Universidad Nacional del Altiplano
Facultad de Ingeniería Estadística e Informática
Docente: Fred Torres Cruz
Alumnos: Alex Dannis Lipa Quispe - Roberto Angel Ticona Miramira

Propuesta técnica para optimización en la búsqueda de información en la base de datos de seguimiento de niños del Centro de Salud de San Sebastián - Cusco

Enlace al repositorio de Github: Repositorio

1. Introducción

En el centro de salud de San Sebastián de Cusco, el acceso a la información de los pacientes pediátricos se realiza actualmente mediante una hoja de cálculo en Excel, donde el personal debe buscar manualmente el registro de cada niño utilizando su número de DNI. Aunque este método es funcional, presenta limitaciones en términos de rapidez y eficiencia, especialmente cuando la base de datos crece y el número de consultas diarias aumenta.

Este proyecto propone la implementación de un algoritmo de búsqueda optimizado que permita localizar de forma inmediata la información de los niños con solo ingresar su DNI. La solución no solo reducirá el tiempo de consulta, sino que también minimizará errores humanos, mejorando significativamente la atención y los procesos administrativos en el centro de salud.

1.1. Contexto del problema

1.1.1. Centro de Salud de San Sebastián

El establecimiento de salud Centro de Salud San Sebastián es de categoría I- en el distrito de San Sebastián y pertenece a la jurisdicción de la Dirección de Salud Cusco. Tiene como objetivo ayudar a la dignidad de las personas, promoviendo la salud, previniendo las enfermedades y garantizando la atención universal de salud de los habitantes de Cusco; ofreciendo y dirigiendo los fines de políticas sanitarias en trato con los sectores públicos y los actores sociales.

