Universidad Autónoma del Estado de Hidalgo

PRÁCTICA 0

ALUMNO: Ian Alejandro Jaen Cruz
Dr. Eduardo Cornejo-Velázquez

1. Introducción

En esta práctica se exploran los conceptos fundamentales de los lenguajes formales y los autómatas, herramientas esenciales en la teoría de la computación. Se abordarán temas como alfabetos, palabras, operaciones con lenguajes, autómatas finitos deterministas y no deterministas, y sus aplicaciones en la computación.

2. Marco teórico

Lenguajes formales: palabras, alfabeto y clausura de Kleene

Un **alfabeto** es un conjunto finito de símbolos (ej: {0, 1}). Una **palabra** es una secuencia finita de símbolos de un alfabeto (ej: "010"). Un **lenguaje formal** es un conjunto de palabras sobre un alfabeto. La **clausura de Kleene** (*) es el conjunto de todas las palabras posibles (incluyendo la palabra vacía) que se pueden formar con un alfabeto .

Operaciones con palabras y lenguajes formales

Las operaciones básicas incluyen:

- **Concatenación**: Unión de dos palabras o lenguajes (ej: "ab" + "ba" = "abba").
- **Unión**: Combinación de todos los elementos de dos lenguajes (L1 L2).
- **Intersección**: Elementos comunes entre dos lenguajes (L1 L2).
- **Diferencia**: Elementos de un lenguaje que no están en otro (L1 - L2).
- **Complemento**: Todas las palabras no incluidas en un lenguaje respecto a *.

Operaciones con lenguajes y aplicaciones

Otras operaciones importantes son:

- **Estrella de Kleene (L*)**: Repetición de un lenguaje cero o más veces.
- **Potencia de un lenguaje (L)**: Concatenación de un lenguaje consigo mismo n veces.
- **Inversa de un lenguaje**: Palabras escritas al revés.
- **Aplicaciones**: Compiladores, procesamiento de lenguajes naturales, verificación de software, etc.

Autómatas: el corazón de la computación

Un **autómata** es un modelo matemático que procesa cadenas de símbolos y decide si pertenecen a un lenguaje. Su función principal es reconocer o generar lenguajes formales.

Autómata finito determinista (AFD)

Un **AFD** es un autómata donde para cada estado y símbolo de entrada, hay exactamente un estado siguiente. No permite ambigüedades y es predecible.

Autómata finito no determinista (AFND)

Un **AFND** es un autómata que puede tener múltiples transiciones para un mismo símbolo desde un estado. Permite ambigüedades y puede tener transiciones vacías ().

Convertir un AFND a un AFD

Para convertir un **AFND** a un **AFD**, se construye un AFD donde cada estado representa un conjunto de estados del AFND. Este proceso se conoce como el algoritmo de construcción de subconjuntos.

Autómata con transiciones epsilon ()

Un **autómata con transiciones epsilon** es un AFND que permite transiciones sin consumir ningún símbolo de entrada (-transiciones). Estas transiciones simplifican la representación de ciertos lenguajes.

Convertir un AFND con transiciones lambda () a un AFND

Para eliminar las -transiciones, se calcula la clausura- de cada estado, resultando en un AFND equivalente sin transiciones vacías.

Pattern matching con autómatas

El **pattern matching** utiliza autómatas para buscar patrones en cadenas de texto. Sus aplicaciones incluyen búsqueda de subcadenas, expresiones regulares y filtrado de datos.

