_2 \dots r_m.$$

Note que $0 < n^* < n$ dado que $0 < r_j < p_1 \leq p_2 \leq \dots \leq p_k$ para cada $1 \leq j \leq m$. Demostraremos a continuación que n^* admite dos factorizaciones primas distintas, lo que es una contradicción.

En efecto, dado que $r_j < p_1$ para cada $1 \leq j \leq m$, existe factorización prima de n^* que no menciona a p_1 (simplemente descomponiendo a los r_j's en factores primos). Por otro lado, podemos demostrar que n^* es divisible por p_1, lo que implica que tiene otra factorización prima que menciona a p_1. De hecho, tenemos por definición que:

$$n^* = (q_1 - a_1 p_1)(q_2 - a_2 p_1) \dots (q_m - a_m p_1).$$

Al desarrollar este producto, podemos entonces escribir a n^* como:

$$q_1 q_2 \dots q_m + p_1 s,$$

para algún entero s. Pero $q_1 \dots q_m$ también es divisible por p_1 ya que:

$$n = q_1 \dots q_m = p_1 \dots p_k.$$

Concluimos que n^* puede expresarse de la forma $p_1 t$ para algún entero t, y que por tanto n^* es divisible por p_1. Esto finaliza la demostración. □

A continuación vemos una aplicación interesante del Teorema 5.1.1:

Corolario 5.1.1. *Sea $p \geq 2$ un primo. Entonces $\sqrt{p}$ es irracional.*

Demostración. Asuma, por contradicción, que $\sqrt{p} = a/b$, donde $a, b \in \mathbb{Z}$ y $b \neq 0$. Por tanto, $a^2 p = b^2$. Esto quiere decir que la (única) factorización prima de $a^2 p$ es igual a la (única) factorización prima de b^2. Note que la factorización prima de $a^2 p$ debe contener un número impar de veces al primo p. Esto se debe a que en tal factorización p aparece $2x + 1$ veces, donde x es el número de veces que aparece en la factorización prima de a. Por una razón similar, p aparece un número par de veces en la factorización prima de b^2. Esto es una contradicción. □

Ejercicio. *Sea $p \geq 2$ un primo. Demuestre que si p divide a c pero no a d, entonces p divide a $\frac{c}{d}$.*

5.2. Infinitud de los primos

Un resultado fundamental de la teoría de números es la infinitud de los números primos. A continuación presentamos una demostración de tal resultado:

Teorema 5.2.1. *Existen infinitos números primos.*

Demostración. Asuma, por contradicción, que existe un número finito de primos solamente, y que todos los primos que existen son:

$$p_1 < p_2 < \dots < p_n.$$

Considere el número $r = p_1 p_2 \dots p_n$. Demostraremos que cualquier primo q en la factorización prima de $r+1$ debe ser estrictamente mayor que p_n, lo que es una contradicción. Asuma, por el contrario, que $q \leq p_n$. Dado que q es un primo, debe ser uno de los p_i's para $1 \leq i \leq n$. Esto quiere decir que q aparece en la factorización prima de r. Pero por definición también aparece en la factorización prima de $r+1$. Esto implica que q divide a 1, lo que es una contradicción. □

Ejercicio. *Demuestre a partir de la demostración anterior que si p_n es el n-ésimo primo, entonces $p_n \leq 2^{2^n}$.*

Ejercicio. *Demuestre que para todo $n \geq 1$ existe un intervalo de n naturales consecutivos que no son primos.*

Muchos problemas relacionados con propiedades fundamentales de los números primos permanecen abiertas. A continuación mencionamos dos de ellas:

- **Conjetura de los primos gemelos:** Existe un número infinito de pares de primos *gemelos*; es decir, de de la forma $(p, p+2)$, como por ejemplo $(5, 7)$ o $(11, 13)$.

  Actualmente se sabe que existe un número $n \leq 246$ tal que existe un número infinito de pares de primos de la forma $(p, p+n)$.

- **Conjetura de Goldbach:** Todo número natural $n \geq 2$ puede escribirse como la suma de a lo más 2 primos. Por ejemplo, $16 = 13 + 3$ y $58 = 53 + 5$.

  Sola la versión débil de la conjetura ha podido ser demostrada. Esta establece que todo número impar puede expresarse como la suma de a lo más tres primos.

5.3. Caracterización de los primos

De ahora en adelante escribimos:

$$p \equiv q \pmod{n}$$

si p y q tienen el mismo resto al ser divididos por n, o, equivalentemente, si $(p - q)$ es divisible por n.