Figura 1: Centro de Salud de San Sebastián de Cusco

Figura 2: Población estimada atendida en el C.S San Sebastián 2022

2. Descripción del problema

2.1. Flujograma de atención

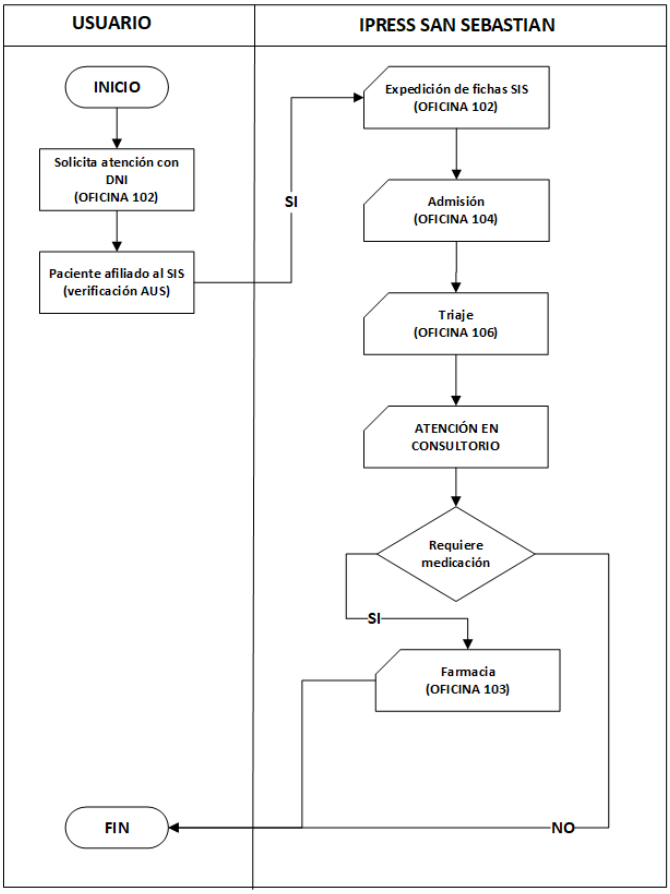

Figura 3: Flujograma de atención

2.2. Flujograma de atención en el consultorio

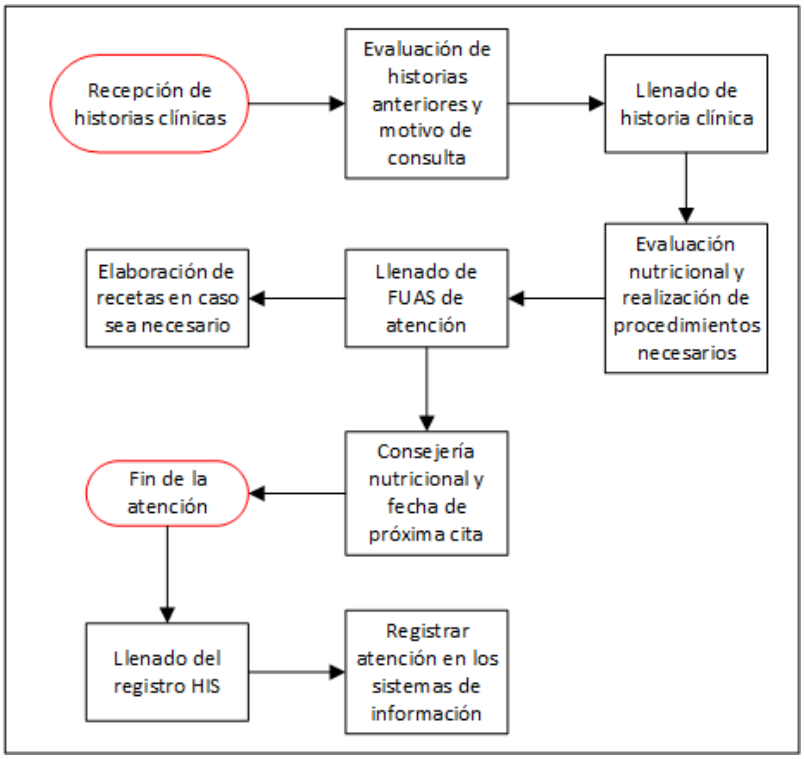

Figura 4: Flujograma de atención en el consultorio nutricional

2.2.1. Sobre el registro y seguimiento de los pacientes

Luego de realizar todos los procedimientos necesarios según el tipo de paciente atendido, el servicio de nutrición del C.S.S.S tiene una base de datos en excel donde se registran los datos de los pacientes atendidos en el día, así como el uso de esta base de datos para registrar las próximas consultas y registrar información necesaria para el seguimiento de los niños.

Base de datos del consultorio nutricional del C.S.S.S: Base de datos

Problemática El sistema actual para gestionar la información de los niños en el centro de salud utiliza una hoja de cálculo en Excel. Si bien este método es funcional para registrar y almacenar datos, no está diseñado para manejar búsquedas rápidas y eficientes, especialmente cuando se trata de bases de datos con un volumen creciente de registros.

La búsqueda manual del registro de un niño mediante su DNI requiere que el personal revise filas de datos, lo que puede ser un proceso lento y propenso a errores. Esto no solo incrementa el tiempo necesario para atender consultas, sino que también dificulta la experiencia del personal, especialmente en situaciones donde la rapidez y precisión son críticas.

Estas limitaciones hacen evidente la necesidad de un sistema más eficiente, capaz de recuperar la información de manera inmediata, optimizando el tiempo de respuesta y mejorando la calidad del servicio ofrecido por el centro de salud.

3. Solución propuesta

La solución propuesta consiste en implementar un algoritmo de búsqueda eficiente que permita acceder a la información de los niños directamente desde la base de datos con solo ingresar el número de DNI. Este algoritmo será integrado en el sistema de gestión del centro de salud y está diseñado para superar las limitaciones del método actual basado en hojas de cálculo.

El enfoque principal de esta solución es optimizar la estructura de la base de datos mediante el uso de un índice en el campo DNI, lo que permitirá realizar búsquedas directas en lugar de recorridos secuenciales. Esto reducirá significativamente el tiempo de búsqueda, incluso en bases de datos con un gran número de registros.

Con esta solución, se espera mejorar la eficiencia operativa, reducir los errores asociados con búsquedas manuales y brindar un servicio más rápido y confiable tanto para el personal como para los usuarios del centro de salud.