Clases de equivalencia de autómatas y lenguajes formales

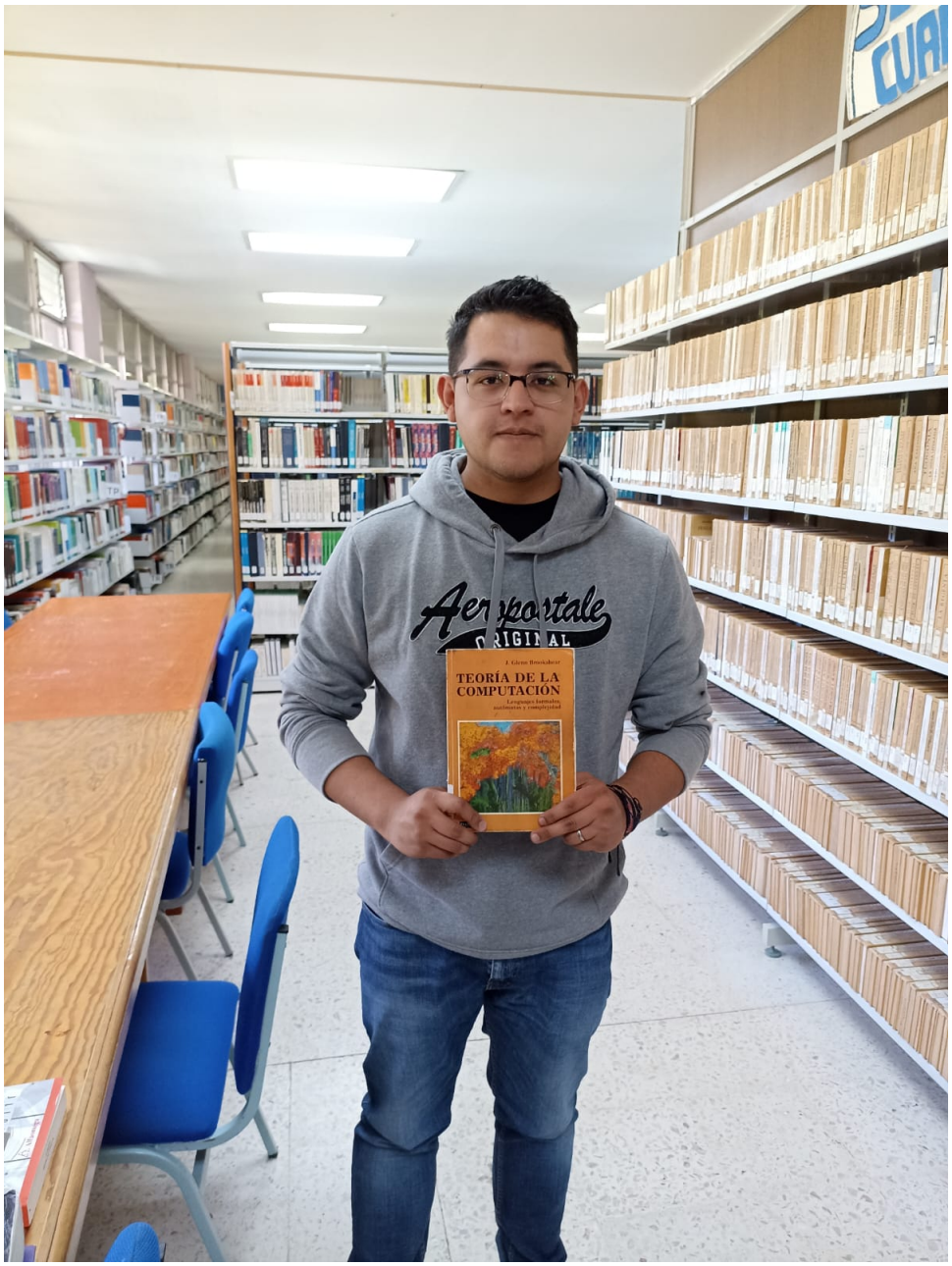
Las **clases de equivalencia** son conjuntos de autómatas que reconocen el mismo lenguaje. Cada clase corresponde a un lenguaje formal específico. La **minimización de autómatas** reduce un autómata a su forma más simple manteniendo su funcionalidad.

