MATEMÁTICAS
EN TECNOLOGÍA Y EDUCACIÓN:
UNA PERSPECTIVA REPUBLICANA

Matemáticas en tecnología y educación : una perspectiva republicana / Darío Alejandro García ... [et al.]. -- Bogotá : Editorial Temis, 2014.
90 p. : il. ; 13 x 21 cm.
Incluye índice general.
ISBN 978-958-35-0960-5
1. Métodos de enseñanza 2. Matemáticas - Enseñanza 3. Enseñanza con ayuda de computadores 4. Tecnología educativa 5. Educación - Innovaciones tecnológicas I. García, Darío Alejandro.
371.3 cd 21 ed.
A1434305

CEP-Banco de la República-Biblioteca Luis Ángel Arango

MATEMÁTICAS EN TECNOLOGÍA Y EDUCACIÓN: UNA PERSPECTIVA REPUBLICANA

Darío Alejandro García
John Edison Castaño
Isaías Marín Gaviria
Magdalena Pradilla Rueda

Bogotá - Colombia
2013

Publicación sometida a pares académicos (*Peer Review Double Blinded*).

ISBN 978-958-5447-10-3

Diagramación y corrección: Editorial TEMIS S.A.
Calle 17, núm. 68D-46, Bogotá.
www.editorialtemis.com
correo elec. editorial@editorialtemis.com

Diseños y gráficos originales de Editorial TEMIS S.A.

Hecho el depósito que exige la ley.

PRESENTACIÓN

La Corporación Universitaria Republicana, crea el programa de matemáticas orientado al desarrollo de competencias en matemáticas y matemática aplicada, con el fin de contribuir al desarrollo de las ciencias básicas en el país, las cuales son base fundamental para el desarrollo de la ciencia, la tecnología y la innovación.

Conociendo la importancia que tienen las matemáticas en el desarrollo tecnológico y económico del país, se presenta el libro *Matemáticas en tecnología y educación: una perspectiva Republicana,* el cual es realizado por el grupo de investigación de matemáticas y ciencias de la información, ofreciendo una caracterización de diferentes temas de impacto en educación matemática y matemática aplicada.

La presente obra destaca las principales ideas matemáticas en el área de aprendizaje automático, mostrando los avances que se han obtenido en la implementación de algoritmos de aprendizaje, en la predicción del comportamiento de un usuario y el reconocimiento de imágenes. También se presentan conceptos básicos en criptografía y criptoanálisis, haciendo un recuento histórico de su desarrollo y principales aplicaciones. Además el libro lleva a cabo un análisis del Lenguaje dentro del desarrollo de la Lógica Matemática, siendo la sintaxis uno de los ejes principales para el funcionamiento del lenguaje. Conjuntamente encontramos un trabajo en pedagogía sobre el desarrollo del pensamiento matemático en la primera infancia y la importancia de implementar el método natural para no interrumpir la capacidad innata de aprendizaje de los niños.

Este libro es un pequeño compendio sobre temas de interés en diferentes campos de la matemática aplicada y de educación matemática con el cual el lector se llevará una idea de cuán importante ha sido la matemática en el desarrollo de sistemas computacionales. También se presenta la importancia y una posible propuesta metodológica para el desarrollo del pensamiento matemático en la primera infancia.

Andrea Valencia
Decana Facultad de Matemáticas y Ciencias de la Información

ÍNDICE GENERAL

Capítulo I

VC-DIMENSIÓN Y APRENDIZAJE AUTOMÁTICO

Darío Alejandro García

PÁG

Resumen 1
1. Introducción 1
2. ¿Qué es aprendizaje automático? 2
3. Algunas aplicaciones del aprendizaje automático 5
   A) Netflix: ¿cómo sugerir películas al usuario? 5
   B) El Ipad y sistemas de reconocimiento de escritura a mano 7
4. Modelos de aprendizaje: el "perceptrón" 10
5. Dimensión de Vapnik-Chervonenkis 15
   A) Algunos ejemplos 16
   B) VC-dimensión en el algoritmo perceptrón 20
   C) Dimensión finita vs. dimensión infinita 22
6. ¿Cómo podemos interpretar la VC-dimensión? 24
   A) VC-dimensión como grados de libertad 24
   B) VC-dimensión y entrenamiento necesario 25

Capítulo II

UNA INTRODUCCIÓN AL MÉTODO NATURAL DEL APRENDIZAJE DE LAS MATEMÁTICAS, EN LA PRIMERA INFANCIA. ESTADO DEL ARTE

John Edison Castaño Giraldo

Resumen 29
1. Justificación 29
2. Diseño metodológico 30
3. Antecedentes de la investigación 30
4. Método natural: enseñanza de la matemática en la primera infancia 31

PÁG

A) Asignación 31
B) Agrupación no posicional 32
C) Agrupación posicional 34
D) Agregación 35
E) Diferencia 36
F) Suma aritmética 37
G) Resta aritmética 37
5. Matemáticas para la vida 38
A) Traducir 38
B) Formular y desarrollar 39
C) Expresar 39
6. Transformación aditiva 41
7. Comparación aditiva 42
8. Algunos resultados 43

Capítulo III

ATAQUES A SISTEMAS CRIPTOGRÁFICOS

Isaías David Marín Gaviria

Resumen 47
1. Introducción 47
2. Conceptos básicos de la criptología 47
3. Sistemas criptográficos y sus ataques 49
A) Ataques a sistemas criptográficos simétricos 49
B) Ataques a sistemas criptográficos asimétricos 56
4. Conclusiones 57

Capítulo IV

BASES EPISTEMOLÓGICAS DE LOS LENGUAJES DE LAS MÁQUINAS LÓGICAS

Magdalena Pradilla Rueda

Resumen 59
1. Introducción 60
2. Representación de las máquinas lógicas 60
A) Propuestas de Turing 61
B) Planteamientos de Kleene y sus revisiones 64

PÁG

3. Analogías epistemológicas cerebro-máquinas-lenguajes 69
A) Relaciones cerebro-máquinas 69
B) Relación máquina-lenguaje 71
4. Lenguajes como formas de representación 71
A) Lenguajes formales 72
5. Bases lógicas y teóricas de los lenguajes 73
A) Lógica como lenguaje 74
B) Programa de Hilbert: teoría de la demostración 77
6. Conclusiones 78

Capítulo I

VC-DIMENSIÓN Y APRENDIZAJE AUTOMÁTICO

Darío Alejandro García*
Corporación Universitaria Republicana

Resumen

En el presente trabajo se resumen las principales ideas matemáticas en el área de aprendizaje automático, incluyendo los conceptos de VC-dimensión, redes neuronales, nociones básicas de la teoría de Vapnik-Chervonenkis, entre otros. Se incluyen también varios ejemplos de aplicaciones de sistemas de aprendizaje automático en la industria digital.

1. Introducción

La VC-densidad ha sido la medida de complejidad de conjuntos más fructífera en estadística y en teoría del aprendizaje: implica una cota uniforme (cuando se varían los conjuntos y las medidas de probabilidad) para la convergencia de los algoritmos que se pueden deducir de la ley de los grandes números. También es la base teórica del método conocido como "máquinas de vectores de soporte" (*support vector machines*), que es hasta el momento el método más exitoso en problemas de teoría de aprendizaje automático, entre los que se encuentran, por ejemplo, el problema de la identificación de escritura a mano por parte de un ordenador.

La teoría de Vapnik-Chervonenkis (VC-*theory*, como es conocida en inglés) es una teoría de aprendizaje automático y computacional desarrollada por Vladimir Vapnik y Alexei Chervonenkis durante un período de más de treinta años (1960-1993), que intenta explicar los procesos de aprendizaje mediante ensayo-error desde un punto de vista puramente estadístico. La VC-dimensión y la VC-densidad son dos invariantes nu-

* Mágister en Matemáticas de la Universidad de los Andes (Bogotá) 2009. Matemático de la Universidad Nacional de Colombia. Actualmente se encuentra cursando el último año de Doctorado en Matemáticas en la Universidad de los Andes. Tesis: *Model Theoretic proofs of combinatorial statements.*

méricas que miden el nivel de complejidad de la familia de conjuntos (por ejemplo, agrupaciones de pixeles) que se establezca desde un principio.

Los resultados obtenidos en esta teoría permitieron desarrollar lo que hoy conocemos como máquinas de vectores de soporte (*support vector machines*), máquinas de aprendizaje automático con métodos de supervisión de datos capaces de clasificar información y de predecir el comportamiento de una variable dada a lo largo de una familia de conjuntos.

A pesar de ser una teoría que data de los años sesenta, solo hasta ahora se ha visto el gran impacto que tiene ya que en la última década se desarrollaron los algoritmos eficientes de reconocimiento de lectura a mano (como el usado por el *Ipad*) y de escáner de superficies (usado en medicina para encontrar células cancerígenas mediante el sistema "Skinchecker"), o la predicción del comportamiento de un usuario (estudios de crédito, *ranking* de películas en *netflix*, etc.).

2. ¿Qué es aprendizaje automático?

El aprendizaje automático es una de las ramas de la inteligencia artificial que se encarga de la implementación de algoritmos que permitan a una máquina obtener y generalizar comportamientos a partir de una serie de datos dada.

El principio esencial del aprendizaje automático sería entonces el de "aprender a partir de datos" (*learning from data*), por lo que para poder aplicar modelos de aprendizaje automático se deben cumplir ciertos principios básicos, que podrían simplificarse de la siguiente forma:

1. Existe un patrón que puede organizar el comportamiento de los datos. Este patrón que queremos entender es justamente el objetivo de usar modelos de aprendizaje automático en una tarea dada.

2. No existe *a priori* una fórmula matemática que dé explicación a este patrón: si dicha fórmula existiese, entonces la usaríamos directamente en vez de tratar de aprender a partir de la información suministrada.

3. Existen datos sobre la tarea de la cual queremos aprender. Sin estos datos de entrenamiento, es imposible que se pueda desarrollar ningún modelo de aprendizaje automático.

Tal vez la forma más fácil de entender qué es el aprendizaje automático es por medio de un ejemplo:

Supongamos que un banco muy importante necesita un programa que permita decidir cuándo otorgar un crédito o no otorgarlo a una persona

que lo esté requiriendo. En principio, no existe ninguna fórmula matemática que permita decidir cuándo una persona es confiable o no lo es al momento de otorgar un crédito. Pero creemos que existe un patrón y una relación entre la información de la persona (salario, historial financiero, edad, etc.) y su buen o mal comportamiento como deudor.

La información con la que el banco cuenta es, básicamente, la de todos sus clientes activos, junto con la información que dice si ellos hicieron que el banco ganara dinero (a estos los llamaríamos buenos[1] clientes) o si, por el contrario, el banco perdió dinero al otorgarles el préstamo (malos clientes).

Podemos, entonces, describir la información de cada uno de los clientes como un vector de factores, un vector cuyas entradas serán números reales que codificará la información de cada persona: el perfil del cliente. También conocemos de antemano cuál fue el comportamiento de estos clientes, que podemos describir con los números 1, si ha sido un buen cliente, o -1 si no lo ha sido. Los datos descritos en este párrafo serán los datos de entrada, a partir de los cuales la máquina tratará de encontrar una fórmula matemática que describa el comportamiento de los mismos.

Pero ¿de dónde sale esta fórmula matemática? Es bien sabido que existen muchos tipos de funciones de varias variables (lineales, cuadráticas, polinómicas, exponenciales, trigonométricas, etc.) y requerir que la máquina que escoja arbitrariamente la que mejor se acomode será una tarea imposible. Por este motivo, se debe seleccionar un subconjunto del conjunto total de posibilidades, que será llamado como el *conjunto de hipótesis*, que cambiará de acuerdo con el modelo particular de aprendizaje que estemos usando.

Al finalizar el aprendizaje, el computador habrá encontrado una función que, dado el perfil de un cliente potencial, podrá decidir cuándo es beneficioso para el banco otorgarle un crédito.

Podemos formalizar este modelo de la siguiente manera:

Componentes del aprendizaje:

• *Datos de entrada*: un conjunto X de vectores de la forma $\vec{x} = (x_1, \ldots, x_d)$ (en el caso del banco, X será el conjunto de posibles perfiles de los clientes).

• *Datos de salida*: un conjunto Y de posibles resultados (en el caso del banco, $Y = \{1, -1\}$ porque se trata de decidir si el cliente será un buen cliente o un mal cliente).

[1] Desde el punto de vista del banco, por supuesto.

• *Objetivo*: encontrar una función $\boldsymbol{f} : X \to Y$ que será la función de asignación óptima (para el banco, la fórmula ideal la para aprobación del crédito).

• *Datos*: los datos serán un conjunto de vectores de la forma

$$(\vec{x}_1, y_1), \ldots, (\vec{x}_N, y_N)$$

donde se espera que *N* (el número de datos) sea lo suficientemente grande para dar una buena aproximación a la fórmula real.

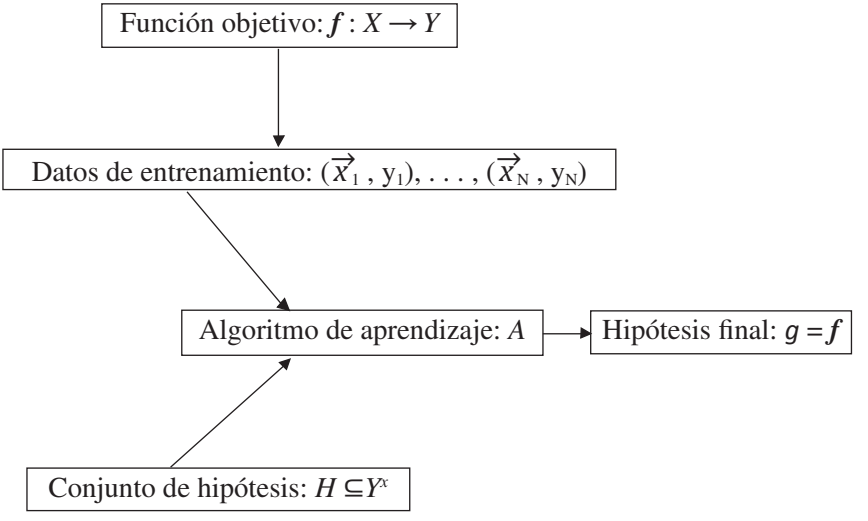
Componentes de la solución:

• *Conjunto de hipótesis*: un conjunto *H* de funciones $h: X \to Y$.

• *Algoritmo de aprendizaje*: *A*.

El conjunto de hipótesis y el algoritmo de aprendizaje, en conjunto, se conocen como el Modelo de Aprendizaje, y por supuesto, dependerán del tipo de programación (lineal, no lineal, dinámica, etc.) que queramos usar en el problema.

En el siguiente diagrama se resume la forma cómo una máquina realiza una tarea de aprendizaje automático:



En la siguiente sección presentaré algunas posibles tareas en las que se han usado de manera eficiente los modelos de aprendizaje automático, con el fin de comprender un poco más acerca de la naturaleza de este método.

3. Algunas aplicaciones del aprendizaje automático

En la sección anterior vimos cuáles son los principales conceptos e ideas detrás del aprendizaje automático. Como ya se dijo, este tipo de aprendizaje busca predecir el comportamiento de ciertas variables y encontrar patrones que permitan organizar información y tomar decisiones de manera más eficaz, en un sistema caótico del que no se conoce sino una gran cantidad de datos sin ninguna estructura.

En la última década, los sistemas de aprendizaje automático han cobrado mucha fuerza, sobre todo por la implementación exitosa de algoritmos de aprendizaje en campos mucho más prácticos. A continuación mostramos dos ejemplos que ejemplifican las principales aplicaciones en este campo: la predicción del comportamiento de un usuario, y el reconocimiento de imágenes.

En estos ejemplos, sin embargo, nos limitaremos a describir las componentes del aprendizaje en cada caso, dejando la explicación del algoritmo de aprendizaje para la siguiente sección.

A) *Netflix: ¿cómo sugerir películas al usuario?*

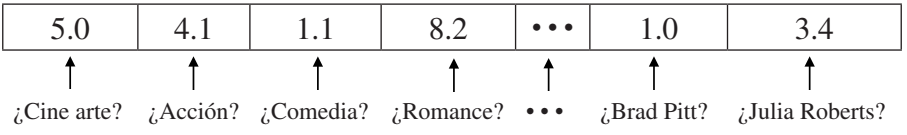
Para quienes no lo saben, *netflix* es un servidor desde el cual el usuario puede ver películas online. Una vez el usuario se ha registrado, *netflix* realiza recomendaciones de películas que posiblemente le gustarán al usuario, y como es de esperar, la compañía quiere mejorar el algoritmo que sugiere películas al usuario, pues en la medida en que estas sugerencias sean más o menos acertadas, el portal tendrá un mayor o menor número de usuarios y eso implicará un aumento o disminución en las ganancias[2].

La forma como se describe matemáticamente este problema es la siguiente: la idea es representar al usuario como un "vector de factores" dependiendo de los gustos del usuario. Esta información se puede obtener por una encuesta hecha directamente al usuario al realizar el registro.

Por ejemplo, el vector de factores puede tener componentes con números que vayan de 1 a 10, respondiendo en cada componente a una pregunta relacionada con las películas. Digamos, ¿al usuario le gustan las películas de acción?, ¿de comedia?, ¿al usuario le gusta el cine arte?,

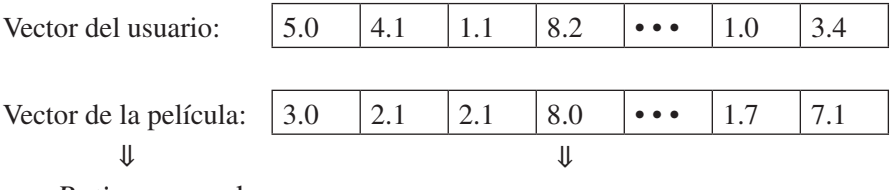
[2] De hecho, *netflix* tiene un premio de un millón de dólares para el equipo de programadores que logren una mejora de más de un 10% en el sistema de sugerencias. Para más información, véase http://en.wikipedia.org/wiki/Netflix_Prize.

¿al usuario le gustan las películas de Brad Pitt?, ¿las películas de Julia Roberts?, etc.



De la misma manera, a cada película se le puede asociar un vector de factores que responda a las preguntas: ¿la película tiene acción?, ¿contiene romance?, ¿actúa Brad Pitt?, etc.

La primera idea que se nos podría ocurrir es revisar el vector de factores de usuario y de película, revisar cuántas coincidencias se encuentran, y sumando todas las contribuciones lograríamos lo que llamaríamos el *rating* esperado para la película *X* dada por el usuario *Z*.



Rating esperado:

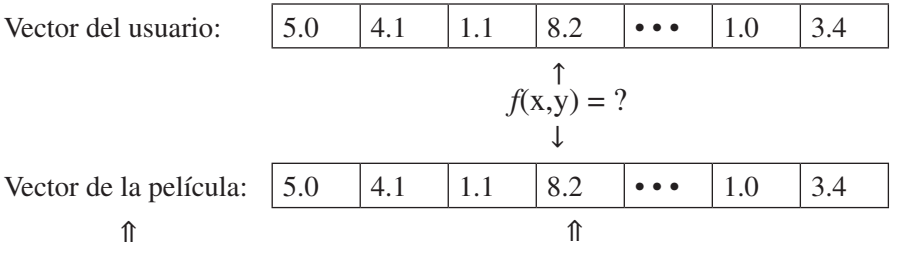
$$\boldsymbol{R} = \frac{1}{n}\sum_{i=1}^{n}(10 - |x_i - y_i|)$$

Existen dos problemas con los cuales un programador tendría que lidiar si usa este método: el primero es que si tenemos en cuenta la complejidad de los gustos que pueda tener un usuario, la encuesta tendría que ser muy exhaustiva, y seguramente ningún usuario estaría dispuesto a completarla de forma fidedigna. El segundo problema, es que para obtener el vector de factores de las películas, se tendría que contratar personas encargadas de ver las películas una a una y de llenar la información (mucho trabajo si pensamos en que *netflix* cuenta con miles de películas *online*), y lo peor, estos vectores de factores corresponderían a opiniones subjetivas de los evaluadores.

Para nuestra fortuna, ese proceso no es aprendizaje automático. La idea principal del aprendizaje automático es la de programar una máquina, y luego sentarse a tomar una taza de café mientras la máquina descubre

por sí sola cuál es el patrón del problema que estamos tratando, en este caso, la mejor forma de predecir el rating de una película.

Lo que haríamos en este problema usando aprendizaje automático, es tratar de hacer ingeniería inversa en este proceso: partiendo de *ratings* de películas hechos por usuarios previamente, la máquina trata de inferir coincidencias entre películas y perfiles de usuarios.



Rating esperado: $$\boldsymbol{R} = \frac{1}{n}\sum_{i=1}^{n}(10 - |x_i - y_i|)$$

En este punto existe una pregunta que deberíamos hacernos por prudencia: si la máquina está tratando de inferir un comportamiento global (la función de predicción perfecta) a partir de comportamientos específicos (los datos que se tienen, los *rankings* hechos previamente), ¿cuáles son las posibilidades de que la máquina encuentre efectivamente un patrón razonable?

La respuesta a esta pregunta es la siguiente: la idea no es que la máquina realiza este proceso con unos cuantos datos, sino con miles (o incluso millones) de datos, y en este caso la teoría de probabilidad y la ley de los grandes números nos dice que, si el número de datos es suficientemente alto, un patrón que dé fe de la información consignada en esos datos será un predictor casi perfecto del comportamiento global de la población.

B) *El Ipad y sistemas de reconocimiento de escritura a mano*

En los últimos años se ha popularizado el uso de las tabletas electrónicas, los *smartphones* y el *Ipad*. Estos dispositivos electrónicos cuentan con un sinfín de aplicaciones que ocuparía mucho tiempo mencionar

en este escrito, así que nos limitaremos únicamente a la capacidad que tienen estos dispositivos para reconocer escritura a mano realizada sobre la pantalla.