3.1. Implementación técnica

3.1.1. Modificación de la base de datos original

Para implementar los códigos solicitados se modifico la base de datos original.

Base de datos del consultorio nutricional del C.S.S.S modificada: Base de datos modificada

3.1.2. Lenguaje de programación utilizado

Se utilizo el lenguaje de programación Python, reconocido por su simplicidad, legibilidad y versatilidad. Su diseño intuitivo y la abundancia de bibliotecas disponibles lo convierten en una herramienta ideal para desarrollar soluciones rápidas y eficientes, especialmente en proyectos relacionados con manejo de datos y algoritmos. Python cuenta con facilidades para la programación orientada a objetos, imperativa y funcional, por lo que se considera un lenguaje multi-paradigmas. Fue basado en el lenguaje ABC y se dice que fue influenciado por otros como C, Algol 60, Modula-3 e Icon según su propio autor. Es un lenguaje de alto nivel ya que contiene implícitas algunas estructuras de datos como listas, diccionarios, conjuntos y tuplas, que permiten realizar algunas tareas complejas en pocas líneas de código y de manera legible. Holguín y cols. (2014)

En el contexto del problema que buscamos resolver, Python ofrece varias ventajas clave:

- Manejo eficiente de datos: Python cuenta con bibliotecas como pandas, que permiten manipular y realizar operaciones sobre conjuntos de datos de manera eficiente. Esto es útil para trabajar con bases de datos como la que utilizamos en el centro de salud.
- Implementación de algoritmos: Gracias a su estructura clara y sus capacidades avanzadas, Python permite implementar algoritmos de búsqueda optimizados de forma rápida y eficaz.

La elección de Python como herramienta principal para este proyecto permite construir una solución sólida, adaptable y fácil de mantener, asegurando la mejora en los procesos de búsqueda de información en el centro de salud.

3.2. Algoritmos de búsqueda utilizados

3.2.1. Búsqueda lineal

La búsqueda lineal es un algoritmo sencillo que consiste en recorrer secuencialmente cada elemento de una lista o conjunto de datos hasta encontrar el valor deseado o hasta llegar al final de la lista. Su simplicidad lo hace ideal para situaciones donde los datos no están ordenados o cuando el tamaño del conjunto de datos es relativamente pequeño.Page (2018)

3.2.2. Búsqueda binaria

La búsqueda binaria es un potente algoritmo diseñado para encontrar de forma eficiente un valor en un conjunto de datos ordenado. La idea central de la búsqueda binaria es sencilla: En lugar de comprobar cada elemento del conjunto de datos uno a uno, como en una búsqueda lineal, la búsqueda binaria reduce el intervalo de búsqueda a la mitad con cada paso, lo que hace que el proceso sea mucho más rápido.Mckee (2024)

3.2.3. Búsqueda por salto

Jump Search es un algoritmo de búsqueda en el cual, en vez de irse de manera secuencial, salta un grupo de elementos de manera repetitiva hasta encontrar el valor buscado o uno mayor. Si se encuentra el valor, termina nuestro algoritmo. Si llegamos a un valor mayor, retrocedemos de uno en uno hasta llegar al valor buscado. Es importante mencionar que el arreglo debe estar ordenado.Tejeda (2023)

3.2.4. Búsqueda exponencial

La búsqueda exponencial, también conocida como búsqueda duplicada o búsqueda con los dedos, es un algoritmo creado para buscar elementos en matrices de gran tamaño. Es un proceso de dos pasos. Primero, el algoritmo intenta encontrar el rango (L, R) en el que está presente el elemento objetivo y luego utiliza la búsqueda binaria dentro de este rango para encontrar la ubicación exacta del objetivo.

Se denomina búsqueda exponencial porque encuentra el elemento que contiene el rango buscando el primer exponente k para el que el elemento en el índice pow(2,k) es mayor que el objetivo. Aunque su nombre es búsqueda exponencial, la complejidad temporal de este algoritmo es logarítmica. Es muy útil cuando los Arrays son de tamaño infinito y convergen a una solución mucho más rápido que la búsqueda binaria.Jindal (2023)

3.2.5. Búsqueda de fibonacci

El método de búsqueda de Fibonacci divide la lista de búsqueda en partes proporcionales basadas en los números de Fibonacci. Los índices de comparación son calculados en función de la relación entre estos números, lo que permite reducir el tamaño del rango de búsqueda en cada iteración.