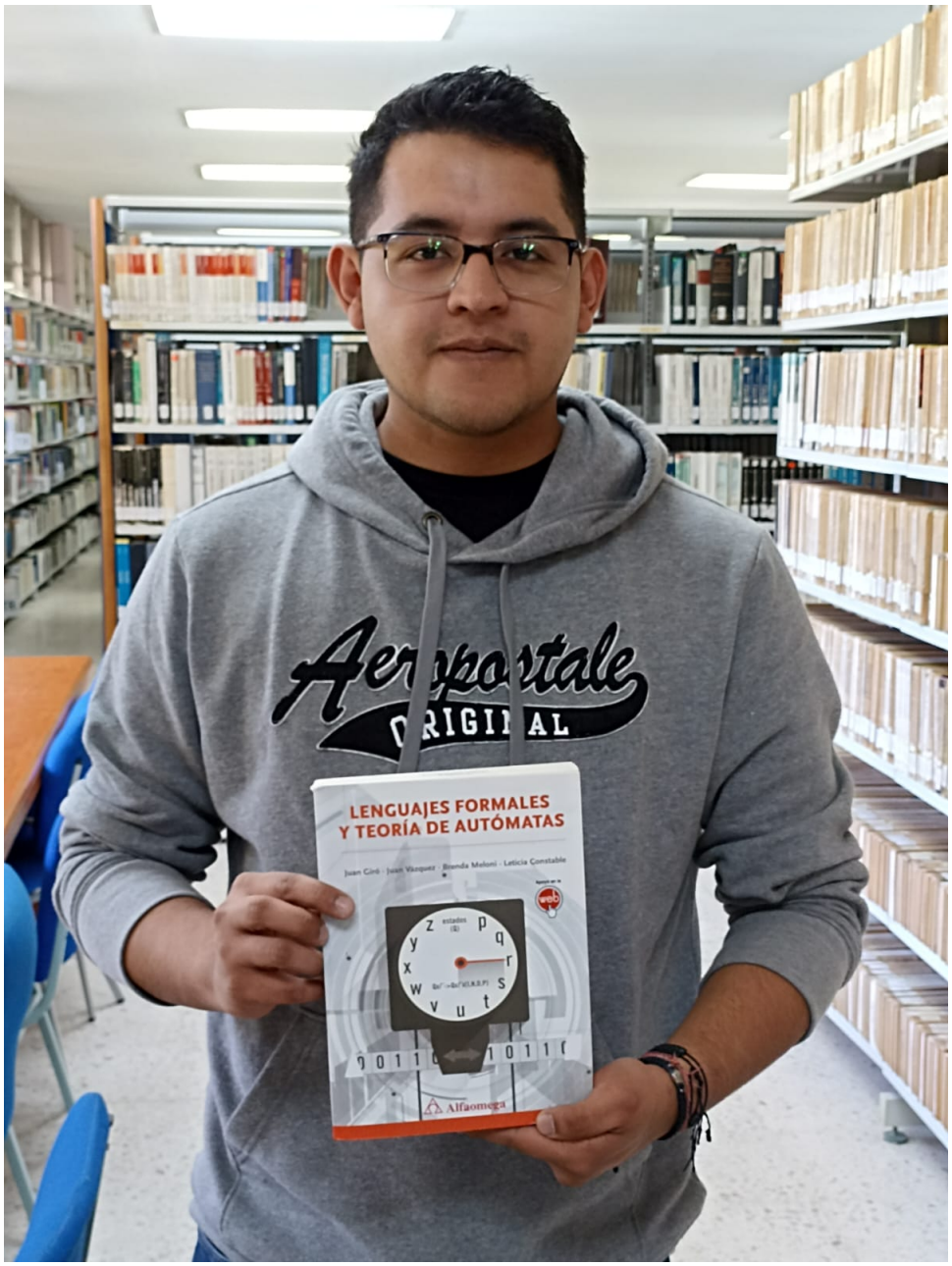
3. Herramientas empleadas

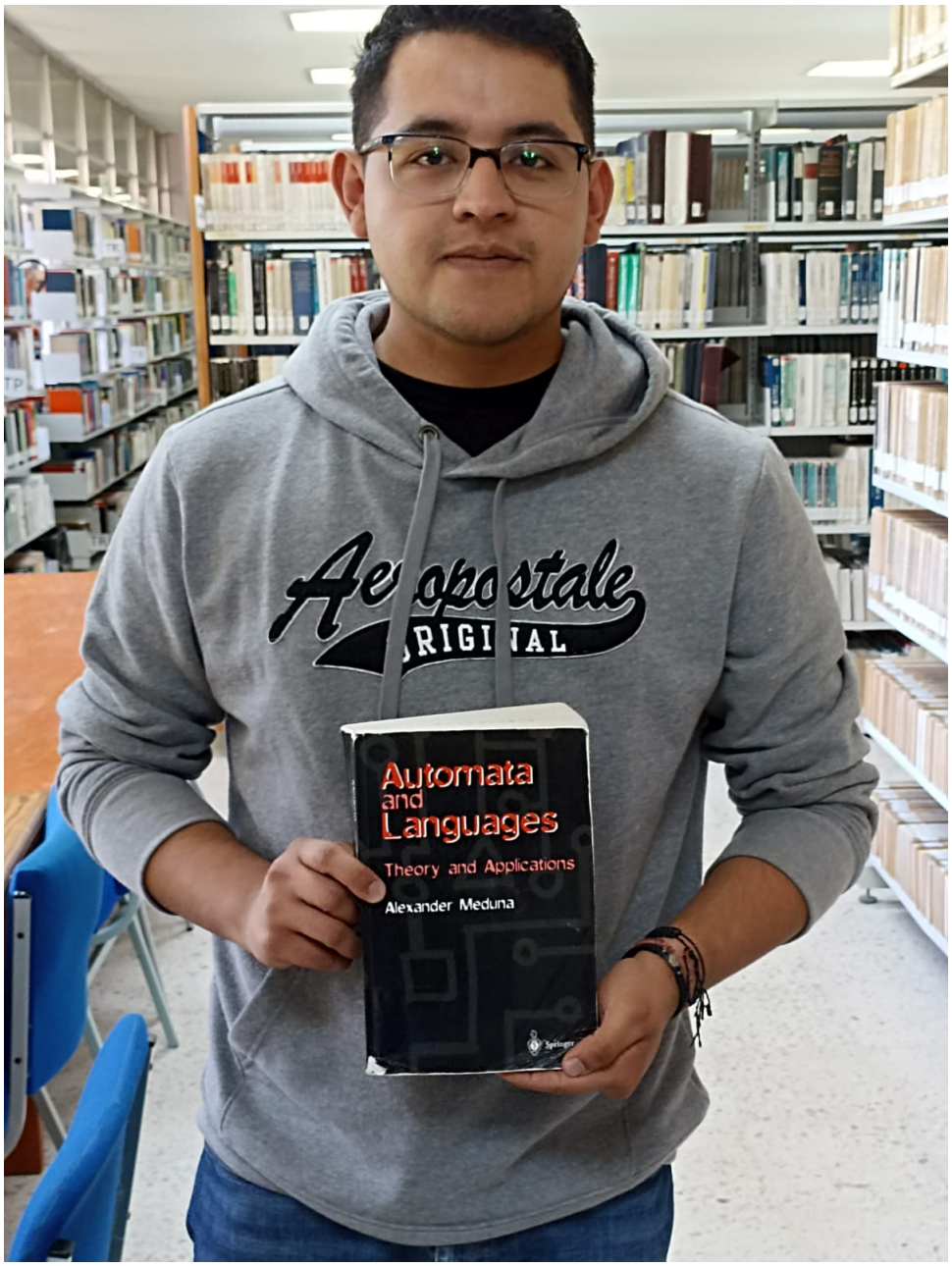
1. YouTube
2. Microsoft Word
3. Notion

4. Desarrollo

Fotografía con los libros usados




LENGUAJES FORMALES
Y TEORÍA DE AUTÓMATAS
Alfaomega



5. Conclusiones

En esta práctica se fortalecieron habilidades en el manejo de lenguajes formales y autómatas, comprendiendo su importancia en la teoría de la computación. Los conceptos de AFD y AFND, junto con sus conversiones. Además, el uso de autómatas para pattern matching abrió nuevas perspectivas sobre aplicaciones prácticas.

Referencias Bibliográficas

References

[1] Meduna, A. (2000). Automata and Languages: Theory and Applications. Springer Science Business Media.

[2] Giró, J., Vázquez, J., Meloni, B., Constable, L. (2015). Lenguajes formales y teoría de autómatas. Alpha Editorial.

[3] Brookshear, J. G. (1993). Teoría de la computación: lenguajes formales, autómatas y complejidad. Addison-Wesley Iberoamericana Espana, S.A.