En primer lugar, tratemos de describir el problema en términos numéricos. Como es bien sabido, una imagen digital no es otra cosa que una grilla de muchos pixeles con un color en cada pixel, que en conjunto nos da la impresión de tener una imagen real. Para que una grilla de pixeles tenga el aspecto de una imagen verdadera, es necesario que el número de pixeles sea lo bastante grande como para no ser reconocible por el ojo humano. Es por esto por lo que hablamos de cámaras fotográficas de 5 megapixeles (5 millones de cuadrados en la grilla!!), o como el *Ipad* de última generación, que tiene 3,1 megapixeles por pulgada cuadrada (un total de 27 megapixeles).

De esta manera, podemos codificar una imagen como un vector de números reales (un número asociado al color en cada uno de los pixeles), y el problema radicaría en determinar si la imagen corresponde a una *A*, a una *O*, a un paréntesis, o a un árbol. Para describir el problema al que nos enfrentamos de manera más amigable, vamos a trabajar con una pantalla de 25 pixeles (una grilla de 5×5, y trataremos de atacar el problema de reconocer una *A* (mayúscula).

La dificultad del problema consiste en que no existe una única forma de escribir una *A*, y a pesar de que existen patrones para determinar si algo es una *A* mayúscula o no (línea oblicua hacia arriba, línea oblicua hacia abajo, una línea que atraviesa las dos), estos patrones no son susceptibles de programación debido a que la lectura del computador son unos cuadrados en blanco o negro, y algo tan fácil como "una línea oblicua hacia arriba" podría tener muchas interpretaciones dependiendo de lo hábil que sea el escritor para no torcerse.

Para ilustrar el problema, considere la figura 1 de la siguiente página, que muestra cuatro posibles *A* en nuestra cuadrícula de 5×5.

Los métodos tradicionales de programación para reconocimiento de objetos se basan en una de dos estrategias: o bien tienen una clasificación de todos los posibles objetos (en nuestro caso, posibles combinaciones de pixeles en blanco y negro) y de cuáles de ellos tienen la propiedad requerida o no (en este caso, ser una *A*); o bien tienen un algoritmo que permite reconocer en un tiempo específico cuándo un objeto tiene la propiedad requerida o no la tiene.

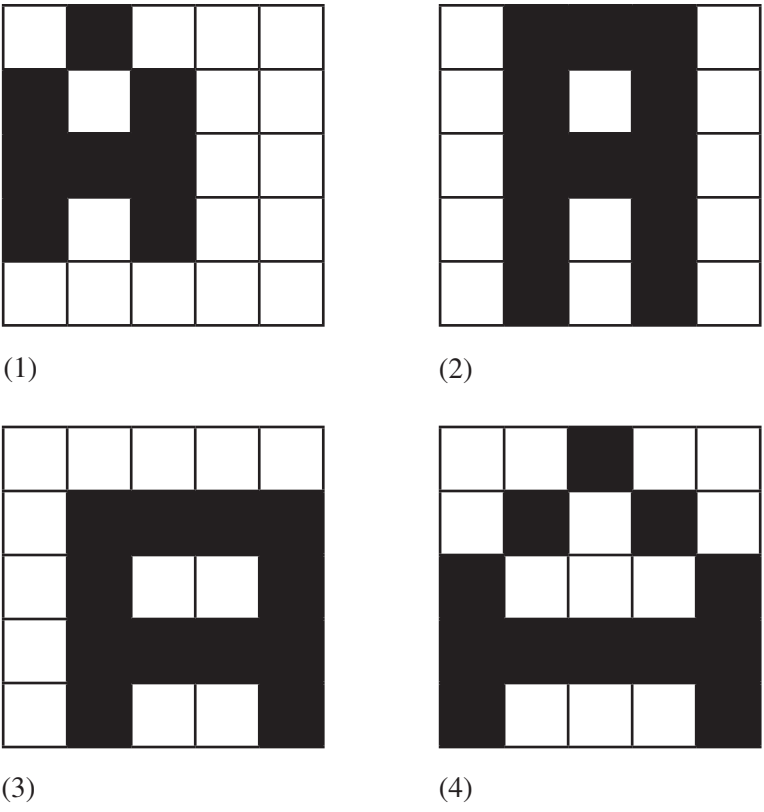


Figura 1. Diferentes versiones de *A* en la cuadrícula de 5 × 5

En el caso de reconocimiento de imágenes nos enfrentamos a los siguientes dos problemas:

1. La clasificación completa es imposible, y no me refiero en este caso a que es muy difícil de programar, sino que es técnicamente imposible. Por ejemplo, si tenemos una pantalla de 1 megapixel, el número total de diferentes objetos (cuadrículas con cuadrados blancos y negros) sería de $2^{1.000.000} > 10^{250.000}$. En este punto vale la pena recordar un dato curioso: los físicos han estimado que el número de átomos en el universo es de aproximadamente 10^{80}.

2. Debido a las diferentes formas de realizar una *A* usando pixeles —la *A* puede tener esquinas superiores de color blanco como en (1), o negro como en (2) y (3), pueden ser hechas con líneas rectas como en

(3) o con líneas "oblicuas" como en (4), etc.—, no existe *a priori* una regla conocida para determinar cuándo una imagen es una *A* o no lo es.

Es entonces cuando entran en juego los métodos de aprendizaje automático:

• ¿Existe un patrón? Ciertamente que sí. Cualquier persona con una mínima formación podría decidir de una imagen si es una *A* o no lo es.

•¿Podemos determinar un algoritmo matemáticamente? En el párrafo anterior respondimos a esta pregunta de forma negativa.

•¿Tenemos datos de entrenamiento? Sí. Es fácil obtener datos de entrenamiento (solo se le pide a personas que escriban arbitrariamente letras en la pantalla, y necesitamos un técnico que diga si es una *A* o no). Estos datos tendrán la forma $(\vec{x}, y)$ donde $\vec{x} = (x_1, \ldots, x_N)$ es un vector de tamaño N (número de pixeles en la pantalla) y cada componente x_i será 0 (si el *i-ésimo* pixel es blanco) o 1 (si el *i-ésimo* pixel es negro). El dato de salida y será 1 o −1, dependiendo de si la imagen en la pantalla corresponde a una *A* o no.

El responder apropiadamente a estas preguntas, y el uso de un algoritmo de aprendizaje apropiado, es lo que nos llevará a una respuesta satisfactoria mediante el uso de un modelo de aprendizaje automático.

4. Modelos de aprendizaje: el "perceptrón"

En esta sección presentaremos el modelo más estándar que se puede tener de una máquina de aprendizaje automático: el perceptrón. Las propiedades que definen a este modelo de aprendizaje son las siguientes:

• Conjunto de hipótesis: funciones lineales en los perfiles de los usuarios.

• Algoritmo de aprendizaje: perceptrón (PLA, por sus siglas en inglés)[3].

El algoritmo de aprendizaje del perceptrón funciona bajo la siguiente hipótesis:

El conjunto de datos de entrada es linealmente separable

que significa que existe un hiperplano H que separa los datos positivos (datos con resultados $y = 1$) de los resultados negativos (datos con resultados $y = -1$).

[3] PLA: Perceptron Learning Algorithm.

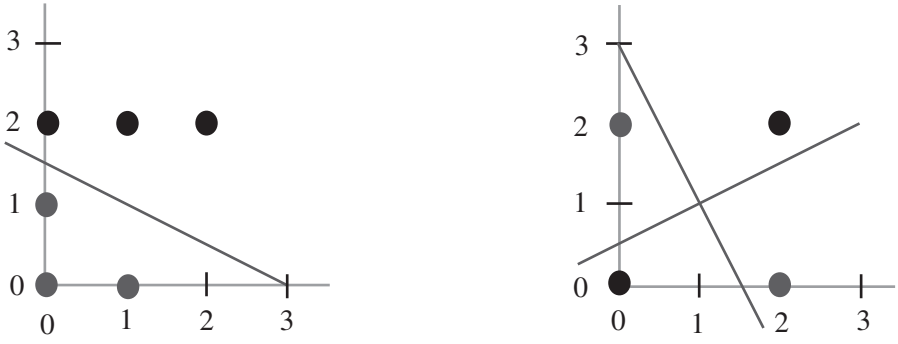


Figura 2. Ejemplo en $\mathbb{R}^2$ de datos que son linealmente separables y de datos que no son linealmente separables

Recordando el ejemplo del banco, los datos de entrada son $\vec{x}= (x_1 , \ldots , x_d)$ donde $x_1, \ldots , x_d$ son números reales que corresponden a la información del nuevo cliente. Podemos asignar un puntaje w_i a cada dato x_i (por ejemplo, podemos asignar un puntaje de 20 a cada $ 100.000 en el salario, o de −10 por cada comentario negativo en su historial crediticio, entre otros) y obtenemos así lo que podríamos llamar la puntuación crediticia del cliente:

$$\boldsymbol{P} = \sum_{i=1}^{d} w_i \cdot x_i$$

Como método de decisión, podemos fijar un umbral U tal que, si la puntuación es mayor que U entonces se concede el crédito, y si $P \leq U$, se niega el crédito.

Una forma de describir este criterio podría ser mediante la fórmula:

$$\boldsymbol{f}(\vec{x}) = sign\left(\left(\sum_{i=1}^{d} w_i x_i\right) - U\right)$$

donde la función $sign(\bullet)$ está dada por

$$sign(t) = \begin{cases} 1 \text{ si } t > 0 \\ -1 \text{ si } t \leq 0 \end{cases}$$

Si introducimos una variable artificial $x_0 = 1$, y escribimos $w_0 = -U$, podríamos escribir ahora

$$f(x) = sign\left(\sum_{i=0}^{d} w_i \cdot x_i\right) = sign\,(\vec{w}^{\,T}\ \vec{x}\,)$$

Es claro que esta función está completamente determinada por el vector $\vec{w} = (w_0, w_1, \ldots, w_d)$, y podemos tomar como conjunto de hipótesis al conjunto

$$H = \left\{ h(\vec{x}) = \sum_{i=0}^{d} w_i x_i : \ \vec{w} \in \mathbb{R}^{d+1} \right\}$$

que no es otra cosa que el conjunto de todas las funciones lineales con variables $x_0 = 1, x_1, \ldots, x_d$.

A continuación describiremos cómo funciona el algoritmo del perceptrón. En primer lugar tenemos la ecuación vectorial clásica que relaciona el ángulo entre dos vectores con el producto punto entre los mismos:

$$cos\,(\measuredangle\ \vec{w}, \vec{x}) = \frac{\vec{w} \cdot \vec{x}}{||\vec{w}|| \cdot ||\vec{x}||}$$

De esta manera, para la función objetivo, $f(\vec{x}) = 1$ si el ángulo θ entre $\vec{w}$ y $\vec{x}$ es menor que $\frac{\pi}{2}$, y -1 si $\frac{\pi}{2} \leq \theta \leq \pi$.

Suponiendo entonces que se haya escogido una función $f(\vec{x})$ (o equivalentemente que se haya seleccionado un vector $\vec{w} = (w_0, \ldots, w_d)$ entonces el dato $(\vec{x}_k, y_k)$ estará bien clasificado si $0 \leq \theta < \frac{\pi}{2}$ y $y_k = 1$, o $\frac{\pi}{2} \leq \theta \leq \pi$ y $y_k = -1$.

Algoritmo 4.1 (algoritmo de aprendizaje del perceptrón).

0. Input: $(\vec{x}_1, y_1), \ldots, (\vec{x}_N, y_N)$
   Variables de conteo: k, c
1. Determine el valor $w_0 = \vec{0} \in R^{d+1}$
2. Para $i = 1, 2, \ldots, N$, $\vec{x}_i := \dfrac{\vec{x}_i}{||\vec{x}_i||}$ haga (normalización)
3. Test

• Si $sign(\vec{w}_k \bullet x_{k+1}) = y_{k+1}$, haga $w_{k+1} = w_k$, $c := c_{+1}$

- Si $c = N$, $w = w_k$ y vaya a *output*

- Si $c < N$, haga $k = k + 1$ y vaya a *test*.

- Si $sign(\vec{w}_k \bullet \overrightarrow{x_{k+1}}) = y_k$, $\vec{w}_k := \vec{w}_k + y_{k+1} \cdot \overrightarrow{x_{k+1}}$, haga $c = 0$ y vaya a test.

4. *Output*: $f(\vec{x}) = sign(\vec{w} \bullet \vec{x})$

Como todo algoritmo, el algoritmo perceptrón tiene un desarrollo matemático que lo soporta; esto es, un teorema que demuestra que bajo las hipótesis dadas el algoritmo funciona correctamente.

Teorema 4.2. Si el conjunto de datos $(\vec{x}_1 , y_1), \ldots, (\vec{x}_k , y_k)$ es linealmente separable, entonces al finalizar el algoritmo perceptrón la función $f(\vec{x})$ clasifica correctamente los datos.

Demostración. Se divide en dos partes:

• El proceso algorítmico en el paso test no se repite indefinidamente.

Para probar este hecho, debemos recordar primero una propiedad importante de los números reales que muchas veces pasa por evidente pero es la base del análisis real:

Propiedad arquimediana en R: dados $a \in R$ y $\varepsilon > 0$, existe un número natural L tal que $a < L \bullet \varepsilon$.

Supongamos ahora que $f_k(\vec{x}_k) = y_k$. Podemos asumir sin pérdida de generalidad que $y_{k+1} = 1$ y $w_k \bullet x_{k+1} < 0$[4]. Tomando $a = -w_k \bullet x_{k+1}$ y $\varepsilon = \|x_k\| > 0$ (porque en todos los puntos se introdujo la variable $x_0 = 1$), obtenemos por la propiedad arquimediana que para algún $L_{k+1} \in \mathbb{N}$, $a < L_{k+1} \bullet \varepsilon$; esto es,

$$L_{k+1} \bullet \epsilon - a = \vec{w}_k \bullet \overrightarrow{x_{k+1}} + M_{k+1} (x_{k+1} \bullet \overrightarrow{x_{k+1}}) = (\vec{w}_k + L_k + 1\overrightarrow{x_{k+1}}) \bullet \overrightarrow{x_{k+1}} > 0.$$

Esto demuestra que la parte del algoritmo correspondiente al paso de test finaliza satisfactoriamente.

• El algoritmo completo finaliza satisfactoriamente:

Para esto, debemos comprobar que efectivamente el contador c alcanza el valor N. Para esto, bastaría con demostrar que el algoritmo perceptrón realiza un número finito de errores.

Dado que asumimos que los datos son linealmente separables, existe un vector $\vec{u}$ tal que para cada $1 \leq i \leq N$, $sign(\vec{u} \bullet \vec{x}_i) = y_i$. Sin pérdida de

[4] Si $y_{k+1} = -1$, basta con cambiar todos los signos.

generalidad, podemos asumir que el vector $\vec{u}$ es unitario, puesto que sea cual sea la magnitud de $\vec{u}$, define el mismo hiperplano dado por

$$H = \{ \vec{x} \in R^n : \vec{x} \bullet \vec{u} = 0 \}$$

que separa los datos positivos de los datos negativos.

Sea $\alpha = \text{mín} \{ | \vec{x}_i \bullet \vec{u} | : 1 \leq i \leq N \}$ (aquí consideramos los vectores $\vec{x}_i$ ya normalizados).

Hecho 1: si el algoritmo corrige en el paso $k + 1$, entonces $\overrightarrow{w_{k+1}} \bullet \vec{u} \geq \vec{w}_k \bullet \vec{u} + \propto$.

En efecto, si $y_k = 1$ entonces $\overrightarrow{w_{k+1}} = \vec{w}_k + \vec{x}_k$ y tenemos

$$\overrightarrow{w_{k+1}} \bullet \vec{u} = \vec{w}_k \bullet \vec{u} + \vec{x}_k \bullet \vec{u} \geq \vec{w}_k \bullet \vec{u} + | \vec{x}_k \bullet \vec{u} | \geq \vec{w}_k \bullet \vec{u} + \propto.$$

En el caso en que $y_k = -1$, $\overrightarrow{w_{k+1}} = \vec{w}_k - \vec{x}_k$ y de nuevo tenemos

$$\overrightarrow{w_{k+}} \bullet \vec{u} = w_k \bullet \vec{u} - \vec{x}_k \bullet \vec{u} \geq \vec{w}_k \bullet \vec{u} + | \vec{x}_k \bullet \vec{u} | \geq \vec{w}_k \bullet \vec{u} + \propto.$$

(Note que en este último, dado que el vector $\vec{u}$ clasifica correctamente todos los puntos, $\vec{x}_k \bullet \vec{u} < 0$ y así $| \vec{x}_k \bullet \vec{u} | := - \vec{x}_k \bullet \vec{u}$).

Hecho 2: $\| \overrightarrow{w_{k+1}} \|^2 \leq \| \vec{w}_k \| + 1$

Supongamos que $y_k = 1$, entonces tenemos:

$$\begin{aligned} \| \overrightarrow{w_{k+1}} \|^2 = \overrightarrow{w_{k+1}} \bullet \overrightarrow{w_{k+1}} &= (\vec{w}_k + \vec{x}_k) \bullet (\vec{w}_k + \vec{x}_k) \\ &= \| \vec{w}_k \|^2 + 2\vec{w}_k \bullet \vec{x}_k + \| x_k \|^2 \\ &\leq \| \vec{w}_k \|^2 + 2\vec{w}_k \bullet \vec{x}_k + 1 \leq \| \vec{w}_k \|^2 + 1 \end{aligned}$$

Esta última desigualdad se cumple pues si w_k se modifica en el paso $k + 1$, es porque clasifica incorrectamente el dato $\vec{x}_k$, es decir, $\vec{w}_k \bullet \vec{x}_k < 0$.

De forma análoga, si $y_k = -1$, $\overrightarrow{w_{k+1}} = \vec{w}_k - \vec{x}_k$ y $\vec{w}_k \bullet \vec{x}_k > 0$, por lo que al final obtenemos la misma desigualdad, a saber,

$$\| w_{k+1} \|^2 \leq \| \vec{w}_k \| + 1.$$

El hecho 1 implica que, luego de M correcciones, $\vec{w}_{M+1} \bullet \vec{u} \geq \alpha \bullet M$ (en cada corrección se añade un sumando de α). Por otra parte, el hecho 2 implica que $\| W_{M+1} \|^2 \leq M$.

Teniendo en cuenta que el vector $\vec{u}$ es unitario, y por la desigualdad de Cauchy-Schwarz, podemos concluir que

$$\alpha \bullet M \leq | \overrightarrow{W}_{M+1} \bullet \vec{u} | \leq \| \overrightarrow{W}_{M+1} \| \bullet \| u \| = \| \overrightarrow{W}_{M+1} \| \leq \sqrt{M}$$

y esta última desigualdad implica que $M \leq \frac{1}{\alpha^2}$ esto es, el algoritmo realiza a lo más $\frac{1}{\alpha^2}$ correcciones, y por lo tanto finalizará en a lo más $N \bullet \frac{1}{\alpha^2}$ pasos[5].

Para terminar este capítulo quisiera resaltar un punto importante: el algoritmo de aprendizaje solo da una fórmula que clasifica correctamente los datos de entrenamiento, pero dicha fórmula será útil solamente en la medida en que realice predicciones acertadas para los nuevos datos. Como ya se dijo anteriormente, esto se logra a medida que se tenga una cantidad considerable de datos.

5. Dimensión de Vapnik-Chervonenkis

La dimensión de Vapnik-Chervonenkis (también conocida como VC-dimensión), es una de los principales conceptos usados para medir la complejidad de una familia de conjuntos.

Durante veinte años, aproximadamente, este concepto formaba parte únicamente del área de la estadística descriptiva, pero fue en la última década del siglo xx cuando Chris Laskowski se dio cuenta de la relación existente entre la VC-dimensión y las teorías dependientes, impulsando de esta manera el estudio de las mismas en el contexto de la teoría de modelos.

Comenzaremos entonces hablando sobre qué es la VC-dimensión y cómo puede calcularse:

Definición 5.1. Supongamos que $\mathbf{C}$ es una familia de subconjuntos de X. Sea $S = \{x_1, \ldots, x_m\} \subseteq X$.

1. Decimos que S' es recortado de S por la familia $\mathbf{C}$ si existe algún conjunto $C \in \mathbf{C}$ tal que $S' = S \cap C$.

2. Definimos la proyección de S en $\mathbf{C}$ como:

$$\Pi_C(S) = \{C \cap S : C \in \mathbf{C}\}$$

3. Decimos que S es pulverizado por la familia $\mathbf{C}$ si todo subconjunto de S puede ser recortado por $\mathbf{C}$, o equivalentemente, si

[5] Es importante señalar que, gracias a que $\vec{u}$ separa correctamente los datos (hipótesis de separabilidad lineal), $|\vec{u} \bullet \vec{x_i}| > 0$ para todo $i \leq N$, y así, $\alpha = 0$.

$S \cap \mathbf{C} := \{S' \subseteq S : S' = S \cap C \text{ para algún } C \in \mathbf{C}\} = \wp(S)$[6]

Por ejemplo, si $X = R^2$ y $\mathbf{C} = \{H_\alpha = \{(x, y) : x + y > \alpha\} : \alpha \in R\}$ (los semiplanos definidos por rectas con pendiente -1 e intercepto positivo)

En este caso, si $S = \{(0, 0), (1, 1), (2, 2)\}$, tenemos lo siguiente:

• El conjunto $S' = \{(1, 1), (2, 2)\}$ es recortado de S por la familia $\mathbf{C}$, porque si tomamos $\propto = \frac{1}{2}$ entonces $C_\alpha \cap S = S'$ (note que $(0, 0) \notin \mathbf{C}\,\frac{1}{2}$).

• S no es pulverizado por la familia $\mathbf{C}$, porque no existe un semiplano en $\mathbf{C}$ capaz de separar el punto (1, 1) de los puntos (0, 0) y (2, 2).