La idea principal es aprovechar la propiedad de los números de Fibonacci: cada número es la suma de los dos anteriores. Esto permite ajustar dinámicamente los rangos de búsqueda y evitar operaciones costosas como la división, que es común en la búsqueda binaria.

4. Validación de la solución

Para evaluar cual seria el mejor código que se debería implementar en nuestro problemática se presenta la siguiente tabla con los resultados:

4.1. Resumen de los algoritmos

Método de búsqueda empleado	Tiempo de ejecución	Número de iteraciones	Complejidad
Búsqueda lineal	0.8053 segundos	1567	O(n + m)
Búsqueda binaria	0.7785 segundos	22	O(n log n)
Búsqueda por salto	0.7955 segundos	22	$O(\sqrt{n})$
Búsqueda exponencial	0.7576 segundos	554	O(log n)
Búsqueda de fibonacci	1.0707 segundos	4	O(log n)

Cuadro 1: Ejemplo de tabla con 4 columnas y 6 filas.

4.2. Resultados obtenidos

4.2.1. Búsqueda lineal

Código en python de búsqueda lineal aplicado en la problemática: Búsqueda lineal

```
PS D:\Programación\Proyecto\Busquedas> python Busqueda2-lineal.py
Ingrese la ruta base donde se encuentra la carpeta 'Datos': D:\Programación\Proyecto\Datos
Ingrese el año y mes (AAAA-MM): 2023-01
Ingrese el DNI que desea buscar: 92962631
Resultados encontrados para el DNI 92962631 en el mes 2023-01:
- Apellidos y Nombres: HONOR CHAVEZ CALEB
- Historial Clínica: 34009 B
- Fecha de Nacimiento: 2022-07-05 00:00:00
- Edad Actual: 2
- 1er Mes de Tratamiento: 2023-01-05 00:00:00
- HB Corregida: nan
- DX: ANEMIA
- 2do Mes de Tratamiento: nan

Tiempo transcurrido para encontrar el DNI: 0.8053 segundos.
Iteraciones realizadas: 1567
Complejidad estimada: O(n + m) con n = 1567 iteraciones.
PS D:\Programación\Proyecto\Busquedas>
```

Figura 5: Resultado búsqueda lineal

4.2.2. Búsqueda binaria

Código en python de búsqueda binaria aplicado en la problemática: Búsqueda binaria

```
PS D:\Programación\Proyecto\Busquedas> python Busqueda1-binaria.py
Ingrese la ruta base donde se encuentra la carpeta 'Datos': D:\Programación\Proyecto\Datos
Ingrese el año y mes (AAAA-MM): 2023-01
Ingrese el DNI que desea buscar: 92962631
Resultados encontrados para el DNI 92962631 en el mes 2023-01:
- Apellidos y Nombres: HONOR CHAVEZ CALEB
- Historial Clínica: 34009 B
- Fecha de Nacimiento: 2022-07-05 00:00:00
- Edad Actual: 2
- 1er Mes de Tratamiento: 2023-01-05 00:00:00
- HB Corregida: nan
- DX: ANEMIA
- 2do Mes de Tratamiento: nan

Tiempo transcurrido: 0.7785 segundos.
Iteraciones realizadas: 22
Complejidad estimada: O(log n) para la búsqueda binaria con n = 22.
Además, el ordenamiento tiene complejidad O(n log n).
PS D:\Programación\Proyecto\Busquedas>
```