En la Antigua China se creía que un número $n > 2$ are primo si, y solo si, $2^{n-1} \equiv 1 \pmod{n}$. Sin embargo, esta conjetura resultó ser falsa. En efecto, el número 341 no es primo, ya que $341 = 11 \cdot 31$, pero por otro lado $2^{341-1} \equiv 1 \pmod{341}$. Esto quiere decir que si $2^{n-1} \equiv 1 \pmod{n}$, para $n > 2$, entonces no necesariamente n es primo. Interesantemente, la otra dirección de la conjetura de los chinos sí es cierta; es decir, si $n > 2$ es primo entonces $2^{n-1} \equiv 1 \pmod{n}$. No solo eso, sino que además $a^{n-1} \equiv 1 \pmod{n}$ para cada a que no es múltiplo de n. Este importante resultado se conoce como *Pequeño Teorema de Fermat*:

Teorema 5.3.1. *Sea $n > 2$ un número primo. Entonces:*

$$a^{n-1} \equiv 1 \pmod{n},$$

para todo a que no es múltiplo de n.

Demostración. La demostración consta de dos partes:

1. Primero demostramos que si $n > 2$ es primo, entonces para todo $a \geq 1$:

$$a^n \equiv a \pmod{n}.$$

2. Luego demostramos que si a no es múltiplo de n, entonces:

$$a^{n-1} \equiv 1 \pmod{n} \iff a^n \equiv a \pmod{n}.$$

Comencemos con la primera parte, la que demostramos por inducción en $a \geq 1$. El caso base $a = 1$ es trivial ya que $1^n \equiv 1 \pmod{n}$. Consideremos entonces el caso inductivo $a + 1$, para $a \geq 1$. Por HI podemos suponer entonces que $a^n \equiv a \pmod{n}$; es decir, que $a^n - a$ es divisible por n. Queremos demostrar entonces que $(a+1)^n - (a+1)$ es también divisible por n.

Sabemos que:

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

Como $a^n - a$ es divisible por n por HI, basta demostrar que $\sum_{i=1}^{n-1} \binom{n}{i} a^i$ es divisible por n. Pero para esto basta demostrar que $\binom{n}{i}$ es divisible por n, para cada $1 \leq i \leq n-1$. Esto se sigue del Ejercicio 5.1 ya que $\binom{n}{i} = \frac{n!}{(n-i)!\cdot i!}$ y tenemos que n divide a $n!$ pero no divide a $(n-i)!\cdot i!$. Esto último se sigue del hecho de que n es primo y todos los factores en $(n-i)!\cdot i!$ son estrictamente menores que n. Es decir, n no aparece en la factorización prima de ninguno de esos factores, y por tanto no puede aparecer en la factorización prima de $(n-i)!\cdot i!$.

Para la parte 2, note primero que si $a^{n-1} \equiv 1 \pmod{n}$, es decir, $a^{n-1} - 1$ es divisible por n, entonces $a \cdot (a^{n-1} - 1) = a^n - a$ también es divisible por n. Concluimos que $a^n \equiv a \pmod{n}$. Asuma, por el contrario, que $a^n \equiv a \pmod{n}$, es decir, $a^n - a$ es divisible por n. Pero entonces $a \cdot (a^{n-1} - 1)$ es divisible por n, lo que implica que a o $a^{n-1} - 1$ es divisible por n (ya que n es primo, y por tanto debe aparecer en la factorización prima de alguna de estas dos expresiones). Pero n no divide a a por hipótesis, por lo que debe dividir a $a^{n-1} - 1$. Es decir, $a^{n-1} \equiv 1 \pmod{n}$. □

Pregunta. *¿Sigue siendo cierto el Teorema si eliminamos la condición de que a no sea múltiplo de n?*

Con la ayuda del Pequeño Teorema de Fermat podemos demostrar la siguiente caracterización de cuando un número es primo:

Proposición 5.3.1. *Un número $n > 2$ es primo si, y solo si, $a^{n-1} \equiv 1 \pmod{n}$ para todo $1 \leq a \leq n-1$.*