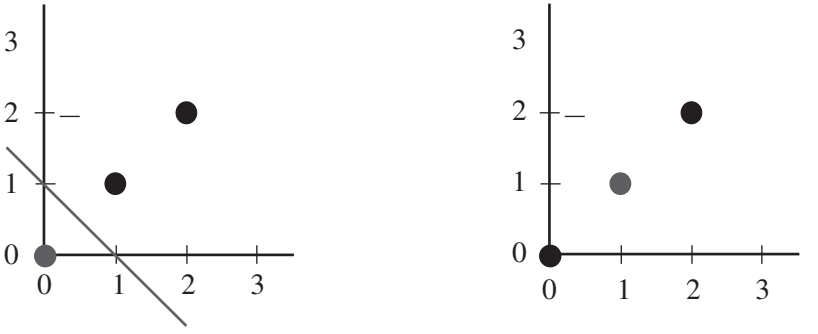


Figura 3. Conjunto de tres puntos que no pueden ser pulverizados usando hiperplanos.

Definición 5.2.

1. La VC-dimensión de una familia de conjuntos $\mathbf{C}$ se define como:

$d_{V\,C}(\mathbf{C}) = \text{máx}\{|S| : S \text{ es pulverizado por } \mathbf{C}\} \in N \cup \{\infty\}$

2. Definimos la función de complejidad como:

$$\Pi_C(m) = \text{máx}\{|\Pi_C(S)| : S \subseteq X, |S| = m\}$$

Podemos entender la VC-dimensión de una familia $\mathbf{C}$ como el máximo tamaño de un conjunto que $\mathbf{C}$ puede pulverizar, mientras que para cada m, $\Pi_C(m)$ representa el máximo número de subconjuntos "recortables" de un conjunto de tamaño m.

A) *Algunos ejemplos*

Para aclarar un poco los conceptos descritos, consideraremos cuatro ejemplos canónicos: rayos positivos, intervalos, perceptrones 2 dimensionales y conjuntos convexos. Los dos primeros ejemplos tendrán como

[6] $\wp(S)$ representa el conjunto potencia de S, la colección de todos los subconjuntos de S.

conjunto ambiente el conjunto de los números reales, mientras que los demás serán familias de subconjuntos del plano R^2. En todos los ejemplos, calcularemos explícitamente la VC-dimensión de cada una de las familias.

Vale la pena aclarar que para calcular la VC-dimensión de una familia de conjuntos (esto es, para hallar n tal que $d_{V\,C}(\mathbf{C}) = n$) se deben realizar dos pasos:

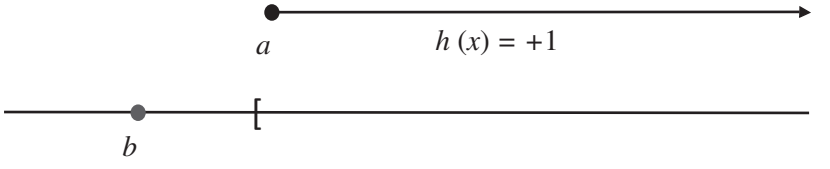
1. Se muestra que $d_{V\,C} \geq n$; esto es, se exhibe un ejemplo de un conjunto con n elementos que pueda ser pulverizado usando conjuntos en la familia $\mathbf{C}$.

2. Se muestra que $d_{V\,C} \leq n$; esto es, se demuestra que no existe ningún conjunto de n_{+1} elementos que pueda ser pulverizado usando elementos de $\mathbf{C}$.

Rayos positivos: consideramos la familia dada por

$$\mathbf{C}_r = \{[a, +\infty) : a \in R\}$$

En este caso, calcularemos explícitamente que $d_{V\,C}(\mathbf{C}_r) = 1$.



En primer lugar, si $S = \{b\}$ es un subconjunto de R con un elemento (m = 1), entonces $\{b\}$ puede ser pulverizado: los subconjuntos de $\{b\}$ son $\emptyset$ y $\{b\}$ y tenemos:

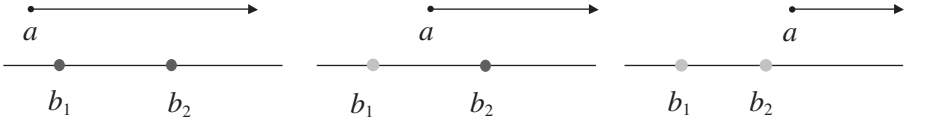
• Si $a < b$, entonces $S \cap [a, +\infty) = \{b\}$.

• Si $a > b$, entonces $S \cap [a, +\infty) = \emptyset$.

Con esto, acabamos de probar que $d_{V\,C} \geq 1$.

Debemos ver ahora que $d_{V\,C} \leq 1$; esto es, que ningún conjunto con dos o más elementos puede ser pulverizado por la familia $\mathbf{C}_r$.

Sea $S = \{b_1 , b_2 \}$ un conjunto con dos números reales. Podemos suponer sin pérdida de generalidad que $b_1 < b_2$. Considere el conjunto de S dado por $\{b_1 \}$. Si $S = \{b_1 , b_2 \}$ pudiera ser pulverizado por $\mathbf{C}_r$, entonces $S' = \{b_1 \}$ sería un conjunto recortado por C; esto es, para algún $C = [a, \infty)$ en $\mathbf{C}_r$, $S' = S \cap C$.



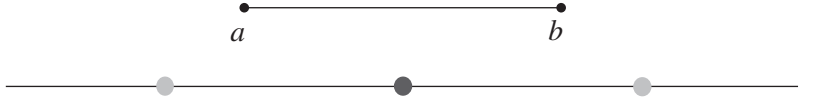
Sin embargo, tenemos las siguientes opciones para $S \cap C$:

$$S \cap C = \{b_1, b_2\} \cap [a, +\infty) = \begin{cases} \{b_1, b_2\} & \text{si } a \leq b_1 \\ \{b_2\} & \text{si } b_1 < a \leq b_2 \\ \{\theta & \text{si } a > b_2 \end{cases}$$

Como podemos ver, en ningún caso es posible obtener el subconjun. to $\{b_1\}$, y como ningún subconjunto de dos elementos puede ser pulverizado por $\mathbf{C}_r$, se concluye que $d_{V\,C} \leq 1$.

Intervalos: consideremos ahora la familia de intervalos con extremos reales; esto es, la familia dada por:

$$\mathbf{C}_{int} = \{(a, b) : a < b,\ a, b \in R\}$$



En este caso la VC-dimensión es 2, como veremos gráficamente:

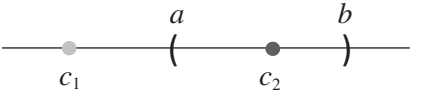
• $d_{V\,C} \geq 2$: podemos obtener todos los subconjuntos de un par $S = \{c_1, c_2\}$ de la siguiente manera:

$S \cap (a, b) = \theta$

$S \cap (a, b) = \{c_1\}$

$S \cap (a, b) = \{c_1, c_2\}$

$S \cap (a, b) = \{c_2\}$



• $d_{V\,C} \leq 2$: si $c_1 < c_2 < c_3$ son tres números reales, entonces no existe ningún intervalo abierto (a, b) tal que $(a, b) \cap \{c_1, c_2, c_3\} = \{c_1, c_2\}$. Por tanto, ningún conjunto de tres o más elementos es pulverizado por $\mathbf{C}_{int}$.

Perceptrones 2-dimensionales: para el algoritmo perceptrón, se usa la familia H_2 de todos los hiperplanos en R^2. Como veremos en la sección 5:B, la familia dada por el algoritmo perceptrón en R^d tiene VC-dimensión $d + 1$. En particular, $d_{VC}(H_2) = 3$.

Conjuntos convexos: estos son subconjuntos del plano con la propiedad de que cualquier línea que una dos puntos del conjunto estará totalmente contenida en el conjunto mismo.

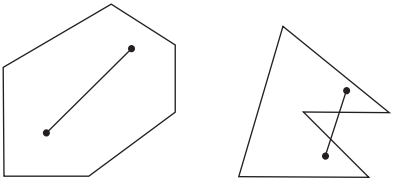


Figura 4. Ejemplo de un conjunto convexo y un conjunto no convexo en R_2

Consideremos la familia $\mathsf{C}_{conv} = \{C \subseteq R^2 : C \text{ es convexo}\}$.

Vamos a mostrar que $d_{VC}(\mathsf{C}_{conv}) = \infty$. Para probar esta afirmación se sigue una estrategia ligeramente diferente: lo que debemos demostrar es que $d_{VC} \geq n$ para todo $n \in N$; esto es, que la familia C_{conv} puede pulverizar conjuntos de tamaño n, para cualquier $n \in N$.

Dado $n \in N$, considere n puntos distribuidos en un círculo de manera uniforme, digamos $S = \vec{x}_1, \ldots, \vec{x}_n$. Dado $S' = \{\vec{x}_{i1}, \ldots, \vec{x}_{ik}\} \subseteq S$ podemos tomar el polígono C (conjunto convexo) cuyos vértices son precisamente $\vec{x}_{i1}, \ldots, \vec{x}_{ik}$.

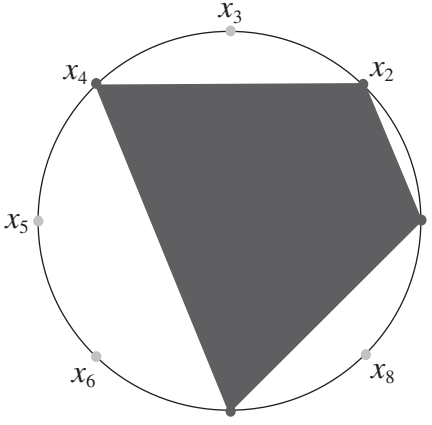


Figura 5. Polígono convexo generado por los puntos x_1, x_2, x_4, x_7.

De esta manera, $S' = C \cap S$ (los demás vértices quedarán fuera del polígono por estar ubicados en la circunferencia), lo que implica que $S = \{ \vec{x}_1, \ldots, \vec{x}_n \}$ puede ser pulverizado. Como este argumento puede repetirse para cada n, podemos inferir que $d_{VC}(\mathbf{C}_{conv}) \geq n$ para todo n; esto es, $d_{VC}(\mathbf{C}_{conv}) = \infty$.

B) *VC-dimensión en el algoritmo perceptrón*

Esta sección tiene un carácter un poco más técnico que las demás, y está dedicada a probar el siguiente teorema que caracteriza la VC-dimensión usada en el algoritmo perceptrón en $\mathbb{R}^d$ [7].

Teorema 5.3. *Sea H_d la familia de todos los semihiperplanos en R^d. Entonces, $d_{VC}(H_d) = d + 1$.*

Demostración. Como es usual, separamos la prueba en dos partes:

• $d_{VC}(H_d) \geq d + 1$: para esto, basta encontrar un conjunto de $d + 1$ punto en R^d que pueda ser pulverizado usando semihiperplanos.

$$A = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

Y tomemos $S = \{ \vec{x}_1, \ldots, \vec{x}_{d} + 1\}$ donde $\vec{x}_i$ es la *i-ésima* fila de la matriz A (note que la matriz A es invertible pues $det(A) = 1$).

Para probar que el conjunto S puede ser pulverizado, es necesario probar que para cualquier vector output $\vec{y}$ de la forma:

$$\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_d + 1 \end{bmatrix} = \begin{bmatrix} \pm 1 \\ \pm 1 \\ \vdots \\ \pm 1 \end{bmatrix}$$

[7] El lector que no esté tan interesado en los detalles técnicos, puede saltar esta sección sin ningún problema.

existe un hiperplano H que separa los elementos con *output* + 1 de los elementos con *output* −1, o equivalentemente, que existe un vector $\vec{w}$ que cumple con la ecuación $sign(A\vec{w}) = \vec{y}$.

Como A es invertible, podemos tomar $\vec{w} = A^{-1}\vec{y}$, y así:

$$A\vec{w} = A(A^{-1})\vec{y} = \vec{y}$$

y en particular, $sign(A\vec{w}) = \vec{y}$. Concluimos entonces que el conjunto S puede ser pulverizado, y así, $d_{VC}(H_d) \leq d + 1$.

• $d_{VC}(H_d) \leq d + 1$: para probar esta afirmación, es necesario ver que ningún conjunto de $d + 2$ puntos en R^d puede ser pulverizado por la familia H_d .

Supongamos que $S = \{\vec{x}_1, \ldots, x_d + 2\}$ es un conjunto de $d + 2$ puntos en R^d. Para ver estos puntos como vectores que pasan por el origen, podemos formar los siguientes vectores en R^{d+1} :

$$\vec{x}_1' = (1, \vec{x}_1), \vec{x}_2' = (1, \vec{x}_2), \ldots, \vec{x}_{d+2'} = (1, \vec{x}_{d+2})$$

Ahora, como la dimensión de $\mathbb{R}^{d+1}$ es d+1, al tener d+2 vectores en $\mathbb{R}^{d+1}$ estos deben ser linealmente dependientes; esto es, para algún j con $1 \leq j \leq d + 2$ tenemos que

$$\vec{x}_j' = \sum_{i \neq j} a_i \vec{x}_i'$$

Considere el conjunto $S' = \{\vec{x}_i' : a_i > 0\}$ (poniendo $a_j = -1$ por definición). Debemos ver ahora que ningún perceptrón puede recortar S' del conjunto S; esto es, no existe ningún vector $\vec{w}$ tal que $w^T x_i = sign(a_i)$ para cada $i = 1, 2, \ldots, d + 2$.

Si dicho vector existiera, entonces al multiplicar por W^T en la ecuación, obtendríamos:

$$w^T \vec{x}_j = \sum_{i \neq j} a_i w^T \vec{x}_i$$

En la parte derecha, $a_i w^T \vec{x}_i > 0$ puesto que $sign(W^T \vec{x}_i) = sign(a_i)$ para cada $i = j$. Sin embargo, $sign(W^T \vec{x}_j) = sign(a_j) = -1$, lo que es absurdo.

Concluimos entonces que ningún conjunto de tamaño $d + 2$ puede ser pulverizado por el perceptrón, por lo cual, $d_{VC}(H_d) \leq d + 1$.

C) *Dimensión finita vs. dimensión infinita*

En una familia arbitraria de conjuntos $\mathbf{C}$, la VC-dimensión puede tomar valores finitos, o el valor $d_{VC} = \infty$. El lema de Sauer-Shelah es la roca que fundamenta la idea más importante en la teoría de aprendizaje automático:

Si la VC-dimensión de $\mathbf{C}$ *es finita, entonces una máquina puede aprender a diferenciar los elementos de C.*

Teorema 5.4 (lema de Sauer-Shelah). Supongamos que $VC - dim(\mathbf{C}) = d < \infty$. Entonces,

$$\Pi_C(m) \leq \Phi_d(m) := \sum_{i=0}^{d} \binom{m}{i} \leq \left(\frac{em}{d}\right)^d = O(m^d)$$

Demostración. Esta prueba la realizaremos por inducción sobre $k = m + d$, con $k \geq 2$. Teniendo en cuenta que $m, d \geq 1$, el caso $k = 2$ implicaría que $m = 1 = d$. Si tomamos un conjunto $\{x\}$ de tamaño $m = 1$ tendríamos que

$$\Pi_C(1) \leq |\{\emptyset, \{x\}\}| = 2 = \binom{1}{0} + \binom{1}{1} = \sum_{i=0}^{1} \binom{1}{i}$$

Supongamos ahora que tenemos el resultado para cualquier conjunto S' y cualquier conjunto de hipótesis H' tal que $|S'| + VC\,dim(H') \leq k - 1$.

Sea ahora H un conjunto de hipótesis con $VCdim(H) = d$ y S un conjunto de tamaño m, tales que $m + d = k$.

Sea s un elemento arbitrario en S, y definamos

$$H' = \{H \in \Pi_H(S) : s \notin H, H \cup \{s\} \in \Pi_H(S)\}$$

Así, un conjunto H está en H' si y solo si se cumplen las siguientes condiciones:

1. $H \subseteq S - \{s\}$
2. $H \in H, H \cup \{s\} \in H$.

• Afirmación 1: $\Pi_{H'}(S) = \Pi_{H'}(S - \{s\})$

Recordemos que los elementos $\Pi_{H'}(S)$ son todos los subconjuntos de $S' \subseteq S$ tal que $H \cap S = S'$ para algún $H \in H$. Por la propiedad (1), los elementos de H son subconjuntos de $S - \{s\}$, por lo cual para todo $H \in H'$, $S' = H \cap S = H \cap (S - \{s\})$.

• Afirmación 2: $V\,Cdim(H') \leq d - 1$.

Supongamos que no. Entonces existe un conjunto de $X = \{x_1, \ldots, x_d\}$ tal que

$$\Pi_{H'}(X) = \{H \cap X : H \in H'\} = \wp(X)$$

En particular, existe algún elemento $H \in H'$ tal que $X = X \cap H$, y por (1) tendríamos que

$$X = X \cap H \subseteq X \cap (S - \{s\}) \subseteq X; \text{ esto es, } X \subseteq S - \{s\}.$$

Considere $X \cup \{s\} = \{x_1, \ldots, x_d, s\}$ y $Y \subseteq X \cup \{s\}$. Si s 6$\in Y$, $Y \subseteq X$ y existe un elemento $H \in H'$ (y por (2), también en H) tal que $Y = H \cap X = H \cap (X \cup \{s\})$. Si $s \in Y$, entonces $Y - \{s\} \subseteq X$ y como X es pulverizado por H', existe $H \in H'$ tal que $Y - \{s\} = H \cap X$. No obstante, como H es un elemento de H' por (2), $H \cup \{s\}$ es un elemento de H, por lo cual $Y = (H \cup \{s\}) \cap (X \cup \{s\})$.

Acabamos de probar entonces que el conjunto $X \cup \{s\} = \{x_1, \ldots, x_d, s\}$ es pulverizado por el conjunto H, contradiciendo el hecho de que $V\,Cdim(H) = d$.

Con los resultados obtenidos podemos argumentar como sigue:

$$|\Pi_H(S)| = |\Pi_H(S-\{s\})| + |\Pi_{H'}(S)|$$
$$= |\Pi_H(S-\{s\})| + |\Pi_{H'}(S-\{s\})|$$
$$\leq \Phi_d(m-1) + \Phi_{d-1}(m-1) \qquad \text{(Por afirmación 1.)}$$
$$= \sum_{i=0}^{d} \binom{m-1}{i} + \sum_{i=0}^{d-1} \binom{m-1}{i} \qquad \text{(Por hipótesis de inducción)}$$
$$(m-1) + d = m + (d-1) = k + 1$$
$$= 1 + \sum_{i=1}^{d} \binom{m-1}{i} + \sum_{i=1}^{d} \binom{m-1}{i-1}$$
$$= 1 + \sum_{i=1}^{d} \left[\binom{m-1}{i} + \binom{m-1}{i-1} \right] \qquad \text{(intercambiando contadores)}$$
$$= 1 + \sum_{i=1}^{d} \binom{m}{i} = \sum_{i=0}^{d} \binom{m}{i} = \Phi_d(m)$$

El hecho de que $\Phi_d(m) = O(m^d)$ se debe a que el término dominante de $\Phi_d(m) = \sum_{i=1}^{d} \binom{m}{d}$ es:

$$\binom{m}{d} = \frac{m!}{d!(m-d)!} = \frac{m\,(m-1)\cdots(m-d+1)}{d!} \leq m\,(m-1)\cdots(m-d+1)$$
$$= m^d + \cdots = O(m^d)$$

Este teorema de tipo puramente combinatorio nos presenta una nueva dicotomía entre las familias de conjuntos: si la VC-dimensión de una familia C es finita, entonces la complejidad del sistema es computable, en el sentido que el número de patrones reconocibles tiene crecimiento polinomial con respecto al número de parámetros que se usan para definir la familia.

Por tanto, cuando la VC-dimensión es finita, existe un algoritmo que resuelve en un tiempo polinomial el problema de saber si el conjunto de datos se puede diferenciar usando elementos en la familia $\mathbf{C}$.

6. ¿Cómo podemos interpretar la VC-dimensión?

Luego de ver cómo se calcula la VC-dimensión en algunos casos particulares, y de mostrar que la VC-dimensión es finita precisamente cuando el aprendizaje para la máquina es programable, vamos a discutir cómo podemos interpretar la VC-dimensión.

A) *VC-dimensión como grados de libertad*

Desde el punto de vista intuitivo, los grados de libertad en un problema corresponden a la cantidad de posibles direcciones en las cuales puede variar una solución. En el aprendizaje, un mayor grado de libertad se relaciona con una mayor complejidad del algoritmo de aprendizaje (mayor complejidad en la familia C), y por tanto a una VC-dimensión mayor.

Una forma de agregar grados de libertad al problema es a través de los parámetros: los números reales usados para definir los elementos en la familia C. Revisemos los ejemplos que hemos estudiado hasta ahora:

• *Rayos positivos*: un rayo positivo es un conjunto de la forma $[a, +\infty)$, y para poder definirlo es necesario un parámetro: el número real a. Además, sabemos que $d_{VC}(\mathbf{C}_r) = 1$.

• *Intervalos*: para definir un intervalo (a, b) son necesarios dos parámetros, y tenemos que $d_{VC}(\mathbf{C}_{int}) = 2$.

• *Perceptrones*: para definir un semihiperplano como los que se usan en el algoritmo perceptrón, es necesario dar un vector que tiene la forma $\vec{w} = (w_0, w_1, \ldots, w_d)$. De nuevo, el número de parámetros coincide con la VC-dimensión, pues $d_{VC}(H_d) = d + 1$.

Pareciera haber entonces una relación entre el número de parámetros que se usan para definir los elementos de una familia, y la VC-dimensión de dicha familia. Sin embargo, esto no sería del todo acertado: si consideramos una cadena de perceptrones actuando sobre un vector $\vec{x}$.

En este caso, la VC-dimensión no cambia (los conjuntos que son pulverizados con un perceptrón son los mismos conjuntos pulverizados por dos o más perceptrones), pero el conjunto de parámetros está creciendo. Este ejemplo es lo que llamaríamos un "ejemplo gracioso", puesto que aunque el número de parámetros está creciendo, dichos parámetros no contribuyen con un aumento en los grados de libertad.

Vemos entonces que la VC-dimensión realmente está midiendo el número efectivo de parámetros; esto es, el mínimo número de parámetros necesarios para definir la familia de conjuntos $\mathbf{C}$.