Demostración. De izquierda a derecha podemos usar el Teorema 5.3.1 ya que a no es múltiplo de n cuando $1 \leq a \leq n-1$. Para la dirección de derecha a izquierda suponga que n no es primo. Es decir existe $1 < a \leq n-1$ que divide a n. Entonces $a^{n-1} \not\equiv 1 \pmod{n}$. En efecto, asuma por contradicción que $a^{n-1} \equiv 1 \pmod{n}$. Entonces $a^{n-1} \equiv 1 \pmod{a}$, ya que a divide a n. Pero a divide a a^{n-1}, por lo que $a^{n-1} \equiv 0 \pmod{a}$. Esto es una contradicción. □

Este resultado da paso a lo que se conoce como *Test de Fermat* para verificar si un número $n > 2$ es primo. La idea es la siguiente: Se elige un número al azar $a < n$ y se verifica si $a^{n-1} \equiv 1 \pmod{n}$. Esto puede llevarse a cabo eficientemente utilizando técnicas de aritmética modular. Si $a^{n-1} \not\equiv 1 \pmod{n}$ podemos estar seguros que n no es primo. En caso contrario elegimos otro $b < n$ al azar y repetimos la prueba. El problema de este test es que para asegurarnos de que n no es primo debemos considerar demasiados números, lo que hace que el procedimiento no sea eficiente.

Afortunadamente, existe un algortimo *probabilista* muy eficiente que ayuda a determinar si un número $n > 2$ es primo. Este se conoce como *Test de Miller-Rabin*. Al igual que el test de Fermat, este procedimiento puede detectar al verificar una condición para un número $a < n$ elegido al azar si n no es primo. En caso de que esta condición no se satisfaga, el test aún no puede estar seguro de que sea n sea primo. Sin embargo, en tal caso la probabilidad de que n no sea primo es menor a $1/2$. Entonces, si el test se sigue iterando n veces y los resultados siguen siendo consistentes con la afirmación de que que n es primo, la probabilidad de que n no lo sea es menor a $(1/2)^n$. Para n suficientemente grande esta probabilidad se vuelve tan pequeña que podemos estar casi seguros de que n es primo.

Se conoce además un algoritmo eficiente que verifica si un número es primo. Este algoritmo es tan complejo, sin embargo, que en la práctica se sigue utilizando el Test de Miller-Rabin debido a su simplicidad (a pesar de la posibilidad de error subyacente). Por otro lado, se cree que no hay algoritmo eficiente que resuelva problema de encontrar la factorización prima de un número natural. Ambas observaciones son particularmente importantes al momento de diseñar los protocolos criptográficos basados en teoría de números que veremos al finalizar este capítulo.

5.4. Algoritmo euclideano

La clave para diseñar algoritmos más avanzados basados en teoría de números es el *algoritmo euclideano*, el que permite determinar de forma eficiente el mayor común divisor $\mathrm{mcd}(a, b)$ de los naturales $a, b \geq 1$. Este algoritmo está basado en la siguiente simple propiedad:

Proposición 5.4.1. *Suponga que $a > b$ y $r \neq 0$ es el resto que se obtiene al dividir a por b. Entonces:*

$$\mathrm{mcd}(a, b) \;=\; \mathrm{mcd}(b, r).$$

Demostración. Basta demostrar que para todo $c \geq 1$ se tiene que c divide a ambos a y b si, y solo si, divide a ambos b y r. Suponga primero que c divide a ambos a y b. Demostraremos que también divide a r. Pero esto se sigue directamente del hecho de que por definición $r = a - bk$, para algún $k \geq 1$ (ya que r es el resto que se obtiene al dividir a a por b). Al contrario, asuma que c divide a ambos b y r. Pero entonces también divide a a ya que $a = r + bk$. □

Desde este resultado directamente obtenemos el algoritmo euclideano, el que se explica mejor con un ejemplo:

Ejemplo 5.4.1. Suponga que queremos determinar $\mathrm{mcd}(55, 34)$. Basados en la Proposición 5.4.1 sabemos que:

$$\begin{aligned}
\mathrm{mcd}(55, 34) &= \mathrm{mcd}(34, 21) \\
&= \mathrm{mcd}(21, 13) \\
&= \mathrm{mcd}(13, 8) \\
&= \mathrm{mcd}(8, 5) \\
&= \mathrm{mcd}(5, 3) \\
&= \mathrm{mcd}(3, 2) \\
&= \mathrm{mcd}(2, 1)
\end{aligned}$$

El algoritmo se detiene en este caso ya que el resto de dividir a 2 por 1 es 0. Podemos concluir entonces que $\mathrm{mcd}(55, 34) = \mathrm{mcd}(2, 1) = 1$. Decimos entonces que 55 y 34 son *primos relativos*, ya que su único factor común es 1. □