B) *VC-dimensión y entrenamiento necesario*

En aprendizaje automático probabilístico, la celebrada fórmula de Vapnik-Chervonenkis está dada por:

$$\mathbb{P}(|E_{input}(g) - E_{output}(g)| > \in) \leq 4\, \Pi_c(2N) e^{-1/8 \in^2 N}$$

Usando el lema de Sauer-Shelah y haciendo un estimativo asintótico del término de la derecha, obtendríamos la fórmula:

$$\mathbb{P}(|E_{input}(g) - E_{output}(g)| > \in) \leq N^d e^{-N}$$

donde N es el número de datos de entrenamiento, y e es el margen de error; esto es, qué tan cerca queremos que esté la hipótesis propuesta (función resultante del algoritmo de aprendizaje) y la hipótesis real (función que separa correcta y efectivamente los datos de entrenamiento).

Usando algunos elementos del Cálculo Diferencial, se puede ver que siempre que la VC- dimensión d sea finita, el término $N^{\underline{d}} e^{-N}$ cuando N tiende a infinito. Esta observación coincide con el hecho innegable de que a mayor entrenamiento, mejor calidad de la función hipótesis.

Sin embargo, deseamos que $N^d e^{-N}$ tome un valor cercano a cero (para que la probabilidad de sobrepasar el margen de error sea muy baja). De esta manera, la cantidad de datos necesarios para un entrenamiento satisfactorio crece en forma exponencial con la VC-dimensión, y esto proporciona un punto de parada para una máquina programada para realizar una tarea de aprendizaje.

REFERENCIAS

[1] Ethem Alpaydin. *Introduction to machine learning*. The Massachusetts Institute of Tecnology (MIT) Press, Cambridge, Massachusetts, 2010.

[2] Matthias Aschenbrenner, Alf Dolich, Deirdre Haskell, Dugald Macpherson, Sergei Starchenko. *Vapnik-Chervonenkis density in some theories without the independence property*, I and II Preprint. Septiembre 2011. arXiv: 1109.5437v2 [math.LO].

[3] Luc Devroye, László Györfi, Gábor Lugosi. *A probabilistic theory of pattern recognition*.

Preprinted book, 2009.

[4] Václav Hlavác. *Vapnik-Chervonenkis learning theory*. Czech Technical University, Department of Cybernetics, 2008.

[5] Pascal Koiran. *Vapnik-Chervonenkis dimension of recurrent neural networks*. Laboratoire de l'Informatique du Parallélisme. Ecole Normale Supériore de Lyon – cnrs, 1996.

[6] Marek Karpinski y Angus macintyre. *Polynomial bounds for VC-dimension of sigmoidal*

neural networks. Electronic Colloquium on Computational Complexity, 1994.

[7] Xu Miao, Lin Liao. *VC-dimension and its applications in machine learning*. Preprint, 2010.

[8] Hung Q. Ngo. *Three proofs of Sauer-Shelah lemma. Computational learning theory.* SUNY at Buffalo. Fall, 2010. Lecture notes.

[9] Alex Smola. S.V.N. Vishwanathan. Introduction to machine learning. Yahoo! Labs., Santa Clara. Department of Statistics and Computer Science. Purdue University, Cambridge University Press, 2008.

[10] V. N. Vapnik. *The nature of statistical learning machine*. Springer-Verlag Press, 1995.

[11] Nicolas Vayatis. *The role of critical sets in Vapnik-Chervonenkis theory*. Equipe de Modelisation Aleatoire. Université Paris x, 2000.

Capítulo II

UNA INTRODUCCIÓN AL MÉTODO NATURAL DEL APRENDIZAJE DE LAS MATEMÁTICAS, EN LA PRIMERA INFANCIA. ESTADO DEL ARTE.

John Edison Castaño Giraldo*

Resumen

Investigaciones previas en cuanto al desarrollo del pensamiento ma temático, en la primera infancia, muestran que debe realizarse éste de forma natural, para no interrumpir la capacidad investigativa y deductiva de los niños. Por tanto, se explica el método natural para el aprendizaje de la matemática en la primera infancia, sus procesos naturales (basados en el desarrollo histórico de la creación de los procesos matemáticos) y sus estadios, deben corresponder a una secuencia didáctica clara y organizada para que acompañe, para tener resultados exitosos. Además, cada operación matemática se muestra como una consecuencia del intento de solucionar una necesidad vital para el aprendiz o matematizador de la realidad;- es así cómo la suma y la resta tienen un papel fundamental en los procesos básicos. Por último, se muestra cómo debe enfrentarse una situación problema para evidenciar así una aplicación clara de cada uno de los objetos matemáticos obtenidos.

1. Justificación

Es necesario conocer y promulgar estrategias de aprendizaje actualizadas y eficientes para las nuevas generaciones, útiles no solo para adquirir conocimiento, sino también para identificar situaciones en las que deba aplicar este conocimiento y resolver esa situación de manera eficiente.

Es oportuno realizarlo, ya que las nuevas generaciones presentan dificultades evidentes al momento de justificar sus conocimientos en

* Docente Investigador

matemáticas; es por esto por lo que se necesitan nuevas prácticas pedagógicas y nuevos ambientes de aprendizaje.

El conocerlo le permite a los docentes tener un soporte para que sus enseñanzas sean más eficientes y para los estudiantes es bastante útil debido a su interés de formación.

2. Diseño metodológico

La investigación se desarrollará a partir de un diseño cualitativo, específicamente aplicando el modelo de la Teoría Fundamentada. Se selecciona este modelo porque el énfasis de la investigación es principalmente la construcción de la teoría a partir de un fenómeno que aunque ha sido estudiado, requiere de un análisis más profundo para determinar cuáles son las características de los entornos de aprendizaje que favorecen el aprendizaje de las matemáticas.

Gracias al tipo de diseño, la definición de variables se determinará de acuerdo con el análisis que se vaya realizando de los datos encontrados y la información que estos aporten a los propósitos de la investigación. De esta manera, las características de población y muestra se determinarán a partir de dicha definición de variables.

3. Antecedentes de la investigación

Se hablará sobre métodos de aprendizaje para matemáticas basados en el método natural y en matemáticas para la vida.

¿Por qué el rendimiento académico en matemáticas es tan bajo?, ¿por qué el nivel de comprensión de las matemáticas es tan deficiente y la motivación para estudiar matemáticas es tan poca? Las matemáticas no se apropian de una manera correcta si no se ve la utilidad clara de la misma, si no se interactúa de manera constante con las matemáticas, es muy difícil aceptarla y utilizarla de forma correcta. La forma de motivarse a estudiar matemáticas tiene que ver con las situaciones problema, que estas le permitan resolver, con la interacción que esta ciencia le permita tener al estudiante con su realidad.

Investigaciones previas

• Creación del método natural: “El desarrollo del pensamiento matemático en la primera infancia”, Carlos Díez - Leonardo Pantano.

• "Ponencia de matemáticas para la vida: una aproximación algorítmica a la resolución de problemas en matemáticas", Carlos Díez - Jorge Luis García - Juan Ramón González.

4. Método natural: enseñanza de la matemática en la primera infancia

Los procesos para el aprendizaje de la matemática se asimilan con el aprendizaje natural, la interacción con el medio que los rodea y se debe motivar la construcción por parte del actor o del educando. Se definen los siguientes procesos, que se desarrollan de la misma manera en la que el ser humano fue reconociendo naturalmente la matemática como una herramienta para solucionar las situaciones que se le presentan: *Los seres humanos somos matemáticos por naturaleza, es decir, somos matemáticos por necesidad*[1].

- Asignación
- Agrupación no posicional
- Agrupación posicional
- Agregación
- Diferencia
- Suma aritmética
- Resta aritmética
- Transformación aditiva
- Comparación aditiva
- Estructura multiplicativa
- Proceso de división

Los dos últimos procesos serán objeto de desarrollo en un próximo artículo.

Cada uno de estos procesos tiene unas etapas que el autor llama "estadios".

A) *Asignación*

Se refiere al proceso inicial y se encuentra relacionado con una técnica para contar; el proceso consiste en *asignar* al objeto que se quiere contar

[1] Introducción, "El desarrollo del pensamiento matemático en la primera infancia", Carlos Alberto Díez - Leonardo Pantano.

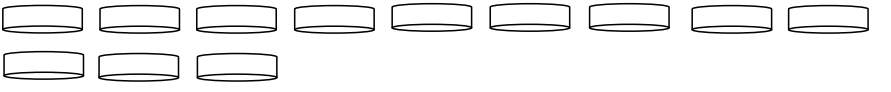
otro objeto que lo represente. El estadio inicial corresponde a diferenciar entre un objeto y muchos objetos; en esta etapa el niño desarrolla su habilidad para reconocer cantidades. En seguida el niño asigna a un nivel más general, aún sin realizar representaciones tan abstractas como la de utilizar números, y es capaz de expresar, evidenciando con los dedos de su mano por ejemplo, la cantidad de objetos que posee. A continuación el niño deberá utilizar una estrategia sistemática para realizar el conteo, pueden ser tan básicas como girar el objeto, marcar el objeto, desplazar a otro lado el objeto contado. Es importante que se desarrolle este orden para realizar el conteo. Como estadio siguiente, se diferencia entre una cantidad de objetos y otra, para decidir las cantidades mayor, menor o igual; en esta etapa es importante poder realizar asignaciones, pues el personaje podrá reemplazar los objetos por sus elementos de conteo para así evitar realizar desplazamientos. Por ejemplo, mover cosas demasiado grandes, cuando las puede representar por medio de fichas.

B) *Agrupación no posicional*

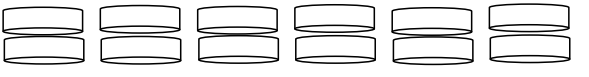
En este proceso se fortalece la necesidad que surge a continuación de realizar asignaciones objeto a objeto (este último más general), y es la de asignar a *grupos de objetos* con la misma cantidad un elemento de conteo; estos elementos deben poderse diferenciar para evitar confusiones. Es muy importante tener en cuenta que en la etapa inicial al realizar agrupaciones en una cantidad de objetos, esta debe mantenerse para las siguientes agrupaciones, todo con el fin de reducir la cantidad de símbolos.

Explicación

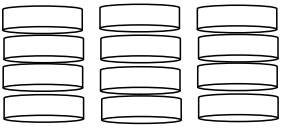
Si se tiene esta cantidad de elementos



Y la primera agrupación corresponde a realizar parejas de objetos.



Si se quiere realizar de nuevo agrupaciones, debe hacerse otra vez en parejas.



Y así sucesivamente.

Esto lleva a los autores a definir binas, binas de binas y binas de binas de binas.

El estadio inicial corresponde a realizar agrupaciones en grupos muy grandes en los que se tiene la necesidad de contar; se puede realizar agrupaciones en grupos más pequeños para contarlos luego como si fueran elementos. El estadio siguiente debe evidenciar el estado natural de los niños de reconocer la cantidad de objetos de un grupo a simple vista, tratando de reducir el proceso de contar uno por uno los objetos; es importante la forma de ubicar los objetos, de esta manera se pueden hacer asociaciones rápidas de la cantidad. A continuación, el niño debe contar utilizando nuevamente objetos abstractos pero sin cambiarlos de color; es decir, no relaciona una nueva agrupación asociándola con un nuevo color, sino que realiza tantas agrupaciones como se lo permitan la cantidad inicial de fichas y de acuerdo con la base, entonces aparecen términos como binas (refiriéndose a dos objetos), binas de binas (para cuatro) y así sucesivamente. Luego es necesario avanzar al estadio siguiente, el cual consiste en cambiar de color para representar una nueva agrupación; entonces, al contar dos fichas de un mismo color el niño habla de una bina y las reemplaza por un nuevo objeto pero de otro color, así esta nueva agrupación reduce el número de objetos utilizados. En este estadio no es importante el color; por el contrario, debe variarse de color para que el aprendiz reconozca que cualquier color puede representar una agrupación de nivel mayor. Como nota del autor[2], es importante que al cambiar un color para representar un nivel de agrupación mayor se utilicen las dos manos: una que quita las fichas y la otra que ubica las nuevas fichas, esto para introducir la idea de equivalencia. Por último, y si el proceso se ha llevado a cabo con éxito, el estudiante estará en la capacidad de representar los objetos que no corresponden a la agrupación; por ejemplo, al agrupar en ternas 10 objetos podrá decir: tres ternas de ternas y una unidad; es importante la forma en que se lee la agrupación de mayor a menor, pues así podrá en un futuro encontrar las agrupaciones no existentes en el conteo. Debe procurarse por realizar el conteo de forma sistemática; una buena forma puede ser tachando los objetos ya contados y cuando se agrupen en niveles superiores se debe cambiar de color, como se hacía antes con los objetos abstractos.

[2] "El desarrollo del pensamiento matemático en la primera infancia", Carlos Alberto Díez - Leonardo Pantano.

C) *Agrupación posicional*

Luego de realizar agrupaciones sistemáticamente, es necesario que el estudiante reconozca ahora que un símbolo no tiene su mayor importancia en la forma o en el color, sino en su posición (esto lo llevará a comprender mejor su sistema de numeración como un sistema de numeración posicional, y que está en base 10). Las reglas para esta agrupación consisten: primero, no debe haber en una posición más elementos que los de la base; segundo, si es así, debe agruparse nuevamente y cambiará de casilla. Se debe utilizar una tabla de conteo (véase tabla 1).

Tabla 1. Tabla de conteo

			UNIDADES

En un principio, solo se nombran las unidades y no las demás posiciones que serán agrupaciones de unidades dependiendo de la base que se escoja, pero no es necesario rotularlos, sería más interesante que lo intentara hacer naturalmente.

Los estadios son los siguientes:

• *Cuente posicionalmente cantidades que requieren agrupación simple*. No se hacen agrupaciones de agrupaciones. Este objeto consolida el hecho de utilizar menos objetos cada vez para cantidades más grandes de objetos. Y los pasos a seguir son los siguientes: primero, agrupe en una casilla superior utilizando los mismos elementos iniciales, el avance es que cambie de posición con cada agrupación; luego, como en la agrupación no posicional, cambie una agrupación superior por otro elemento, aunque esto no es lo ideal, pues el niño debe reconocer que la agrupación cambia de casilla por su posición y no por su forma o color, y por último, reemplace cada agrupación superior por un objeto similar

al inicial, pero que represente una agrupación superior porque cambia de casilla; este era el objetivo.

• *Cuente cantidades que requieren agrupación compuesta*. En este momento el estudiante debe haber comprendido muy bien la noción de agrupar posicionalmente, pues ahora va agrupando en casillas superiores dependiendo de las agrupaciones superiores; por ejemplo, para agrupar 15 objetos el resultado será una terna de ternas, dos ternas y cero unidades.

• *Haga agrupaciones sin retirar objetos de las casillas*. Esto lo hace en el papel y puede utilizar el conteo sistemático.

D) *Agregación*

Nace de la necesidad de evitar contar de nuevo, si se requiere contar dos cantidades el estudiante tiene dos opciones: contar dos veces o realizar una agregación; el objetivo es poder realizarla simbólicamente en un futuro próximo. Se deben realizar dos pasos: primero agregar y luego agrupar. Utilícese una tabla de agregación (véase tabla 2).

Tabla 2. Tabla de agregación

	UNIDADES DE MIL	CENTENAS	DECENAS	UNIDADES
Agrupación				
Cantidad 1				
Cantidad 2				
Agregación				

Los estadios son los siguientes:

• *Agregue sin necesidad de agrupar*. Es importante que el estudiante reconozca que la agregación se realiza entre objetos de la misma naturaleza; esto será muy conveniente al momento de estudiar el proceso de la suma y después la generalización de sumas como lo es el álgebra o el cálculo, pues no tendrá aquellos errores que tradicionalmente cometemos los que hemos aprendido con la matemática tradicional. Con en este método es importante ofrecer un proceso general para que el estudiante pueda identificar de forma natural los procesos particulares, pues esto es muy diferente a los métodos tradicionales, que ofrecen procesos particulares para que el estudiante los particularice. En este primer estadio se agrupan las cantidades agregadas con el ánimo de que el niño naturalmente particularice y llegue al siguiente estadio.

• *Agregue sin realizar agrupaciones*. Este estadio es el paso natural que debe realizar el estudiante, cuando vea que no es necesario realizar agrupaciones.

E) *Diferencia*

En este proceso deben revertirse los pasos del proceso de agregación; se reconocen dos pasos: primero, debe comprobarse si los objetos de la *cantidad 1* en cada casilla son mayores o iguales a los de la casilla dos; de no ser así, debe desagruparse la posición inmediatamente superior, para obtener más elementos en la posición anterior, y luego s**í** puede realizarse la diferencia. Debe utilizarse una tabla de diferencia (véase tabla 3).

Tabla 3. Tabla de diferencia

	UNIDADES DE MIL	CENTENAS	DECENAS	UNIDADES
Cantidad 1				
Cantidad 2				
Diferencia				

Los estadios son los siguientes:

• *Haga diferencias en las que es necesario realizar desagrupaciones*. En general sucede cuando las cifras de la *cantidad 1* son menores que las cifras de la *cantidad 2*.

• *Haga diferencias en las que no es necesario realizar desagrupaciones*. Al igual que en la agregación, el estudiante realiza un tránsito natural para no tener que realizar desagrupaciones.

Y después de realizar un buen proceso con el conteo y la agrupación, es posible pasar a mostrar la suma y la resta aritméticamente, lo cual corresponde al mayor nivel de abstracción que los niños alcanzan para estas dos operaciones básicas.

F) *Suma aritmética*

Este proceso se hace con números; es decir, es necesario que los estudiantes realicen la transición adecuada entre cantidades y su representación simbólica. Y el proceso se lleva a cabo siguiendo el mismo procedimiento que el trabajado en la agregación: se suman los números de cada cantidad empezando por las que se encuentren más a la derecha; si es necesario realizar agrupaciones pueden hacerse, y se sigue así hasta llegar a la cifra ubicada más a la izquierda. Los estadios son los siguientes:

• *Represente en números las cantidades antes de sumar*. Es importante en este estadio realizar descomposición de los números en sus respectivas cantidades; es decir, en centenas, decenas y unidades dependiendo del número.

• *Haga sumas que requieren hacer agrupaciones*. Es importante realizar este proceso antes de cualquier otro, debido a que evita cometer errores futuros, como el de olvidar tener en cuenta la cantidad agrupada.

• *Haga sumas que no requieren realizar agrupaciones*. Debe ser una consecuencia natural a la que lleguen los niños.

G) *Resta aritmética*

Al igual que en la suma, esta operación se hace sobre números, por tanto debe hacerse el tránsito de cantidades a lenguaje simbólico. Y al igual que en la diferencia, se entiende como el proceso de revertir la suma. Y debe tenerse en cuenta también que cada cifra debe ser mayor en la *cantidad 1* que en la *cantidad 2*; de no ser así, debe desagruparse la cifra ubicada en la casilla inmediatamente superior.

Los estadios son los siguientes:

• *Represente cantidades en números antes de restar*. Debe utilizar los símbolos numéricos así como ubicarlos posicionalmente de la manera correcta para poder realizar la resta.

• *Reste realizando desagrupaciones*. Permite evitar errores cometidos con frecuencia en el desarrollo de la resta; como olvidar que se ha descompuesto una cifra.

• *Resta que no requiere de desagrupaciones*. En este caso la resta se puede realizar directamente pues en el paso 2 cada vez que se comprobó si la primera cantidad es mayor o igual a la anterior resulto ser cierto que era así.

5. Matemáticas para la vida

Es un método propuesto para la enseñanza de las matemáticas en una segunda etapa y está basado en el hecho de solucionar situaciones problemas matematizables; el actor principal es el estudiante que se involucra como el personaje principal de la situación que se va a resolver. Las situaciones problema corresponden a un evento real en el que el "matematizador" de la realidad puede escribir en lenguaje matemático la situación que se le plantea, para poderla resolver de forma algorítmica y entregarle a la realidad nuevamente una respuesta y unas implicaciones adecuadas. Lo anterior obliga a que se les dé una utilidad clara a las matemáticas, que sean una herramienta importante para solucionar problemas del entorno físico. En una situación problema se plantea un "vacío de información", que consiste en resolver la pregunta: ¿qué necesito saber para poder resolver mi problema?

Los procesos para resolver una situación problema matematizable se pueden resumir en tres: traducir, formular y desarrollar, y expresar.

A) *Traducir*

Es el proceso por medio del cual el estudiante toma una situación susceptible de ser solucionada matemáticamente y la escriba en términos del lenguaje abstracto, no sin antes evidenciar qui**é**n es el personaje para ponerse en su situación y luego preguntarse qu**é** necesita resolver este. En este momento el estudiante toma la posición del personaje de la situación y se plantea la pregunta: ¿qué me hace falta para solucionar el problema planteado? Este paso se conoce como "definir el vacío de información".

Con esto se está preparado para traducir la situación problema, es decir, definir el objeto matemático.

B) *Formular y desarrollar*

En este paso y teniendo claro el objeto matemático, debe desarrollarse la estrategia, la cual consiste en describir el método para desarrollar el objeto matemático; esto puede ser a través de un flujograma o de un listado de pasos. Y por último, la estrategia exige que se evidencien los insumos con los que se cuenta para resolver el objeto matemático.

Luego de *formulada* la estrategia, debe desarrollarse el objeto matemático, para obtener, efectivamente, el producto matemático. Por lo general, esto es lo que la enseñanza tradicional motiva, ir directamente a resolver el ejercicio sin preocuparse por el contexto ni las implicaciones que este pueda tener en su entorno físico.

Cuando ya se tiene el producto matemático, debe regresarse de nuevo al lenguaje natural; es decir, expresar lo que se desarrolló matemáticamente en la situación real. Este paso se identifica como el proceso de *expresar*.

C) *Expresar*

Una vez obtenido el producto matemático, debe expresarse en términos del "vacío de información"; es decir, ya el matematizador se encuentra en condiciones de llenar su vacío de información. Este proceso se identifica con el nombre de *producto de información*.

Luego de rellenar su vacío de información, el matematizador está en condiciones de entregar sus conclusiones; pero el método y sus autores proponen que se realicen unas *implicaciones*, en un principio guiadas, para que tengan coherencia con la situación pero que también son susceptibles de ser diversas, pues depende de cada personaje y de su forma de interpretar los resultados. Es muy importante en este momento que se intente dar respuesta al propósito vital, pues este le dio el significado para que sea una aplicación a su entorno físico.

El método descrito se puede representar mediante un mentefacto procedimental, conocido como el "método de la U". Cada proceso se encuentra en un espacio diferente, que se representa gráficamente por un rectángulo curvo y se sigue un flujo de procesos lineal.

Veamos a continuación un ejemplo de una situación problema matematizable básica y su representación esquemática.

Don Alberto necesita decidir si la afirmación de Enrique es correcta.

PV
¿Quién es el personaje? ¿Qué necesita para resolver su problema?

IMPLICACIONES

La sugerencia de Enrique es muy buena para construir la piscina.

Don Alberto necesita conocer el área que ocupará la piscina con la medida que propone Enrique.

VI
¿Qué necesita saber para poder resolver su problema?

PI
¿Cuál es la respuesta del VI con base en el PM?

El área que ocupará la piscina es de 9 metros cuadrados.

Potenciación de cantidades absolutas.

OM
¿Cuál es la traducción del vacío de información?

ESTRATEGIA

PM
¿Cuál es la respuesta del OM?

La potenciación de cantidades absolutas es 9

INSUMOS

Medida del lado = 3 metros

MÉTODO

Para calcular la potencia de un número natural.

OM

Potenciación de cantidades absolutas.

Complementando la casa

"Enrique es un obrero, que se especializa en la construcción de piscinas, don Alberto es un amigo personal de Enrique que le pide el favor de construir una piscina en su casa; luego de observar el patio trasero de la casa y tomar unas medidas, Enrique dice que la piscina debería ser cuadrada y tener máximo 3 metros por cada lado, pues él conoce una ley que dice que si se sobrepasan los 10 metros cuadrados de área en una piscina casera, debe pedirse un permiso especial a la curaduría local. Don Alberto es enemigo de los procedimientos largos, por lo que acepta la sugerencia de Enrique, y empiezan a construir[...]".

¿Es correcto que don Alberto acepte de esta manera la sugerencia de Enrique?

La solución y representación esquemática se observa en la página anterior.

Después de observar cómo se solucionan las situaciones problema. Regresamos a los niveles básicos de la suma y resta aritmética, para resolver el equivalente a las situaciones problema dadas en ellos, los cuales son: *transformación aditiva* y *comparación aditiva*.

6. Transformación aditiva

Debe ahora preguntarse por aquellas situaciones en las cuales se aplican las operaciones de adición y diferencia; el proceso de transformación aditiva consiste en presentar situaciones que tienen un estado inicial, sufren una transformación para obtener un estado final. Implícitamente se motiva en el estudiante la noción de incógnita debido a que alguno de los estados puede ser desconocido, los otros dos son datos para resolver el anterior. También es una introducción al concepto de función, lo cual es importante pues el pensamiento algebraico debe motivarse desde los primeros aprendizajes.

La complejidad de las situaciones depende del estado y de su posible aumento o disminución. Para la solución de estas situaciones se utiliza el *algoritmo de resolución de problemas*.

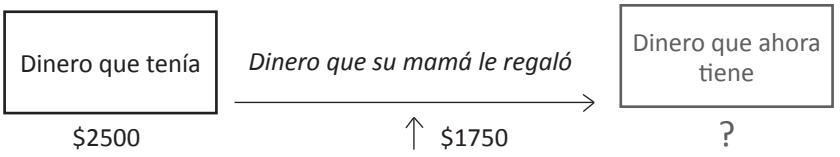
A continuación un ejemplo implementado en niños de segundo de primaria, para entender mejor el proceso.

La situación

Daniela tenía ahorrados $ 2.500 y su mamá le regaló $1.750. ¿Cuánto dinero tiene ahora Daniela?

La solución

1. Traducir



2. Formular_

Como tengo que hallar ***EF (estado final)*** *y la* ***T (transformación)*** *aumenta, entonces tengo que sumar.*

3. Desarrollar

$$\begin{array}{r} \$2.500 \\ +\ \$1.750 \\ \hline \$4.250 \end{array}$$

4. Expresar

Daniela tiene ahora $ 4.250.

Y los estadios de este proceso dependen de lo que se debe encontrar, entonces tenemos los siguientes:

• Solución de situaciones de transformación aditiva donde debe hallarse el estado final.

• Solución de situaciones de transformación aditiva donde debe hallarse la transformación.

• Solución de situaciones de transformación aditiva donde debe hallarse el estado inicial.

La *complejidad* varía dependiendo de las necesidades de los estudiantes y depende del dominio del lenguaje.

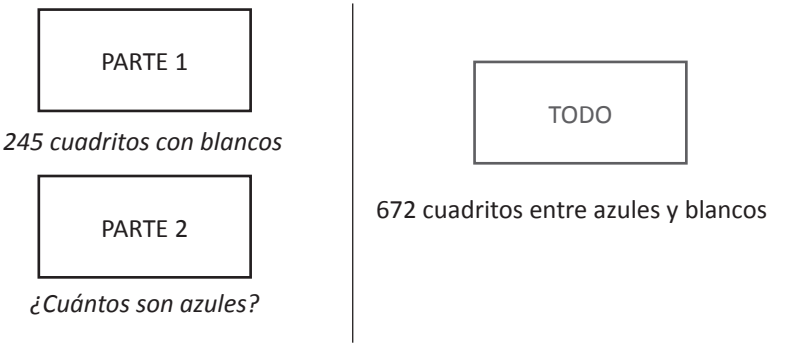
7. Comparación aditiva

Existen también situaciones en las cuales debe compararse dos cantidades, un todo con una parte, bien sea para agregar o para diferenciar. A diferencia de las situaciones anteriores, estas ocurren en un solo instante del tiempo, debe utilizarse también el algoritmo de resolución de problemas. Veamos un ejemplo para entenderlo mucho mejor:

El Chavo ha recortado 672 *cuadritos de papel de colores entre azules y blancos para un trabajo de la escuela. Si* 245 *cuadritos son blancos, ¿cuántos cuadritos son azules?*

Solución

1. Traducir



2. Formular

Como tengo que hallar una de las partes, debo restar al TODO la otra parte.

3. Desarrollar

$$\begin{array}{r} 672 \\ -\ 245 \\ \hline 427 \end{array}$$

4. Expresar

La cantidad de cuadritos azules es de 427.

8. Algunos resultados[3]

Primero se implement**ó** el modelo de matemáticas para la vida en el departamento de Caldas, en estudiantes de bachillerato. En general, los resultados son satisfactorios —expresan los autores— y se ven algunos efectos colaterales como la eficiente solución de preguntas tipo TIMSS. Otro resultado satisfactorio del método y que se ha evidenciado en las

[3] Resultados presentados en la "Ponencia de matemáticas para la vida: una aproximación algorítmica a la resolución de problemas en matemáticas", Carlos Díez - Jorge Luis García - Juan Ramón González.

investigaciones anteriores, es el mejoramiento en la comprensión de lectura de los estudiantes, gracias a que una de las bases para la creación del algoritmo, es el modelo del lector óptimo, estrategia creada e implementada en la Fundación Alberto Merani.

En el Liceo Hermano Miguel de La Salle se viene implementando el método natural desde el 2011 en estudiantes de preescolar, primero y segundo. Se espera para un próximo escrito contar con los datos para evaluar los resultados y poder así evidenciar el gran desempeño que los estudiantes obtienen al trabajar con el método.

Se propone:

• Exponer el método natural para realizar la multiplicación y la división. Que consiste en explorar los ambientes multiplicativos y de división, siguiendo el mismo procedimiento; de lo concreto a lo simbólico siguiendo un proceso natural.

• La investigación que se lleva a cabo actualmente ofrece resultados para expresar la potenciación y la radicación de forma natural; se espera poder presentar el procedimiento.

• Implementar el modelo en estudiantes de primaria. Hoy, algunas instituciones de la ciudad ya implementan el método para la primera infancia.

• Explorar pensamientos como el aleatorio-variacional, que si bien el método trata una variedad de situaciones problema que motivan este pensamiento, no existe una estrategia clara para la didáctica del aprendizaje de tipo aleatorio.

• Se espera también ofrecer unos resultados cuantitativos, para comparar y reafirmar los buenos resultados de una enseñanza basada en estos métodos; por tanto, se han definido unas posibles variables para analizar.

Indicadores

• Resultados académicos de estudiantes con enseñanza tradicional.

• Resultados académicos de estudiantes con enseñanza activa.

• Resultados afectivo-sociales de los estudiantes que aprenden con métodos basados en pedagogías contemporáneas, específicamente con los métodos presentados en este documento.

Se propone también analizar una relación entre los malos desempeños académicos en la educación superior y la falta de formación adecuada en matemáticas desde el inicio del aprendizaje.

BIBLIOGRAFÍA

Boule, F. *Manipular, organizar, representar. Iniciación a las matemáticas*, Madrid, Narcea, S. A., 1995.

Castro, E. y Castro, E. "Primeros conceptos numéricos", en E. Castro (ed.), *Didáctica de la matemática en la educación primaria* (págs. 123-175), Madrid, Síntesis, 2001.

Chamorro, C.; Belmonte, J.; Ruiz, M. y Vecino, F. *Didáctica de las matemáticas para educación preescolar*, Madrid, Pearson, 2005.

Cid, E.; Godino, J. D. y Batanero, C. *Sistemas numéricos y su didáctica para maestros*, Departamento de Didáctica de las Matemáticas, Universidad de Granada, 2003. Recuperable en http://www.ugr.es/~jgodino/edumat-maestros/welcome.htm.

Clements, D. and Sarama, J. *Learning and teaching early math. The learning trajectories approach*, University of Buffalo, State University of New York, Routledge Taylor & Francis Group, 2009.

Curso dictado en 9° Encuentro Colombiano de Matemática Educativa, Valledupar, Colombia.

De Zubiría-Samper, M. *El mito de la inteligencia y los peligros del cociente intelectual*, Bogotá, Fundación Internacional de Pedagogía Conceptual Alberto Merani, 2004.

Dickson, L.; Brown, M. y Gibson, O. *El aprendizaje de las matemáticas*, Madrid, Ministerio de Educación y Ciencia, Labor, 1991.

Diez, Carlos; García, Jorge Luis y González, Juan Ramón. Ponencia de matemáticas para la vida: una aproximación algorítmica a la resolución de problemas en matemáticas, Bogotá 2008.

Diez, Carlos y Pantano, Leonardo. El desarrollo del pensamiento matemático en la primera infancia: método para el aprendizaje natural de las matemáticas, Bogotá, Fundación para el desarrollo educativo y pedagógico EDP, 2012.

Federici, C. *Sobre la resolución de problemas y la numerosidad*, Colombia, Fondo de Publicaciones del Gimnasio Moderno, 2001.

Fernández, K.; Gómez, M.; Gutiérrez, I.; Jaramillo, L. y Orozco, M. "El pensamiento matemático informal de niños en edad preescolar. Creencias y prácticas de docentes en Barranquilla" (Colombia), *Zona Próxima*, Revista del Instituto de Estudios Superiores en Educación, Universidad del Norte, 5, 42-73, 2004.

García-Cruz, J. A. (2005). "La didáctica de las matemáticas: una visión general". Recuperado en marzo de 2009, de http://www.gobiernodecanarias.org/educacion/rtee/didmat.htm.

Gellon, G. *Había una vez el átomo o cómo los científicos imaginan lo invisible*, España, Siglo XXI Editores, 2007.

GODINO, J. D. y BATANERO, C. *Medida y su didáctica para maestros*, Departamento de Didáctica de las Matemáticas, Universidad de Granada, 2003. Recuperable en *http://www.ugr.es/~jgodino/edumat-maestros/welcome.htm*.

GONZÁLEZ, A. y WEINSTEIN, E. *La enseñanza de la matemática en el jardín de infantes: a través de secuencias didácticas*, Rosario, Homo Sapiens Ediciones, 2011.

GRAVEMEIJER, K. and TERWEL, J. "Hans Freudenthal: a mathematician on didactics and curriculum theory", *J. Curriculum Studies*, *32*(6), 777-796, 2000.

IEA (2001). "PIRLS 2001". Recuperado en marzo de 2009, de *http://timss.bc.edu/pirls2001.html*.

OECD. "PISA 2006 results". Recuperado en marzo de 2009, de *http://www.oecd.org/document/2/0,3343,en_32252351_32236191_39718850_1_1_1_1,00.html*.

PUTMAN, R.; LAMPERT, M. and PETERSON, P. "Alternative perspectives on knowing mathematics in elementary schools", *Review of Research in Education*, vol. 16, 57-150, 1990.

REVERAND, E. "Construyendo la aritmética formal a partir de la informal: un estudio de caso", *Revista de Pedagogía*, *25*(72), 7-72, 2004.

Secretaría de Educación Distrital. "Orientaciones curriculares para el campo de pensamiento matemático", *Serie de cuadernos currículo*, Bogotá, 2007.

SERRANO, L. "Elementos geométricos y formas planas", en E. Castro (ed.), *Didáctica de la matemática en la educación primaria* (págs. 379-400), Madrid, Síntesis, 2001.

VERGNAUD, G. *El niño, las matemáticas y la realidad: problemas de la enseñanza de las matemáticas en la escuela primaria*, México, Trillas, 1991.

VILLARROEL, J. "Investigación sobre el conteo infantil", Ikastorratza, *e-Revista de Didáctica*, *171*(4), 1-24, 2009.

WATSON, P. *Ideas: historia intelectual de la humanidad*, Barcelona, Edit. Crítica, 2008.

Capítulo III

ATAQUES A SISTEMAS CRIPTOGRÁFICOS

Isaías David Marín Gaviria*

Resumen

Se presentan los conceptos básicos de la criptología y un estudio a través del tiempo de los principales ataques que pudieron descifrar o romper los sistemas criptográficos más importantes.

Palabras clave y frases: criptología, criptografía, criptoanálisis, criptosistema, cifrar, criptografía simétrica, criptografía asimétrica.

1. Introducción

Desde la Antigüedad, el hombre ha visto la necesidad de compartir información o mensajes sin que otras personas lo conozcan. Se cree que esta práctica data desde el antiguo Egipto, 4.000 a.C. Desde entonces, el hombre ha ideado más y mejores procedimientos para ocultar la información, implementando la mayor parte en el campo político y militar.

La criptografía es utilizada para proyectos gubernamentales en seguridad nacional y por equipos diplomáticos. En la segunda guerra mundial se vio la importancia del criptoanálisis, al ser este el procedimiento empleado para descifrar los códigos del ejército alemán y poderle dar fin a este conflicto bélico [1].

Se presenta un recuento histórico de los principales sistemas criptográficos que se han podido criptoanalizar, rupturas tanto de criptosistemas simétricos como de criptosistemas asimétricos desde la segunda guerra mundial hasta la actualidad.

* Docente investigador de la Corporación Universitaria Republicana. Estudiante de último semestre de Maestría en Ciencias - Matemáticas de la Universidad Nacional.

2. Conceptos básicos de la criptología

A continuación se definen algunos términos y conceptos básicos utilizados en el presente capítulo para una mayor comprensión del mismo.

La *Criptología* es la disciplina que estudia el cifrado y descifrado de la información de una forma segura. Se divide básicamente en criptografía y en criptoanálisis.

La *Criptografía* es el arte de escribir de manera secreta. El origen de la palabra "criptografía" viene del griego *kryptos* que significa 'secreto', y *graphein* que significa 'escribir'. También es el estudio de técnicas matemáticas relacionadas con aspectos de la seguridad de la información tales como la confidencialidad, la integridad de datos, autenticación de la entidad y autenticación del origen de los datos [2].

El *Criptoanálisis* es el estudio de las diferentes técnicas criptográficas para descifrar o decodificar la clave con la cual se cifran los mensajes.

Cifrar es convertir el mensaje original (texto plano) en un texto sin significado aparente llamado "texto cifrado".

Un *criptosistema* L es una quíntupla (P, C, S, e_s, d_s) de conjuntos finitos, donde:

P: textos planos.

C: textos cifrados.

S: claves.

e_s: son funciones de cifrado; $e_s : P \rightarrow C, \forall\, s \in S$

d_s: son funciones de descifrado; $d_s : C \rightarrow P, \forall\, s \in S$

que satisfacen: $d_s(e_s(p)) = p, \forall\, p \in P; \forall\, s \in S$[3].

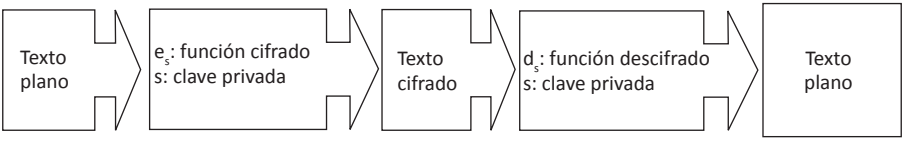
El *cifrado por bloques* es un cifrado donde el texto original se divide en bloques, cada uno de igual longitud, y por último se cifra cada bloque con el sistema criptográfico que se esté usando.

Por ejemplo, si queremos cifrar por bloques el texto "*la clave es verdes*", se elige un tamaño fijo para cada bloque, en este caso elegimos un tamaño de 5 (aunque en la realidad se utilizan tamaños grandes de bloques para evitar que los puedan descifrar fácilmente), y los bloques quedan de la siguiente manera:

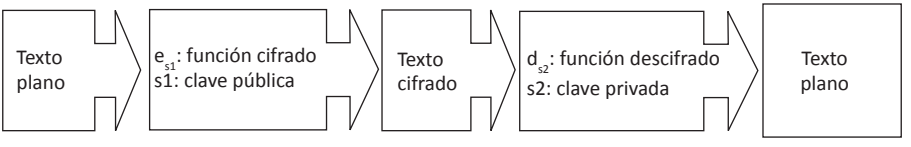
Bloque 1 *lacla*; Bloque 2 *veesv*; Bloque 3 *erdes*,

Luego cada bloque se cifra mediante el algoritmo seleccionado, el tamaño de cada bloque después de cifrado es el mismo.

La *criptografía simétrica* es un procedimiento criptográfico, donde las funciones de cifrado (e_s) y descifrado (d_s) se mantienen públicas. Se utiliza la misma clave para cifrar y descifrar el texto plano y esta única clave se mantiene en secreto entre el emisor y receptor del mensaje cifrado. El siguiente diagrama muestra de manera general como funciona esta criptografía.



La *criptografía asimétrica* es un procedimiento criptográfico, donde la función de cifrado (e_s) y su correspondiente clave de cifrado son públicas, en tanto la función de descifrado (d_s) y su correspondiente clave de descifrado se mantienen en secreto. Este es un procedimiento de dos claves distintas, también llamado "criptografía de clave pública". El siguiente diagrama muestra de manera general como funciona esta criptografía.



Un *ataque* es un intento de conseguir la clave de un sistema criptográfico, o de decodificar el mensaje cifrado. Un ataque se dice efectivo cuando por medio de algún método de criptoanálisis se logra descifrar un mensaje cifrado.

3. Sistemas criptográficos y sus ataques

Este estudio se enfocará en los sistemas criptográficos de una y dos claves, es decir, en los sistemas simétricos y asimétricos.

Tanto para la criptografía simétrica como la asimétrica estudiamos los cifrados más famosos en sus categorías; para la criptografía asimétrica RSA y para la criptografía simétrica DES, TRIPLE DES, AES. En los sistemas simétricos también se incluye la máquina de cifrado electromecánica ENIGMA.

A) *Ataques a sistemas criptográficos simétricos*

Los sistemas simétricos funcionan con una sola clave, y ésta es la misma para cifrar y descifrar la información, aunque es un buen sistema en cuanto a tiempo de cómputo para cifrar y descifrar. La desventaja del sistema es cuando el emisor quiere transferir la clave al receptor; es este punto donde el sistema se hace vulnerable.

Se inicia con la máquina de cifrado más conocida a través de la historia, la usó el equipo de cifrado y descifrado del ejército alemán en la segunda guerra mundial. Este sistema criptográfico consistía en utilizar una máquina llamada *Enigma* [1].

Figura 1. Máquina *Enigma* de cifrado rotatorio [4].

Es durante la segunda guerra mundial que los alemanes utilizaron la máquina *Enigma* para cifrar toda la información en los campos político y militar, por medio de la cual enviaban a sus compañías militares tanto las coordenadas de sus puntos de ataque como los horarios y los regimientos que los iban a perpetrar.

La figura 2 muestra un ejemplo del funcionamiento y cifrado de la máquina *Enigma*, simulado en flash.

Texto original: “Se elige un conjunto de símbolos (para el caso de *Enigma* son los siguientes 26 símbolos: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z) para formar el texto plano, que pueden ser palabras con o sin sentido” **HOLA**.

Algoritmo de cifrado: es la combinación que hacen los rotores (para este ejemplo tres, de los cinco que tenía *Enigma*) de las 26 letras por medio de su cableado.

Texto cifrado: "Se elige un conjunto de símbolos para presentar el texto cifrado (para el caso de *Enigma* son los mismos 26 símbolos) que, por lo general, son palabras sin sentido" **IIBG**.

Clave: "Posición de los rotores en el conjunto de símbolos, es decir, la letra con la que inicia cada rotor". Para este ejemplo: Rotor I **A,** Rotor II **A,** Rotor III **A (A, A, A)**. En total hay $26^3 = 17.576$ claves diferentes.