El siguiente resultado establece la eficiencia del algoritmo en términos del número de pasos que debe desarrollar:

Proposición 5.4.2. *El algoritmo euclideano realiza a lo más* $(\log_2 a + \log_2 b)$ *pasos antes de detenerse.*

Demostración. Note que si r es el resto que se obtiene de dividir a a por b, asumiendo que $a > b$, entonces $r < a/2$ (ya que de otra forma b cabría una vez más en a). Luego $br < ab/2$. Después de k pasos tenemos entonces que los números c y d considerados por el algoritmo son estrictamente menores que $(ab/2^k)$. El algoritmo se detiene un paso antes de que cd sea igual a 0. Esto ocurre cuando $(ab/2^k) < 1$, es decir, apenas $k > \log_2 ab = \log_2 a + \log_2 b$. □

Una importante consecuencia del algoritmo euclideano es que el mayor común divisor de dos números a y b siempre puede expresarse como una combinación lineal de a y b:

Corolario 5.4.1. *Para todo $a, b \geq 1$ existen $s, t \in \mathbb{Z}$ tal que:*

$$sa + tb \;=\; \mathrm{mcd}(a, b).$$

Demostración. Suponga que queremos determinar $\mathrm{mcd}(a_0, a_1)$, y que para esto el algoritmo realiza los siguientes pasos:

$$\begin{aligned}
\mathrm{mcd}(a_0, a_1) &= \mathrm{mcd}(a_1, a_2) \\
&= \ldots \\
&= \mathrm{mcd}(a_{m-1}, a_m).
\end{aligned}$$

En particular, a_{i+1} corresponde al resto de dividir a_{i-1} por a_i, para cada $1 \leq i < m$, y a_m divide a a_{m-1}. Es decir, $a_m = \text{mcd}(a_0, a_1)$. Demostraremos por inducción que para todo $0 \leq i \leq m$ existen $s, t \in \mathbb{Z}$ tal que $a_i = sa_0 + ta_1$.

Para a_0 basta elegir $s = 1$ y $t = 0$, y para a_1 lo contrario. Asuma por HI que $a_{i-1} = sa_0 + ta_1$ y $a_i = s'a_0 + t'a_1$, para $1 \leq i < m$. Dado que a_{i+1} es el resto que se obtiene al dividir a_{i-1} por a_i, tenemos que existe $k \geq 1$ tal que:

$$a_{i+1} \;=\; a_{i-1} - ka_i \;=\; sa_0 + ta_1 - k(s'a_0 + t'a_1).$$

Es decir, $a_{i+1} = (s - ks')a_0 + (t - kt')a_1$. Esto demuestra el resultado. □

A partir del Corolario 5.4.1 podemos demostrar la existencia de "inversos" de a modulo m, en la medida que a y m sean primos relativos.

Corolario 5.4.2. *Asuma que* $\text{mcd}(a, m) = 1$. *Entonces existe* a^{-1} *tal que* $aa^{-1} \equiv 1 \,(\text{mod } m)$.

Demostración. Sabemos que existen $s, t \in \mathbb{Z}$ tal que $sa + tm = \text{mcd}(a, m) = 1$. Pero entonces $sa - 1$ es divisible por m, es decir, $sa \equiv as \equiv 1 \,(\text{mod } m)$. Podemos entonces elegir a a^{-1} como s. □

Pregunta. *¿Qué pasa con el resultado anterior si no se cumple* $\text{mcd}(a, m) = 1$*?*

Ejemplo 5.4.2. Asuma que $\text{mcd}(a, m) = 1$. Entonces existe solución a la ecuación:

$$ax \;\equiv\; b \;(\text{mod } m).$$

En efecto, a partir del Corolario 5.4.2 sabemos que existe inverso a^{-1} de a modulo m; es decir, $aa^{-1} \equiv 1 \,(\text{mod } m)$. Luego, $a^{-1}b$ es solución de la ecuación. De hecho, $a(a^{-1}b) \equiv (aa^{-1})b \equiv b \,(\text{mod } m)$. Esto último se cumple pues el valor de yz modulo m corresponde al valor de z modulo m cuando $y \equiv 1 \,(\text{mod } m)$. □

A partir de esto podemos demostrar el siguiente famoso resultado:

Teorema 5.4.1 (Resto Chino)**.** *Asuma que* $m_1, \ldots, m_p$ *son enteros tal que* $\text{mcd}(m_i, m_j) = 1$ *para todo* $1 \leq i < j \leq p$. *Entonces el sistema de ecuaciones:*

$$\{x \;\equiv\; a_i \;(\text{mod } m_i) \;\mid\; 1 \leq i \leq p\}$$

tiene solución.

Demostración. Sea $M = m_1 \ldots m_p$ y $M_i = M/m_i$. Note que $\text{mcd}(m_i, M_i) = 1$ para todo $1 \leq i \leq p$. Si no fuera así, m_i y M_i tendrían un factor primo $q \geq 2$. Tal factor primo tendría que aparecer en la factorización prima de m_i y de m_j, para $j \neq i$, lo que contradice $\text{mcd}(m_i, m_j) = 1$. Por el Corolario 5.4.2 concluimos entonces que existe y_i tal que $M_i y_i \equiv 1 \,(\text{mod } m_i)$ para todo $1 \leq i \leq p$.

Definamos:

$$t \;=\; \sum_{i=1}^{p} a_i M_i y_i.$$

Entonces t es solución al sistema de ecuaciones. En efecto, considere un $1 \leq i \leq p$ arbitrario. Luego, para todo $j \neq i$ tenemos que $a_j M_j y_j$ es divisible por m_i (ya que m_i aparece en M_j). Es decir, $a_j M_j y_j \equiv 0 \,(\text{mod } m_i)$. Basta demostrar entonces que $a_i M_i y_i \equiv a_i \,(\text{mod } m_i)$. Pero esto se sigue directamente del hecho que $M_i y_i \equiv 1 \,(\text{mod } m_i)$. □

Ejercicio. *Demuestre que la solución al sistema de ecuaciones anterior es única modulo* M.

5.5. Código criptográfico RSA

Al contrario de los códigos criptográficos tradicionales, este código es de *llave pública*; es decir, todos conocen la llave e necesaria para encriptar un mensaje, pero solo el receptor conoce la llave d necesaria para descrifrarlo. Lo interesante es que es posible obtener d desde e, pero solo asumiendo un costo computacional estratosférico.

Supongamos que A quiere enviarle códigos encriptados a B. Entonces B parte por generar dos primos "grandes" p, q, además de un e "grande" que es primo relativo con $(p-1)(q-1)$; es decir:

$$\text{mcd}(e, (p-1)(q-1)) = 1.$$

Todo esto puede hacerse rápidamente con méodos probabilistas y el algoritmo euclideano. El valor e es entonces publicado y declarado la llave pública. También se publica $n = pq$. Por otro lado, B se guarda a p y q. A partir de ellos genera la llave descifradora d, la que satisface que:

$$de \equiv 1 \pmod{(p-1)(q-1)}.$$

Sabemos que tal d existe por el Corolario 5.4.2. Además, puede ser computado eficientemente a partir de e y $(p-1)(q-1)$ utilizando el algoritmo euclideano.

Note que solo B puede computar d eficientemente ya que conoce a p y q. Es posible calcular p y q desde el valor público n, ya que $n = pq$. Sin embargo, encontrar una factorización prima de un número "grande" como n es un problema computacionalmente muy difícil.

Para enviar un mensaje M el remitente A computa el valor encriptado:

$$C \equiv M^e \pmod{n}.$$

Esto puede realizarse eficientemente utilizando aritmética modular. Al ser recibido por B este ocupa su llave descifradora d y computa el valor modulo n de C^d. A continuación demostraremos que este valor corresponde precisamente a M (asumiendo que $M < n$).

En efecto, sabemos que:

$$C^d \equiv M^{de} \equiv M^{1+k(p-1)(q-1)} \pmod{n},$$

para algún entero $k \geq 0$. Pero entonces:

$$C^d \equiv M(M^{(p-1)})^{k(q-1)} \pmod{n}.$$

Asumiendo que $M < p$ (o en su defecto que M no es múltiplo de p) tenemos por Pequeño Teorema de Fermat que $M^{p-1} \equiv 1 \pmod{p}$. Luego:

$$M(M^{(p-1)})^{k(q-1)} \equiv M \pmod{p}.$$

Equivalentemente, es posible demostrar que:

$$M(M^{(q-1)})^{k(p-1)} \equiv M \pmod{q}.$$

Pero entonces, dado que p y q son primos, tenemos que:

$$M^{1+k(p-1)(q-1)} \equiv M \pmod{n}.$$

Luego, $C^d = M$ ya que $M < n$.