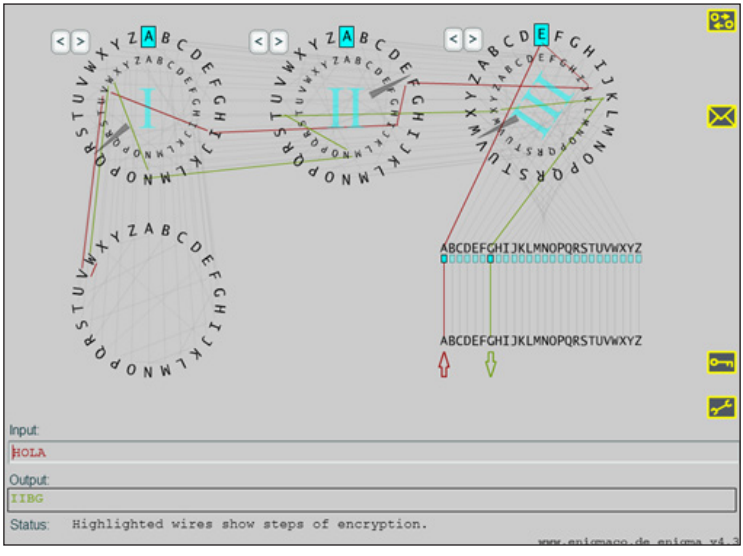


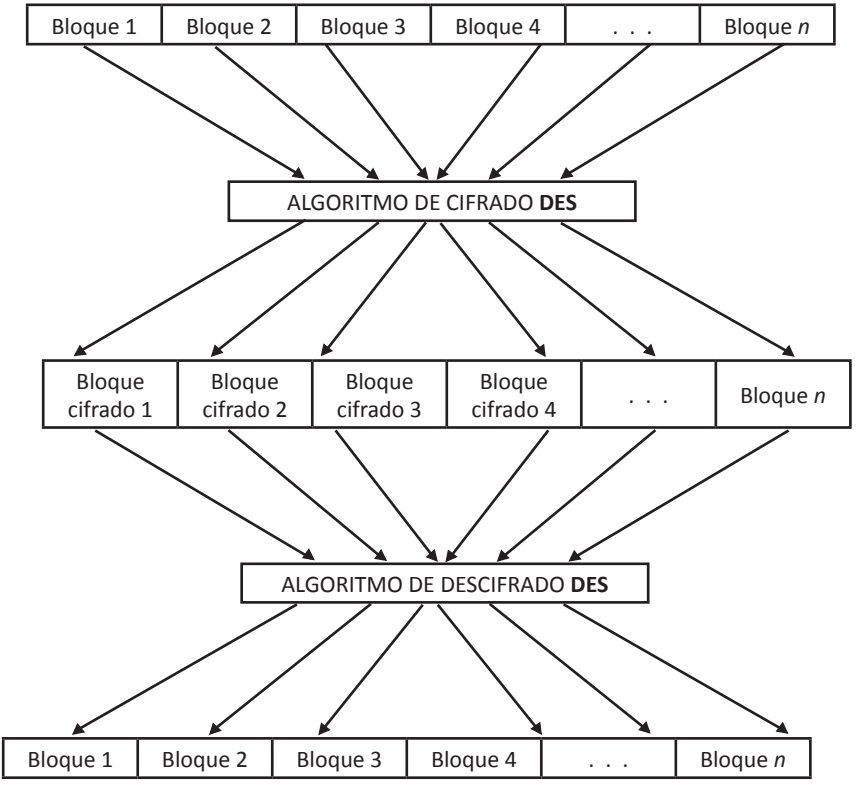
Figura 2. Simulación de cifrado con *Enigma* [5]

La clave de encriptación para este sistema criptográfico simétrico era la ubicación de los rodillos dentro de la máquina, lo cual daba una posibilidad de millones de combinaciones para el mensaje original, por lo que los alemanes creían que esta máquina era indescifrable. Sin embargo, no lo creían así el ejército polaco y sus aliados militares, después de que el ejército polaco interceptara una máquina y tras varios años de intentar descifrarla, Alan Turing y su equipo por parte de los aliados y Marian Rejewski y su equipo a cargo del ejército polaco lograron descifrarla; después de esto se pudieron develar los planes de los alemanes y se logró ponerle fin a ese conflicto bélico.

Desde la segunda guerra mundial con lo hecho por los equipos de Alan Turing y Marian Rejewski, cobra gran importancia el criptoanálisis. Desde entonces es donde se ven los mayores avances en criptografía y en criptoanálisis de su época, aunque con anterioridad ya existía teoría al respecto.

DES (*data encryption standard*) es el más conocido mecanismo criptográfico de la historia, que utiliza una clave secreta o privada de 64 bits, pero en realidad utiliza 56 bits, ya que los restantes 8 bits son para paridad y usa un cifrado de la información por bloques, el cual fue desarrollado por IBM (International Business Machines) en el período 1973-1974. También llamado DEA (*data encription algorithm*), este fue un algoritmo diseñado para implementarse en hardware.

El siguiente diagrama muestra de manera general cómo es el funcionamiento de DES. Se toma el texto plano, se divide en bloques de 64 bits cada uno. Después del cifrado y descifrado con DES el tamaño de los bloques no varía, la clave que se utiliza es de 56 bits.



Tras años de que la autoridad en estándares de Estados Unidos (NBS) abriera un concurso público de un algoritmo para cifrar la información confidencial que cumpliera unas reglas muy rigurosas, DES fue escogido, además se eligió como un estándar para FIPS en los Estados Unidos. Después de esto, el uso de DES se masificó a escala mundial; fue un método estándar para la seguridad del comercio electrónico (tarjetas de crédito) para muchas instituciones financieras de todo el mundo. A continuación se presenta un historial de los reconocimientos obtenidos por DES [6]:

- En 1976 fue aprobado como *estándar federal*.
- En 1977 fue aprobado como *FIPS PUB 46*.
- En 1983 fue confirmado como *estándar*.
- En 1988 fue revisado como *FIPS-46-1*.
- En 1993 fue revisado como *FIPS-46-2*.
- En 1998 fue revisado como *FIPS-46-3* (triple DES).

En 1990, ELI BIHAM y ADI SHAMIR desarrollan un criptoanálisis con el cual, en particular, podían atacar algoritmos criptográficos que cifraran en bloque, y publicaron los resultados obtenidos en un ataque, al cual DES resultó ser resistente. Este nuevo criptoanálisis se llama *criptoanálisis diferencial*, pero DES era resistente a este ataque ya que su desarrollador IBM conoció este criptoanálisis; entonces, en su desarrollo diseñaron a DES resistente a ataques con criptoanálisis diferencial, el mayor interés de este ataque se centra en la parte teórica más que en la práctica [7, 8].

MITSURU MATSUI desarrolla un criptoanálisis para atacar sistemas que cifraran en bloque, llamado *criptoanálisis lineal*. En 1994 se publica la información sobre el ataque a DES con este criptoanálisis, pero resultó con alto costo de tiempo y de equipos descifrar un mensaje cifrado con DES. Este fue un ataque teórico [9].

También en 1994, LANGFORD y HELLMAN proponen una nueva versión de un criptoanálisis combinando los dos anteriores. Este es llamado *criptoanálisis diferencial-lineal*, que combina ambos criptoanálisis en un solo ataque. Cabe notar que estos son ataques en teoría; es decir, son ataques que resultan improbables de aplicar en la realidad.

Finalmente, un sencillo ataque de fuerza bruta, que consiste en probar todas las claves existentes hasta encontrar la correcta y así poder descifrar el mensaje, descifró un mensaje cifrado con DES. En años anteriores no era posible descifrar un mensaje, porque no existía una máquina capaz de realizar todas estas operaciones. Pero en 1998, los avances en microelec-

trónica le permitieron a la Fundación Fronteras Electrónicas (EFF, por sus siglas en inglés), en Estados Unidos, crear la máquina *DES cracking* que rompió un mensaje cifrado con DES tras varios días de cómputo [2, 10].

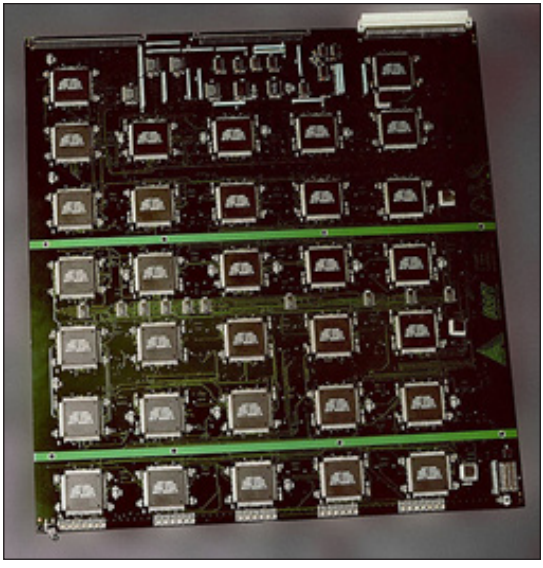


Figura 3. Tarjeta de circuito impresa *DES cracking* que contiene varios chips Deep Crack

En 1999, *DES cracker* de EFF, junto con Distributed.Net, volvieron a descifrar un código cifrado con DES, y lograron hacerlo en menos de un día.

El costo económico de *Des cracking* era muy alto y, por tanto, no era accesible a particulares, pero sí lo era para entidades gubernamentales o entidades privadas con gran capital económico [11].

Con las múltiples rupturas de textos cifrados con DES, era obvio que este algoritmo dejara de ser un sistema seguro desde el punto de vista computacional, con lo que era inevitable pensar en otro sistema de cifrado simétrico más seguro, aunque DES se sigue utilizando en la actualidad.

TRIPLE DES: este nuevo algoritmo de cifrado no es más que por medio de DES encriptar tres veces, pero esto se hace de cuatro maneras diferentes, acotando que la más segura de todas es DES-EEE3:

• DES-EEE3: consiste en encriptar tres veces con DES, pero con tres claves distintas.

• DES-EDE3: consiste en operar tres veces DES de la siguiente manera: encriptar-desencriptar-encriptar, pero con tres claves diferentes.

• DES-EEE2 y DES-EDE2: estos dos métodos son como sus correspondientes métodos anteriores, pero en la primera y la tercera operación utilizan la misma clave.

Como una clave de 64 bits era débil frente a los ataques de fuerza bruta, lo que se buscaba con TRIPLE DES era mejorar la longitud de la clave sin necesidad de cambiar el algoritmo de cifrado y todo esto se logra con el algoritmo de cifrado TRIPLE DES [11].

Como DES dejó de ser seguro ya que se le pueden realizar ataques por fuerza bruta por medio de un hardware en particular, en 1998 Estados Unidos deja de utilizar el algoritmo DES como estándar. Por esta razón, se propone utilizar a TRIPLE DES como estándar hasta que NIST (National Institute of Standards and Technology) seleccione un AES.

AES (*advanced encryption standard*) es el sistema criptográfico más usado hoy día, en cuanto a sistemas simétricos de cifrado en bloque, desarrollado por los belgas JOAN DAEMEN y VINCENT RIJMEN.

Como ya se venía viendo la vulnerabilidad de DES, por sus múltiples rupturas, NIST en 1997 realizó un concurso para un nuevo algoritmo de cifrado simétrico para el siglo XXI, el cual cumpliera con ciertas condiciones, que mejorara las debilidades de DES, entre ellas la longitud de clave, con lo cual se pedía un algoritmo con una longitud de clave de 128, 192 y 256 bits; a este nuevo algoritmo NIST lo llamó AES.

En 1998 se realizó la primera conferencia AES, y es allí donde NIST anuncia los 15 algoritmos admitidos en el concurso (CAST-256; CRYPTON; DEAL; DFC; E2; FROG; HPC; LOKI97; MAGENTA; MARS; RC6; RIJNDAEL; SAFER+; SERPENT; TWOFISH).

En 1999 con la segunda conferencia AES, NIST seleccionó 5 algoritmos finalistas de los 15 que tenían (MARS; RC6; RIJNDAEL; SERPENT; TWOFISH).

Después de someter a todos los algoritmos a pruebas exhaustivas y admitir análisis públicos para estos, se realiza una votación sobre los 5 algoritmos finalistas y en 2000 en la tercera conferencia AES, NIST anuncia al ganador (RIJNDAEL) con 86 votos.

En noviembre de 2001, el algoritmo AES o también conocido como algoritmo RIJNDAEL se publicó como FIPS 197 y, finalmente, en 2002 la agencia NSA decide que AES fuera el reemplazo de DES [12].

Hasta el 2005 se han producido varios ataques teóricos contra AES, pero ninguno de estos ha podido romper este cifrado. En ese mismo año, Daniel J. Bernstein y Adi Shamir, por separado publican ataques efectivos no contra el cifrado de AES, sino que son ataques temporizados de cache. Este tipo de ataques no afecta al algoritmo, sino a una mala implementación de él, ya que va analizando la información encriptada y transmitida por medio de un servidor SSL [13].

B) *Ataques a sistemas criptográficos asimétricos*

RSA (Rivest, Shamir y Adleman) es el primero y más utilizado sistema criptográfico de dos claves, o de clave pública desarrollado por Ronald Rivest, Adi Shamir y Leonard Adleman en 1977. La seguridad de RSA se basa en la dificultad de factorizar un número entero grande [14].

El coautor del sistema RSA, Ronald Rivest, para probar la robustez de RSA propuso en 1977 que se factorizara un número entero de 129 dígitos. Este problema se le llamó RSA-129. Fue hasta 1994 que Merced desarrolló un método llamado QS (*quadratic sieve*) que le permitió factorizar un número entero de 126 dígitos.

RSA-129 = 11438 16257 57888 86766 92357 79976 14661 20102 1829672124 23625 62561 84293 57069 35245 73389 78305 9712356395 87050 58989 07514 75992 90026 87954 3541 = 34905 29510 84765 09491 47849 61990 38981 33417 76463 84933 87843 99082 0577 x 32769 13299 32667 09549 96198 81908 3446141317 76429 67992 94253 97982 88533 [15, 16].

En 1991, los laboratorios RSA publicaron nuevos retos para hallar números RSA; es decir, semiprimos, números con exactamente dos factores primos, que motivaron nuevos desarrollos en cuanto a la factorización de números enteros, además de premios en dinero para las personas o entidades que pudieran factorizarlos.

En 1991 RSA-100 como el entero de menor longitud propuesto que fue factorizado.

- En 1992 se factorizó RSA-100
- En 1992 se factorizó RSA-110
- En 1993 se factorizó RSA-120
- En 1996 se factorizó RSA-130
- En 1999 se factorizó RSA-140 y RSA-155
- En 2003 se factorizó RSA-160

- En 2004 se factorizó RSA-150
- En 2005 se factorizó RSA-200 y RSA-640
- En 2009 se factorizó RSA-170 y RSA-768
- En 2010 se factorizó RSA-180 y RSA-190
- En 2012 se factorizó RSA-704

Los laboratorios RSA decidieron en 2007, retirar los premios para los números RSA que faltaban por factorizar propuestos en sus desafíos [17, 18].

4. Conclusiones

El avance de la tecnología desde las máquinas electromecánicas hasta las máquinas con nanotecnología permitieron el avance en sistemas tanto criptográficos como criptoanalíticos.

Avances en matemáticas en teoría de números permitieron el criptoanálisis de sistemas como RSA-129 y RSA-130.

Dado el acelerado avance en el tiempo de cómputo de los ordenadores de la actualidad, DES resulta ser inseguro por su pequeña longitud de clave, aunque sigue siendo utilizado hoy día; ya es un sistema obsoleto.

AES es un algoritmo seguro después de los ataques de cache que ha sufrido.

Después de que los ordenadores pueden factorizar enteros cada vez de mayor longitud RSA sigue siendo seguro, ya que puede ir ampliando la longitud de sus claves.

REFERENCIAS

[1] D. Kahn. *The Codebreakers*, New York, The MacMillan Company, 1996.

[2] A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of applied cryptography*, CRC Press, Boca Ratón, FL, 1997.

[3] W. Willems y I. Gutiérrez. *Una introducción a la criptografía de clave pública*, Bogotá, Edic. UniNorte, 2008.

[4] *http://www.de1939a1945.com/imagenes/at_enigmacilindros.jpg*.

[5] *http://enigmaco.de/enigma/enigma.swf*.

[6] R. Durán, L. Hernández y J. Muñoz. "Ataques a DES y módulos factorizados", en línea 5 de diciembre de 2012, disponible en *http://revistasic.com/revista40/agorarevista_40.htm*.

[7] E. Biham and A. Shamir. "Differential cryptanalysis of DES-like cryptosystems", *Journal of Cryptology*, 4, 3-72, 1991.

[8] S. Landau. "Standing the test of time: The data encryption standard", *Notices of the AMS*, 47, 341-349, 2000.

[9] M. Matsui. *Linear cryptanalysis method for DES cipher. Advances in cryptology*, Eurocrypt '93 Proc., 386-397, Springer-Verlag, 1994.

[10] Electronic Frontier Foundation. *Cracking DES: Secrets of encryption research. Wiretap politics, and chip design*, O'Reilly & Associates, 1998.

[11] B. Schenier. *Applied cryptography*, New York, John Willey & Sons Inc., 1996.

[12] Computer Security Objects Register (CSOR): *http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf*.

[13] D. Arne, A. Shamir, and E. Tromer. *Cache attacks and countermeasures: The case of AES revised*, 2005.

[14] R. Rivest, A. Shamir, and L. Adleman. *A method for obtaining digital signatures and public-key cryptosystems*, Comm. acm 21, 120-126, 1978.

[15] R. Rivest. "RSA-129-Challenge", *Scientific American*, August, 1977.

[16] D. Atkins, M. Graff, A. K. Lenstra, and P. C. Leyland. *The magic words are squeamish ossifrage*, Proc. Asiacrypt '94, lncs 917, 263-277, New York, Springer Verlag, 1995.

[17] R. M. Elkenbracht-Huizing, J. Buchmann, J. Loho, and J. Zayer. *An implementation of the number field sieve*, Exp. Math. 5, 231-252, 1996.

[18] A. K. Lenstra, H. W. Lenstra, M. S. Manasse, and J. M. Pollard. *The number field sieve*, 11-42, New York, 1990.

Capítulo IV

BASES EPISTEMOLÓGICAS DE LOS LENGUAJES DE LAS MÁQUINAS LÓGICAS

Magdalena Pradilla Rueda*

Resumen

El planteamiento de las *máquinas l*ógicas por los matemáticos y lógicos de los años treinta, como solución al requerimiento de procedimientos de cálculo efectivo, mecánico, tienen como base de *representación* el cerebro, el pensamiento o las facultades de pensar o calcular. Esta representación necesita de una formalización y simbolización, que se realiza por medio de *formas de representación*, cuya prioridad está dada por los *lenguajes.*

Estos *lenguajes* han seguido una importante evolución desde las incipientes formas de lenguaje del inicio hasta los lenguajes sofisticados de las máquinas físicas actuales. En general, sus características se enfocan en la precisión, no ambigüedad, no polivalencia y la referencialidad, lo que conlleva buscar sus bases epistemológicas en los aportes realizados en el nacimiento de la lógica matemática.

Palabras clave: lenguajes formales, lógica matemática, epistemología de la informática.

1. Introducción

Hablar de *lenguaje,* en general, conlleva preguntarse sobre los procesos del *pensamiento.* Hablar de *lenguaje*, en particular, de las *máquinas lógicas*, conduce a preguntarse sobre la posibilidad del pensar

* Doctor en Informática y Matemáticas Aplicadas a Ciencias Sociales, Universidad de Grenoble (Francia), 1983. Tesis: *Búsqueda de descriptores en indexación automática*; Doctor en Filosofía, Universidad Paris 1 Panthéon-Sorbonne, 2008. Tesis: *Hacia una epistemología de la teoría informática*. Actualmente, investigadora en el Centro de Investigaciones de la Corporación Universitaria Republicana.

de estas máquinas. Quisiéramos, en primer lugar, evocar los momentos centrales del nacimiento de la pregunta sobre esta posibilidad, su entorno y los elementos que la contextualizan, es entonces la problemática de la representación que está en el centro de la reflexión.

En sus inicios se plantea como soporte o modelo de representación el organismo (cerebro, función de pensar,…), donde la pregunta se traslada a las *formas de* representación, tanto en planteamientos como los de A. Turing o S. Kleene. Las propuestas de formalización y simbolización de estos primeros años, señalan a los *lenguajes* como una forma prioritaria de representación. El análisis de estas propuestas presenta reflexiones sobre las relaciones entre los elementos de la representación: *objeto de representación* y *lo representado*, que permiten situarnos en el núcleo de los *lenguajes formales* y sus *bases lógicas* y *epistemológicas*, que es el del nacimiento de la lógica matemática de finales del siglo xix.

2. Representación de las máquinas lógicas

Es conocido que las ciencias en general y la matemática en particular son creadoras de formas, de manera que en este último caso, se tienen formas de diversas clases: estructuras, modelos, fórmulas, lenguajes, diagramas, esquemas e igualmente las formas geométricas, como figuras y representaciones gráficas. Así, el quehacer matemático es, en primera instancia, una "puesta en forma", lo que conlleva a la representación, simbolización, formalización, modelización..., en donde la matemática es esencialmente morfología o ciencia de las formas. En segunda instancia, este quehacer se refiere al cálculo que puede ser numérico, algebraico, formal… Esto nos lleva a decir que la producción matemática se puede presentar como cálculos, algoritmos o como creación de formas.

En este sentido, lo que llama nuestra atención para la argumentación, es la puesta en forma a partir de la representación. Así, al hablar de las *máquinas lógicas*, nos estamos refiriendo a máquinas abstractas que no tienen todavía un elemento físico o material y que en sí mismas, en algunos casos, son la representación de un modelo físico, llamado cerebro o red nerviosa.

El núcleo de esta *representación*, lo podemos observar al situarnos en el contexto de los ejes de la teorización de las *máquinas lógicas* o *abstractas*[1]. Este contexto podría pensarse, según M. L. Minsky (1967),

[1] Se llaman "máquinas lógicas" porque responden a las modelizaciones de los procedimientos de cálculo, sin tener en cuenta su parte material o física.

en su obra *Computation: Finite and infinite machines*, como si estuviéramos "sumergidos en una nueva revolución tecnológica concernida por la mecanización de los procesos intelectuales".

Esta mecanización de procesos intelectuales conducen a preguntarse sobre la manera de pensar y cómo se piensa, e igualmente sobre la manera de calcular y cómo se calcula y, a su vez, ¿cómo se representan formalmente estos procesos? Vamos a responder a estos interrogantes, aproximándonos, por un lado, a la problemática planteada por ALAN TURING, sobre los diferentes elementos, funciones y aplicaciones que ellos pueden realizar y sus representaciones, primero en su estudio de 1936 (A. TURING, 1937), y luego con la pregunta sobre las "máquinas que piensan", planteada en 1950 (A. TURING, 1950, pág. 266)[2]. Por otro lado, nos aproximamos a la problemática sobre la representación de autómatas, cuyo punto de referencia es un artículo de KLEENE, "Representation of events in nerve nets and finitude automata" (S. KLEENE, 1956) y sus revisiones posteriores.

A) *Propuestas de TURING*

TURING va a presentarnos una máquina lógica, la *máquina de TURING*, como modelo de la función de calcular, en el artículo de 1936, con su formalización y simbolización requeridas. Su objetivo es la realización de cálculos caracterizando lo que él llama las funciones y números calculables (J. LASSEGUE, 1998, págs. 18 y 19)[3]. Él presenta la existencia de esta máquina que imita una facultad particular del espíritu y que constituye entonces el modelo de cálculo en este caso particular (A. TURING,

[2] "Pienso, sin embargo, que al final del siglo (XX), el uso de las palabras y la opinión general educada habrán evolucionado tanto que se podrá hablar de las máquinas pensantes sin que nadie contradiga esto".

[3] *Calculable* es el resultado de una operación que conduce a la determinación exacta de un número. Una función se llama *calculable* si su valor para cada número calculable en el conjunto de partida es un número calculable. Una de las tareas capitales de la investigación matemática, consiste en encontrar el medio de aproximar por medio del cálculo un cierto número de funciones; esto es lo que se llama el *análisis numérico*: se trata de encontrar métodos algorítmicos que permitan hallar los elementos característicos que hacen posible el cálculo aproximado de la función estudiada. Por ejemplo, la función $\sqrt{x}$ definida sobre el conjunto de los números reales que a toda x en el conjunto de los números naturales haga corresponder la raíz $\sqrt{x}$, descrita a cualquier nivel de aproximación decidido con anterioridad, es una *función calculable* porque siempre es posible exhibir el resultado único de la puesta en correspondencia entre x y $\sqrt{x}$.

1937, págs. 51, 77-84) del espíritu[4]. El modelo de cálculo de las máquinas de TURING es puramente determinista: él describe *estados discretos*. Estos son definidos como acciones o funciones determinadas y finitas en un período de tiempo, identificados al funcionamiento de una persona que calcula y actúa siguiendo los "estados del espíritu". Cada estado determina el siguiente estado, de manera que es necesario finalizar un estado para seguir con el otro, es decir, un desarrollo de forma secuencial.

La máquina se aplica por medio de un programa finito (control finito), que puede manipular una lista lineal de casillas, llamada banda, por una "cabeza" de acceso (cursor) y cuyo objeto es el desarrollo de cualquier cálculo; con ello, TURING quiere caracterizar los números calculables.

En este sentido, los componentes previstos de la *máquina lógica* son muy simples:

• una banda, cuyo lado izquierdo es limitado pero el lado derecho es infinito. La banda está dividida en casillas de la misma talla; las funciones pueden realizarse sobre la banda y son: escribir, leer y memorizar las etapas intermedias de un cálculo.

• un conjunto finito de símbolos, por ejemplo: 0, 1 (números); *s* (separar expresiones); *d* (principio de la banda); *f* (fin del cálculo).

• una cabeza de escritura y de lectura, que se desplaza sobre la banda.

• un conjunto finito de estados: estos estados permiten distinguir varios comportamientos posibles (el principio, la parada, la lectura, la supresión, la adición,…, etc.)

• un conjunto finito de instrucciones.

TURING va más allá de este caso particular de su máquina y generaliza su argumento: su máquina podría imitar todas las máquinas. Así, él plantea la existencia de una *máquina de TURING universal* capaz de imitar, de reproducir, de simular el comportamiento de cualquier otra máquina de TURING. Es necesario, entonces, escribir sobre la banda un símbolo (conjunto de marcas) que codifica una tabla de funcionamiento de la máquina que se va a imitar. Esta máquina universal constituye el "modelo de los modelos" y, en consecuencia, es el modelo del "espíritu",

[4] "Un hombre que calcula el valor de un número real puede compararse a una máquina susceptible de encontrar un número finito de estados $q1$, $q2$, …, qr, que se pueden llamar *m*- configuraciones" […] "el comportamiento de este hombre que calcula está determinado a cada instante por los símbolos que observa y por su «estado de espíritu» de cada momento".

si se define el espíritu como el modelo de la facultad de modelizar. En este caso, modelizar es simular o reproducir en la máquina el funcionamiento de la persona que calcula como una actividad de conocimiento. Es decir, que con esta máquina se está representando la capacidad de calcular de cualquier persona.

Con esta *máquina universal*, TURING caracteriza lo que se llamaría una *función recursiva parcial universal*.

Por otro lado, TURING, con la pregunta: "¿Las máquinas pueden pensar?", en 1950, propone reemplazar esta pregunta por otro problema, el del "juego de la imitación" (A. TURING, 1950, pág. 265). Este se juega a tres, un hombre, una mujer y uno que interroga. Este debe hacer preguntas bajo condiciones bien precisas, de manera que se pueda distinguir el hombre de la mujer. La nueva pregunta de TURING es: "¿Qué pasa si una máquina toma el lugar del hombre en el juego?". A lo que responde: el que interroga (máquina en este caso) se equivoca igualmente que cuando el juego se desarrolla entre un hombre y una mujer. Si una máquina puede imitar el que interroga, entonces, se puede hablar de *máquinas que piensan*. TURING va más lejos en su experimento, y examina de cerca la posibilidad de que las máquinas aprendan y concluye así:

> "[...] se puede esperar que las máquinas competirán finalmente con el hombre en los campos puramente intelectuales"[5].

Así se puede plantear la *máquina* lógica como modelo que piensa. Pensar es para TURING "imitar"; aquí, él presenta el pensamiento de las máquinas en términos epistemológicos, porque el pensamiento no puede tomar cuerpo, sino al interior del campo del conocimiento.

> "Si el pensamiento no se identifica sino en y por sus productos, hay que abandonar la idea que uno pueda aprehenderlo *a priori*, habría que aceptar concebirlo como una constatación, como el fruto de una inferencia[...] El pensamiento es actualmente presentado como generador de lógicas, no como encerrado en una de ellas[...], la cibernética de los años cuarenta adopta como paradigma la *máquina de TURING*, como medida de cualquier característica que determina el control del pensamiento por él mismo"[6].

[5] De esta manera, en los años 1960, se encuentran los promotores de la *inteligencia artificial*, a partir de las propuestas de TURING.

[6] *Sciences cognitives. Textes fondateurs* (*1943-1950*), págs. xxxi-xxxii.

B) *Planteamientos de Kleene y sus revisiones*

a) Teoremas de Kleene

La base de nuestro planteamiento son dos de los teoremas presentados por Kleene, en el artículo anunciado y sus revisiones realizadas por los comentaristas de este[7]. Estos dos teoremas consideran dos maneras de concebir una máquina finita: una compuesta (según sus componentes) y la otra global (general). En el primer teorema, Kleene considera un organismo descrito por sus componentes y en el segundo, él define un autómata[8] a partir de un número finito de estados globales (funciones o comportamientos).

Su argumentación se basa en la representación de un organismo y un autómata que recibe un estímulo y que efectúa ciertas acciones; algunas de ellas son consideradas como respuestas a un estímulo, si este se produce. Estas acciones son representadas por un estado del organismo o del autómata, y el problema está en definir qué tipo de estímulo o de evento[9] puede ser representado por el estado del organismo o del autómata.

Particularmente, Kleene se interesa en los eventos que surgen en las redes nerviosas y en los autómatas finitos. Él plantea este tipo de redes como ilustración de la teoría general de autómatas, incluyendo robots, computadores u otros. En el primer caso, introduce la noción de *evento representado* por una red nerviosa y en el segundo, *evento representado* por un autómata finito.

En la primera parte, Kleene retoma las redes nerviosas introducidas por McCulloch y Pitts en 1943[10]. Estas redes están constituidas

[7] Vamos a retomar la argumentación de Evelyn Barbin, en su artículo: "Les deux faces des théorèmes de Kleene et la question des machines", en *Calculs et formes*: Ouvrage collectif coordonné par Jacqueline Boniface, Paris, Ellipses Editions Marketing, 2003.

[8] *Autómata*, entendido como una máquina de estados finitos.

[9] *Evento*, entendido particularmente como una propiedad de la entrada a una red nerviosa o a un autómata finito y, generalmente, como lo que puede representar un organismo o un autómata.

[10] W. S. McCulloch, "Une comparaison entre les machines à calculer et le cerveau", en "Les machines a penser", Colloque 1951, cnrs, Paris, 1953, págs. 425-443: "Los relés del sistema nervioso se llaman neuronas. Se trata de células vivas que presentan un cierto número de propiedades con la capacidad de graduaciones finitas; una de estas propiedades es crucial en lo que concierne a la transmisión de los signos, la que transmite «todo o nada». Esta propiedad es la que posibilita la elaboración de un cálculo que se puede aplicar a todas las redes de relés, utilizando así las respuestas de «todo o nada»[...]".

por un número finito de neuronas ligadas entre ellas. Existen dos clases de neuronas: aquellas que tienen una función de entrada y aquellas que tienen una función interna (que conforman los componentes del organismo). Cada neurona tiene dos terminaciones, *estimulada* o *inhibida*, y en cada instante, está encendida o apagada[11]. KLEENE formaliza esta noción asociando la neurona a una fórmula lógica, él presenta $P(t)$ para la neurona P encendida en el tiempo t, utilizando símbolos lógicos; así los componentes de una red de neuronas pueden ser una red de conjunción, de disyunción o de iteracción.

Un *evento* se define por un período de tiempo fijo y puede ser expresado por una fórmula booleana y representa una red nerviosa si su aparición en el momento p provoca el encendido de una cierta neurona interna en el momento $p + 1$. La representación de la noción se realiza bajo la forma de tablas y de gráficos para los componentes iteractivos.

En la segunda parte, para definir la noción de autómata finito, KLEENE considera el tiempo como sucesión de *momentos discretos*. Un autómata se construye con un número finito de *células* que son células de entrada o internas, y que tienen también dos estados posibles, 0 y 1. El estado de cada célula interna en el tiempo t ($t > 1$), se determina por los estados de las otras células en el tiempo $t - 1$, porque los razonamientos se hacen sobre el conjunto global de los estados de las células del autómata finito. KLEENE define un estado completo del autómata como constituido de los k estados de las células de entrada $N_1 \ldots N_k$ (estado externo completo) y de m estados de las células internas $M_l \ldots M_m$ (estado interno completo).

KLEENE introduce la noción de *eventos regulares* como aquellos representados por los autómatas finitos. Esta noción de regularidad se funda en el comportamiento global del autómata finito. De la misma manera, él señala el aspecto finito del número de células y de estados completos que intervienen en su argumentación.

Con este tipo de representación propuesta por KLEENE, se pueden considerar varios aspectos:

• La diferencia entre la representación de las redes neuronales y los autómatas: presenta una primera formalización basada en las neuronas simples para ir hacia las redes de neuronas en general. Con la representación basada en los autómatas, él hace un salto hacia una máquina, es

[11] Más precisamente, una neurona se alumbra al tiempo t si al menos un cierto número de terminaciones inhibidas vienen de neuronas alumbradas al tiempo $t - 1$.

decir, hacia una formalización mecánica. Se encuentra aquí un primer esbozo de la relación entre un organismo vivo y un autómata.

• La definición de *evento* como el objeto de la representación, que bajo este ángulo particular es una *propiedad* de la entrada de una red nerviosa o de un autómata y por generalización, el *evento* representa el funcionamiento general de la red o del autómata. De esta manera, este *evento* se aproxima de aquello que se puede llamar un *procedimiento de cálculo efectivo*, con sus características de temporalidad (período de tiempo), su formalización y expresión por medio de una fórmula lógica booleana. El período de tiempo es fijo en el caso de un organismo y, en caso del autómata, el tiempo es una sucesión de momentos discretos que imparten una regularidad sobre el comportamiento global del autómata finito.

• Así mismo, KLEENE subraya el aspecto finito del número de células, de estados y estados completos que intervienen en su argumentación.

Esto implica que a partir del *evento*, se pueda pensar en una forma de analogía entre los organismos y los autómatas, porque los dos tienen como origen de su representación la actividad de una neurona y se representan por el *evento* y su definición temporal.

• Los diferentes medios o herramientas utilizados para representar el organismo y el autómata, pueden clasificarse en dos tipos: las formas simbólicas (gráficos y tablas) y las formas numéricas a partir de las formas binarias.

• La definición de componentes de los organismos o autómatas, entre los cuales se tienen las neuronas o células que aseguran las funciones sugeridas por KLEENE: es el punto de partida *físico* o *material* que realizan las funciones de estímulo e inhibición de la célula.

Estos aspectos de la representación planteados por KLEENE, nos dan algunos elementos primarios sobre el funcionamiento de una *máquina lógica*, como el procedimiento dinámico que incluye la temporalidad y representa la máquina y el organismo; las diferentes formas de representación, para las dos entidades: las simbólicas (gráficos y tablas) y las numéricas (binarias); y un esbozo de componentes físicos, como las células que realizan las funciones de los organismos.

b) Teoremas de KLEENE revisados (1956-1960)

Un grupo de lógicos y matemáticos posteriormente, entre 1956 y 1960, retoman los teoremas de KLEENE, simplificándolos, formalizándolos

y dándoles nuevas perspectivas para el estudio de una máquina o de un autómata (E. Barbin, 2003). Ellos precisan los elementos formales de las máquinas, así:

• Copi, Elgot y Wright (1958, págs. 181-196) utilizan las redes lógicas de Burks y Wright, quienes habían mostrado que la lógica simbólica con dos valores puede ser empleada para caracterizar el comportamiento de los circuitos de calculadores digitales electrónicos, impartiéndoles un fundamento formal y riguroso. A su vez, Copi, Elgot y Wright representan por medio de gráficos, estos comportamientos para tres estados: la entrada, la salida y el estado completo, en períodos consecutivos de tiempo. Al contrario de Kleene, ellos distinguen entre un *evento* y sus expresiones, para lo cual definen un lenguaje, su alfabeto y sus fórmulas definidas por inducción, así mismo que los *procedimientos efectivos* para encontrar las expresiones correspondientes a una red. Igualmente, se representan por medio de gráficos los elementos de las redes (flechas y círculos marcados con operadores que facilitan la lectura de una red).

Vemos aquí la importancia de la independencia de la máquina y sus "expresiones"; es decir, éstas como los medios o formas de representación simbólica de lenguaje con un alfabeto bien definido y los gráficos, que pueden semejarse a los posteriores lenguajes de programación y organigramas. Nos encontramos, entonces, con la simbolización y formalización de las formas o herramientas de representación.

• Moore (1956, págs. 129-153) distingue dos clases de experiencias de naturaleza conceptual: aquella que introduce en la máquina secuencial una cadena de símbolos de entrada y aquella que recibe una cadena de símbolos de salida. La máquina secuencial está concernida por cadenas de símbolos, su comportamiento es estrictamente determinista, en el sentido que el estado actual depende únicamente de la entrada y estado precedente, y la salida actual depende de este estado. Así, una entrada es una simple cadena de 0 y 1, pensada sin referencia al tiempo; tanto para Moore como para Kleene, una entrada es un evento (o propiedad) que se produce en un período de tiempo, representado por una tabla dimensionada para este período. Moore considera las máquinas como cajas negras descritas en términos de entradas y salidas, sin que ninguna construcción interna se conozca.

Así mismo, Moore representa las máquinas por medio de tablas y gráficos (*diagramas de transición de entradas y salidas*).

Vemos aquí una simplificación de la máquina (*caja negra*), lo mismo que la simbolización y formalización de las formas de representación,

como son los gráficos y las tablas, así que las entradas y las salidas como *cadenas de 0 y 1*.

• M. O. Rabin y D. Scott (1959, págs. 114-125) dan una versión algebraica en el artículo "Los autómatas finitos y sus problemas de decisión". La estructura interna del autómata (máquina) debe ser determinada por los estados estables en tiempos discretos. El conjunto de estados "definidos" por un autómata está formalizado completamente y matematizado algebraicamente. Ellos consideran las máquinas con un número finito de estados, que permite realizar un procedimiento efectivo finito para comparar dos autómatas, permitiendo de esta manera, determinar su equivalencia.

• R. McNaughton y U. Yamada (1960, págs. 39-47), en el estudio llamado: "Expresiones regulares y grafo de estados para autómatas", traducen el *grafo de estados* que describe un autómata, en un lenguaje específico del autómata. Este lenguaje presenta una descripción formalizada similar a la escritura tradicional de izquierda a derecha, la cual facilita su utilización, sin muchos cálculos o reflexiones excesivas. Este enfoque tiene un parecido a la introducción posterior de los *lenguajes de programación*, más fáciles para el usuario que los lenguajes de máquina (en 0 y 1), y de aquella de los organigramas que representan gráficamente los programas del usuario.

En su desarrollo, McNaughton y Yamada identifican *objetos* con descripciones: así, un autómata no se representa aquí por un grafo, él es descrito por un grafo, y una expresión no expresa lo que hace un autómata, sino lo que describe un autómata. Se trata, por tanto, de encontrar procedimientos que permitan traducir una descripción gráfica en una descripción en lenguaje, y viceversa. Ellos se preocupan más por las "expresiones" de la representación de las máquinas que por la representación misma, de ahí la importancia dada a las formas de representación: los lenguajes y los grafos, es decir, la simbolización y la formalización.

A partir de los presupuestos de la representación de funciones de calcular o pensar en las máquinas lógicas, tanto por Turing como posteriormente por Kleene y sus comentaristas, vemos un esbozo de los diferentes elementos que representan estas máquinas, tanto en los procedimientos de cálculo que incluyen la temporalidad, las formas de representación simbólica (lenguajes, gráficos y tablas) y las formas de representación numérica (binarias). De la misma manera, se abre la posibilidad de realizar

las máquinas físicas, con el planteamiento de componentes elementales, físicos, de los organismos (células) y de los autómatas.

Lo que llama la atención, en estos presupuestos, es la importancia dada a la puesta en forma de la representación; es decir, de las formas de representación basadas en los lenguajes y los procedimientos, lo que amplía la manera de realizar cálculos, no solo de manera numérica (que aquí es binaria), sino también simbólica; por lo cual, el lenguaje va a tomar el puesto central en el desarrollo de estas máquinas lógicas, convertidas luego en máquinas físicas.

Igualmente, estos presupuestos nos conducen a presentar desde el punto de vista epistemológico, reflexiones suscitadas entre la aproximación cerebro-máquina y la importancia dada al lenguaje en esta aproximación, por lógicos, neurofisiólogos y matemáticos, de esos momentos.

3. Analogías epistemológicas cerebro-máquinas-lenguajes

A) *Relaciones cerebro-máquinas*

La representación de una red nerviosa, que recibe un estímulo y que efectúa acciones, representada en un *evento* o estado de un organismo o máquina (autómata), o igualmente, la representación por medio de una *máquina lógica* de Turing, de funciones realizadas por una persona que calcula, nos plantea una problemática presentada por la relación "organismo-autómata" o de una forma más general, "cuerpo-máquina".

Vemos, por ejemplo, cómo en el caso de la representación de una red de neuronas, estamos concernidos por preguntas localizadas en el ámbito del cerebro específicamente, es decir, que el cuerpo es cerebro fisiológico; al contrario, si la representación contiene el "funcionamiento del cuerpo" en general, estamos concernidos por los "estados de espíritu" de la persona que calcula (Turing), y si se trata de representar el "pensamiento", estamos concernidos por una particularidad de estos "estados de espíritu"[12].

De esta manera, "los fisiologistas" especialistas del cerebro que se sirven de redes electrónicas semejantes a las redes nerviosas, se preguntan si el cerebro funciona como una máquina que calcula (C. E. Shannon y

[12] Estas son definiciones que corresponden específicamente a las necesidades de los desarrollos teóricos de los lógicos y neurofisiólogos, pero que muestran la dificultad que existe en definir y delimitar rigurosamente los conceptos como el cerebro, el espíritu, el pensamiento y el cuerpo, en general.

J. McCarthy, 1956)[13]: estamos en el caso de la máquina como "modelo del cerebro". Así mismo, si las máquinas pueden ejecutar ciertas funciones que se asemejan a las funciones realizadas por una persona que calcula, la máquina puede ser un "modelo de la función de calcular". Si las capacidades de las máquinas que calculan son cada vez más grandes, nos podemos preguntar si una máquina puede imitar el cerebro en el momento que se piensa, en este caso, la máquina es entonces el modelo de un "cerebro que piensa".

En este sentido, se tienen principalmente las investigaciones de McCulloch y Pitts (1943, págs. 115-133), que conciernen la estructura interna del cerebro y la máquina y quieren asimilar el cerebro a una máquina lógica, que puede ser considerada, tanto en su estructura como en su comportamiento, como una idealización de la anatomía y de la fisiología del cerebro, dotada de todas las facultades atribuidas al espíritu. Esto constituye un avance definitivo, porque no es solo el cerebro en su estructura (cerebro material), pero también el cerebro en su función (el espíritu), que se puede asimilar a un mecanismo lógico.

El núcleo epistemológico de estos planteamientos, se enfoca en el estatuto de una máquina, en donde las máquinas "hechas a mano" no son cerebros, pero los cerebros son una variedad mal entendida de *máquinas lógicas*. Por esto, al relacionar el mundo neurofisiológico y aquel del espíritu, no se trata aquí de humanizar la máquina pero de mecanizar lo humano en su conjunto (J. P. Dupuy, 1994, pág. 41).

B) *Relación máquina-lenguaje*

Así mismo, sabiendo que la representación del cerebro, la función de calcular o la función de pensar, concierne prioritariamente las formas de representación simbólica, como el lenguaje; lógicos como Putnam, van a referirse notablemente a este tipo de representación.

Él se sitúa primero en la problemática cerebro-espíritu, para luego referirse a la problemática cuerpo-máquina y concluye con la relación *lenguaje-máquina*. Quiere mostrar que las preguntas sobre este tema son completamente de carácter del lenguaje y de la lógica. En su estudio "Minds and machines" de 1960 (Putnam, 1960, págs. 110-134), él señala la posibilidad o no de identificar los eventos mentales y los eventos físicos. Putnam pretende mostrar que es posible un equivalente lógico. Él lo

[13] Ellos nos indican que el sistema nervioso frecuentemente comparado con un intercambiador telefónico o máquina de cálculo dirige los datos sensoriales y motrices.

examina, con relación a una máquina de TURING generalizada, considerada a la vez como una máquina lógica (constituida de estados) y como una máquina material (compuesta de circuitos, lámparas,…).

PUTNAM, en su argumentación va a asimilar los estados lógicos de la máquina lógica a los estados mentales del humano, y los estados estructurales de la máquina material a los estados físicos del humano (identidad cuerpo-máquina). La identidad cuerpo-espíritu, que necesita identificar los estados mentales con los estados físicos, puede identificar, a su vez, los estados lógicos con los estados estructurales o materiales de la máquina; los cuales, según la argumentación de PUTNAM, no se pueden identificar, y frases como "el estado mental ψ es idéntico al estado cerebral ϕ" no son verdaderas y no pueden ser utilizadas actualmente para expresar una identificación teórica, porque ninguna identificación semejante se ha realizado. Para él, es necesario utilizar "precondiciones" para esta identificación teórica.

Así, si el problema cuerpo-espíritu es análogo al problema de la relación entre los estados mentales y lógicos, pero no idénticos, la solución presentada por PUTNAM es una analogía entre el lenguaje máquina y el lenguaje del humano.

4. LENGUAJES COMO FORMAS DE REPRESENTACIÓN

Siguiendo los lineamientos anteriores sobre la representación de las máquinas lógicas, podemos plantear que las formas de representación simbólica por naturaleza son los *lenguajes*.

Se ha visto que las instrucciones del procedimiento de cálculo, que representan las configuraciones de una máquina de TURING, son claras, precisas, sencillas y sin ambigüedades, pues ellas van a mecanizarse. Para KLEENE, tanto los eventos o procedimientos, las tablas y los grafismos son igualmente precisos; y, además, en la mayoría de los comentaristas de KLEENE vemos que tienen una preocupación, sobre la puesta en forma de sus descripciones, procedimientos y cálculos, donde en algunos se presenta claramente los elementos del lenguaje. Igualmente, a partir de los desarrollos ulteriores de los autómatas, se crea una independencia entre la máquina y la forma de representarla, y la importancia y desarrollo gira hacia las formas de representación en sí mismas y no en la representación de la máquina.

La pregunta que nos compete, en este caso, sería: ¿Cómo definir los lenguajes aceptados por las máquinas lógicas? En ese sentido,

describir los procedimientos de cálculo en lenguaje ordinario, sin rigor particular, corre el riesgo de presentar interpretaciones que dependen de los observadores, y por tanto, las instrucciones pueden ser entendibles de diferentes maneras, es decir, como expresiones multivalentes[14]. Por otro lado, desde la tradición de TURING, sabemos que el desarrollo de las formas de representación de lenguaje requiere la relación del cálculo con el procedimiento de cálculo efectivo; esto es, no solo desde el punto de vista de la posibilidad de realizar el cálculo, sino también de las condiciones prácticas de su realización. Es decir, que el lenguaje requerido es un lenguaje formalizado (*lenguaje formal*) para que sea admisible por una *máquina lógica*.

A) *Lenguajes formales*

Estos lenguajes están en la línea de la lógica matemática; su estructuración tiene como base un conjunto de símbolos, por un lado, y un conjunto de reglas de formación y de transformación, por otro. La función de las reglas es la manipulación de los símbolos independientemente de cualquier contenido, durante una temporalidad discrecional del procedimiento de cálculo y no de una temporalidad contingente de las cosas de la realidad.

De esta manera, las reglas permiten reconocer la estructura del lenguaje, definiendo:

• Las cadenas de palabras que son expresiones bien definidas o correctas del lenguaje, lo que se llama la *sintaxis* del lenguaje.

• Las cadenas de palabras que son expresiones significativas del lenguaje, lo que se llama la *semántica* del lenguaje.

• Los símbolos del lenguaje son de diferentes tipos:

- Alfabeto: se trata del conjunto finito de símbolos de base bien definidos, compuestos de letras, números, valores, operadores, signos de puntuación[...], etc. Sobre este alfabeto, por medio de las reglas de transformación, se constituyen las expresiones bien definidas.

[14] GUILLAUME WATIER, "Le calcul confié aux machines", Paris, Ellipses Editions Marketing: "Para no correr ese riesgo, CHURCH y TURING proponen formular las instrucciones en un lenguaje específico que fuera entendido por una máquina-intérprete. ¡Esto necesitaba que tal máquina existiera!, lo cual requería contar con las herramientas mínimas (el lenguaje con reglas y la máquina intérprete), que pudiera escribir cualquier procedimiento. En ese sentido, CHURCH desarrolló un formalismo matemático llamado «lambda-calcul» y, al mismo tiempo, TURING inventó su *máquina de TURING*", pág. 29.

- Datos: corresponden a diferentes tipos: entero, real, complejo, cadenas, objeto.

- Constantes y variables.

- Expresiones definidas.

En general, se pueden presentar las características de la dinámica de este lenguaje como:

• *denotacional* o *referencial*: cada signo designa, denota o hace referencia a un "objeto" dentro de la memoria de la máquina lógica. En el caso de la *máquina de Turing*, el signo en la casilla de la banda que es modificado o borrado.

• *inferencial*: se refiere a la existencia de reglas de inferencia aplicadas a los signos del alfabeto que producen conclusiones que pueden ser interpretadas.

• *verificacional*: se refiere a la verificación de las expresiones o instrucciones del lenguaje como expresiones bien definidas y verdaderas.

5. Bases lógicas y teóricas de los lenguajes

Sabemos que el desarrollo de las *máquinas lógicas* y el esbozo de sus lenguajes se desarrollan hacia 1936, pero las bases fundacionales de estos lenguajes se presentan desde finales del siglo XIX, con el renacer de la *lógica matemática* que en principio se encarga, de manera privilegiada, de la problemática relativa a los *fundamentos de matemáticas*, encargada de fomentar las matemáticas sobre un soporte seguro[15]. En este sentido, se encuentran dos líneas prioritarias: la representada por Frege y Russell, conocidos como *logicistas*, porque fundamentan las matemáticas en la lógica y consideran la *lógica* como *lenguaje*. La línea formalista que compite con los logicistas, el *programa de Hilbert*, que representa la primera forma histórica de la *teoría de la demostración.*

A) *Lógica como lenguaje*

Según esta línea, el *universo*, al cual se refieren los enunciados y, por consecuencia, aquel en el cual estos enunciados pueden ser verificados como *verdaderos* o *falsos*, es único y no sería otro que el *mundo real*, la totalidad de aquello que es (incluye las entidades lógico-matemáticas).

[15] Robert Blanché y Jacques Dubucs (1996).

De aquí, la creencia en la unicidad y unidad del lenguaje. Como consecuencia de esta tesis, se excluyen las consideraciones metalingüísticas o metateóricas en lógica; no se puede salir del universo de la lógica para dar explicaciones de las propiedades en otro lenguaje, porque la lógica es lo verdadero y el fundamento último del enunciado que se cuestiona.

Esta tradición se consolida, en primera instancia, con GOTTLOB FREGE (1848-1925), en la *Begriffsschrift* (1884) (G. FREGE, 1999). Con esta obra se libera la lógica de las matemáticas, pero al mismo tiempo prepara una interrelación profunda entre las dos ciencias. Él sienta las bases de la semántica lógica y su preocupación es la ciencia, en la cual la lógica es prioritaria. Pretende un ideal matemático científico, poseuclidiano, en el que se expliciten los principios propiamente matemáticos, que dan a la ciencia su contenido, pero también los principios lógicos que aseguran su estructura formal.

FREGE tiene una concepción verdaderamente moderna de la lógica (cuantificación, axiomatización). Pretende que las nociones lógicas fundamenten e introduzcan las nociones matemáticas (por ejemplo: el *número* es definido en términos de relaciones biunívocas entre los elementos de dos clases y estima que los axiomas de la aritmética pueden ser extraídos de las leyes lógicas fundamentales).

De esta manera, propone una *lengua formularia* o *lengua perfecta*, caracterizada por la precisión de los términos que tienen como objeto la definición explícita y el empleo unívoco de signos; este mismo determinado por reglas y operaciones explícitas y a partir de estas las proposiciones pueden ser derivadas. La *lengua perfecta* es artificial y simbólica: la *ideografía*, que es, a la vez, cálculo para la aritmética y una *lengua característica* o expresión de un pensamiento con símbolos, se compromete así con la formalización. FREGE rechaza el análisis del contenido conceptual de las fórmulas y no retiene sino las relaciones lógicas que existen entre sus elementos. Con este desarrollo, él buscaba quitar cualquier laguna (*lü**ckenlos*) en la cadena de deducción, sabiendo que el gran obstáculo para alcanzar su objetivo es la inadecuación del lenguaje.

Él sabe que muchas de las nociones lógicas sobre las cuales se apoya, exigen ser analizadas y va a hacer precisiones importantes para la adecuación del lenguaje, de donde se tienen tres artículos de 1891 y 1892: *Función y concepto, Concepto y objeto* y *Sinn und Bedetung* (*Sentido y referencia*). En este último, se interroga sobre la verdadera connotación de la relación de *identidad*: si *A* es idéntica a *B*: ($A = B$), ¿de qué estamos hablando? El signo representa dos aspectos: lo que significa o lo referido

(objeto extralingüístico), lo cual presenta una aporía en la identidad, y él va a incluir, entonces, el elemento del *sinn* (*sentido*) que va a aclarar la relación de identidad.

Frege se enfoca, igualmente, hacia la noción de un cálculo lógico, por medio de la ideografía, la cual debe asegurar también la verificación de la secuencia de los enunciados que respondan a las reglas de la lógica deductiva. Él se interesa por el componente lógico del pensamiento, solo la verdad importa al lógico y, en el orden del pensamiento formal, esta se sustenta sobre la validez de una demostración.

Los herederos directos de Frege son B. Russell y Ludwig Wittgenstein. Russell (1872-1970) emprende su programa logicista con su obra: *Principa mathematica, 1910-1913*, donde sus análisis sobrepasan el dominio único de las matemáticas y de la lógica, interesándose en la ciencia (a partir de la teoría nomológica). Esta obra se tiene como la *Biblia* de la nueva lógica. Posteriormente va a desarrollar dos de las teorías importantes en lógica contemporánea y el desarrollo de lenguajes: la *teoría de tipos* y la *teoría de las descripciones definidas*. En la *teoría de tipos*, Russell va a hacer uno de los aportes más significativos para el desarrollo de los lenguajes, que soluciona, en parte, el problema de la autorreferencia, la reflexibilidad y la indecibilidad. Así, los tipos presentan dos aspectos: a) para hablar o definir los objetos extralingüísticos, es necesario un *lenguaje de tipo cero*, que es el *lenguaje referencial*; b) para hablar o definir este *lenguaje de tipo cero*, es necesario un *lenguaje* de tipo 1, tipo 2, tipo 3[...], etc. Con este planteamiento, no se podrían incluir en un conjunto *E* el nombre que permite designar este conjunto y el criterio o término que lo identifica, de forma que un nombre que designa los objetos, no sea utilizado para designarse él mismo como parte de estos objetos. De esta manera, se diferencia el conjunto *E* del término que permite identificar este conjunto, con el fin de no perder la distinción entre niveles lógicos.

Por su lado, Ludwig Wittgenstein (1889-1951) pretende elaborar una *teoría del lenguaje a priori* y se pregunta sobre las condiciones de posibilidad y validez de un lenguaje. Él presenta los límites del lenguaje al marcar las fronteras de lo *decible* y lo *indecible*, lo que determina, a su vez, los límites del *sentido* y *sin sentido* en un lenguaje estructurado correctamente. Su obra, *Tractatus logicus philosophicus* (Wittgenstein, 1993), contiene los conceptos previos para el planteamiento de un lenguaje *ideal*. Las características exigibles para este lenguaje son las siguientes: a) *explícito*: provisto de una notación material donde el sentido

de la *proposición* es visiblemente expresada y transparente; b) *unívoco*: sin ambigüedad, ni polivalente que pueden llevar a la confusión y la indecisión: a cada nombre corresponde un objeto; c) *funcionable*: es el complemento operativo de la exigencia precedente: a todo signo corresponde una función; d) *distinción de niveles lógicos*: debe ser claro si se habla de lo real (de los hechos) o un nivel del lenguaje (proposiciones, nombres,…). Esta precisión del lenguaje evita la confusión de los tipos o categorías. Es necesario distinguir el nombre propio de los predicados: el 'nombre propio' designa un objeto (referencia extralingüística) que pertenece al lenguaje de los objetos (realidad extralingüística). El término 'predicado' describe una clase de objetos y pertenece a un nivel conceptual. Lo que designa un predicado no es un objeto extralingüístico, sino una construcción lingüística (clase).

WITTGENSTEIN va a plantear tres ejes complementarios de estructuración del lenguaje: *proposicional*, *analítico* y *representacional*. Con estos ejes, él presenta dos perspectivas del lenguaje: por un lado, la base para la realización de las expresiones del lenguaje (proposicional y analítico), y por otro, la base para la creación de la relación entre el lenguaje y lo real (representacional).

Así mismo, desarrolla con el grupo de Cambridge, la concepción del *atomismo l*ógico, visto como una tentativa de *concepción del mundo* que va a dar luz a los sistemas lógicos, precisando la diferencia entre aquello que es dado y aquello que se infiere. En este sentido, aclara la definición del objeto de conocimiento, ya que este pertenece a los hechos en sí mismos y el hecho se expresa por medio de la *proposición,* a la cual se le puede conferir su estatuto de *veracidad*.

B) *Programa de Hilbert: teoría de la demostración*

En la perspectiva fundacionista, HILBERT presenta la necesidad de sentar las bases de las teorías matemáticas en una prueba de *coherencia*: se trata de probar que el uso regular de las inferencias lógicas no podían conducir, a partir de axiomas, a demostrar, a la vez, un enunciado y su negación. Él da una prueba *sintáctica*, directa de la imposibilidad de deducir de estos axiomas un enunciado y su negación, a la vez; y no una prueba *semántica*, en la cual se va a demostrar que los axiomas de la aritmética son simultáneamente válidos en algunas interpretaciones. En este objetivo, se hace abstracción de toda *significación* atribuida a las fórmulas de la aritmética, considerándolas como concatenaciones de símbolos en las cuales solo importan las características *morfológicas*. Es

así que se ve la extrema importancia que da HILBERT al *problema de la decisión* (*Entscheidungsproblem*); es decir, a determinar mecánicamente si una fórmula es o no es un teorema de un sistema formal dado, como característica fundamental de la *teoría de la demostración*.

De esta manera, si se presentaron desarrollos positivos con respecto a la teoría, igualmente se presentaron otros negativos. En el sentido negativo, se encuentran los resultados conocidos de TURING-CHURCH (1936): no hay procedimientos mecánicos de decisión ni para la lógica de primer orden, ni para la aritmética entera; GÖDEL y su prueba de la *incompletud*, en la cual exhibe un enunciado verdadero pero no demostrable en la aritmética de PEANO, supuesta coherente y con esta prueba se hace distinción entre la verdad y la prueba; así mismo, que la prueba negativa de POST. Hacia los años 1920, HILBERT reconoce que la prueba de *coherencia* que busca no es *absoluta* en sentido estricto.

Desde el punto de vista práctico, para el desarrollo de su teoría, va a proponer: la formalización rigurosa de un sistema deductivo, de signos que son vacíos de cualquier contenido y de un conjunto de axiomas que sirven de base al sistema deductivo; la definición del lenguaje *metamatemático* que no pertenece al sistema, pero sus aserciones se refieren a los signos que figuran en el sistema; y la utilización de procedimientos finitistas de cálculo.

6. CONCLUSIONES

Es claro que la representación de las máquinas lógicas en sus comienzos, se inicia con el modelo de la función de calcular y pensar, o cerebro, redes neuronales u organismo en general; es decir, que hay una representación directa con el objeto representado, se representa directamente y sus formas de representación se basan en lenguajes incipientes pero con lineamientos formales, cuyas bases fundacionales se presentan en el renacer de la lógica matemática, ligados primordialmente sobre el fundamento referencial y verificacionista (con referencia al mundo de los hechos). El desarrollo posterior de los lenguajes de computación, dejan su característica primaria de representación (objeto de representación), para centrarse sobre su estructuración y abstracción propia, creando lenguajes mucho más pertinentes para los requerimientos del desarrollo de la parte física, material y electromagnética de las máquinas de calcular o de computación producidas posteriormente. Es decir, que han seguido un desarrollo sin seguir el rigor de una representación de un hecho real (ce-

rebro, por ejemplo) y, por el contrario, su centro de interés está situado en una gran "tecnicidad", indispensable para su inteligibilidad y pertinencia.

Sin embargo, el desarrollo de los lenguajes requiere una ampliación continua de las formalizaciones, precisiones y abstracciones, y surgen nociones que complementan las características de las máquinas lógicas, actualmente físicas, tales como la complejidad, información, evaluación, aleatoriedad, lo que hace que se investiguen nuevas formas de representación, de lenguajes y de modelos de cálculo. En consecuencia, el centro de representación gira una vez más sobre elementos de la naturaleza, como la biología, la física y el cerebro. En este caso, las formas de representación y los modelos de cálculo que resultan son mucho más sofisticados, abstractos, formalizados, de acuerdo con los nuevos objetos de representación y condiciones técnicas.

BIBLIOGRAFÍA

Blanché, Robert et Dubucs, Jacques. *La logique et son histoire*, Paris, Armand Colin (Collection "U", série Philosophie), 1996.

Barbin, Evelyn. "Les deux faces des théorèmes de Kleene et la question des machines", en *Calculs et formes*, Ouvrage collectif coordonné par Jacqueline Boniface, Paris, Ellipses Éditions Marketing, 2003, págs. 24-51.

Copi, I. M.; Elgot, C. C. and Wright, J. B. "Realization of events by logical nets", *Journal of the Association for Computing Machinery*, vol. 5, no. 2, April, 1958.

Dupuy, Jean Pierre. *Aux origines des sciences cognitives*, Paris, Éditions La Découverte, 1994.

Frege, G. *Idéographie*, traduction, préface, notes et index par Corine Besson. Postface de J. Barnes, Paris, Librarie Philosophique J. Vrin, 1999.

Kleene, S. C. "Representation of events en nerve nets and finitude automata", en *Automata studies*, éd. C. E. Shannon et McCarthy, Princeton, Princeton University Press, 1956.

Lassegue, Jean. *Turing*, Paris, Éd. Les Belles Lettres, 1998.

McCulloch, Warren S. "Une comparaison entre les machines a calculer et le cerveau", en *Les machines a penser*, Colloque 1951. cnrs, Paris, 1953.

McCulloch, Warren S. et Pitts, Walter. "A logical calculus of the ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics*, vol. 5, 1943.

McNaughton, R. and Yamada, H. "Regular expressions and state graphs for automata", *Transactions of the ire Professional Group on Electronic Computers*, vol. EC-9, no. 1, Mars 1960.

Minsky, M. L. *Computation: Finite and infinite machines,* Prentice Hall, Englewood Cliffs, New Jersey, 1967.

MOORE, E. F. "Gedanken-experiments on sequential machines", dans *Automate studies*, éd. C. E. Shannon et McCarthy, Princeton, Princeton University Press, 1956.

PUTNAM, HILARY. *Minds and machines*, Dimensions of mind: A symposium, Sydney Hook, ed., New York, University Press, New York, 1960. Trad. française, *Pensée et machine*, éd. A. R. Anderson, Seyssel, Éditions du Champ Vallon, 1983.

RABIN, M. O. and SCOTT, D. "Finite automata and their decision problems", IBM, *Journal of Research and Development*, vol. 3, no. 2, April 1959.

SHANNON, C. E. and MCCARTHY, J. *Automata studies*, Princeton, Princeton University Press, 1956.

TURING, ALAN M. "On computable numbers, with application to the Entscheidungsproblem", publicado en *Proceedings of the London Mathematical Society*, 1937. *Théorie des nombres calculables, suivie d'une application au problème de la décision*, traduit de l'anglais et annoté par Julien Basch, dans *La machine de Turing*, Paris, Editions du Seuil, 1995.

— "Computing machinery and intelligence", *Mind*, vol. 59, no. 236, 1950. Trad. française, dans *Sciences cognitives. Textes fondateurs (1943-1950)*, Paris, PUF, 1995.

WATIER, GUILLAUME. *Le calcul confié aux machines*, Paris, Ellipses Editions Marketing, s/f.

WHITEHEAD, ALFRED NORTH and RUSSELL, BERTRAND. *Principia mathematica*, Cambridge, Cambridge U. P., vol. 1, 1910; vol. 2, 1912; vol. 3, 1913.

WITTGENSTEIN, LUDWIG. *Tractatus logico-philosophicus*, trad. de G. G. Granger, Gallimard, 1993. *Tractatus logico-philosophicus, suivi de investigations philosophiques*, trad. de l'allemand par Pierre Klossowski, Paris, Gallimard, 1961.

NOTAS

NOTAS

NOTAS

NOTAS

Este libro se terminó de imprimir en los talleres de Nomos Impresores, el día nueve de febrero de dos mil catorce, aniversario del nacimiento de Alfredo Vásquez Carrizosa
(n. 9, ii, 1973 y m. 19, xii, 2001).

LABORE ET CONSTANTIA