Figura 6: Resultado búsqueda binaria

4.2.3. Búsqueda por salto

Código en python de búsqueda por salto aplicado en la problemática: Búsqueda por salto

```
PS D:\Programación\Proyecto\Busquedas> python Busqueda3-salto.py
Ingrese la ruta base donde se encuentra la carpeta 'Datos': D:\Programación\Proyecto\Datos
Ingrese el año y mes (AAAA-MM): 2023-01
Ingrese el DNI que desea buscar: 92962631
Resultados encontrados para el DNI 92962631 en el mes 2023-01:
- Apellidos y Nombres: HONOR CHAVEZ CALEB
- Historial Clínica: 34009 B
- Fecha de Nacimiento: 2022-07-05 00:00:00
- Edad Actual: 2
- 1er Mes de Tratamiento: 2023-01-05 00:00:00
- HB Corregida: nan
- DX: ANEMIA
- 2do Mes de Tratamiento: nan

Tiempo transcurrido: 0.7955 segundos.
Iteraciones realizadas: 22
Complejidad estimada: O(√n) con n = 22.
PS D:\Programación\Proyecto\Busquedas>
```

Figura 7: Resultado búsqueda por salto

4.2.4. Búsqueda exponencial

Código en python de búsqueda exponencial aplicado en la problemática: Búsqueda exponencial

```
PS D:\Programación\Proyecto\Busquedas> python Busqueda5-exponencial.py
Ingrese la ruta base donde se encuentra la carpeta 'Datos': D:\Programación\Proyecto\Datos
Ingrese el año y mes (AAAA-MM): 2023-01
Ingrese el DNI que desea buscar: 92962631
Resultados encontrados para el DNI 92962631 en el mes 2023-01:
- Apellidos y Nombres: HONOR CHAVEZ CALEB
- Historial Clínica: 34009 B
- Fecha de Nacimiento: 2022-07-05 00:00:00
- Edad Actual: 2
- 1er Mes de Tratamiento: 2023-01-05 00:00:00
- HB Corregida: nan
- DX: ANEMIA
- 2do Mes de Tratamiento: nan

Tiempo transcurrido para encontrar el DNI: 0.7576 segundos.
Interacciones realizadas: 554
Complejidad estimada: O(log n) para la fase exponencial + búsqueda lineal.
PS D:\Programación\Proyecto\Busquedas>
```

Figura 8: Resultado búsqueda exponencial

4.2.5. Búsqueda de fibonacci

Código en python de búsqueda de fibonacci aplicado en la problemática: Búsqueda de fibonacci

```
PS D:\Programación\Proyecto\Busquedas> python Busqueda4-fibonacci.py
Ingrese la ruta base donde se encuentra la carpeta 'Datos': D:\Programación\Proyecto\Datos
Ingrese el año y mes (AAAA-MM): 2023-01
Ingrese el DNI que desea buscar: 92962631
Resultados encontrados para el DNI 92962631 en el mes 2023-01:
- Apellidos y Nombres: HONOR CHAVEZ CALEB
- Historial Clínica: 34009 B
- Fecha de Nacimiento: 2022-07-05 00:00:00
- Edad Actual: 2
- 1er Mes de Tratamiento: 2023-01-05 00:00:00
- HB Corregida: nan
- DX: ANEMIA
- 2do Mes de Tratamiento: nan

Tiempo transcurrido: 1.0707 segundos.
Iteraciones realizadas: 4
Complejidad estimada: O(log n).
PS D:\Programación\Proyecto\Busquedas>
```

Figura 9: Resultado búsqueda fibonacci

5. Conclusiones

Se pudo realizar la implementación de un distintos códigos que solucionaban el mismo problema de la base de datos obtenida del Centro de Salud de San Sebastián, evaluando su tiempo de ejecución, el número de iteraciones y la complejidad que demandan; concluyendo que la mejor opción considerando que los datos están desordenados, es aplicar la búsqueda lineal; pero si aplicamos un algoritmo para ordenar los datos, el algoritmo que realiza la búsqueda en un tiempo reducido es la búsqueda exponencial.

Referencias

Holguín, C., Díaz-Ricardo, Y., y Becerra-García, R. A. (2014). The programming language python. *Ciencias Holguin*. Descargado de http://www.linuxjournal.com/article/2959

Jindal, H. (2023). *Busqueda exponencial.* Descargado de https://www.delftstack.com/es/tutorial/algorithm/exponential-search/

Mckee, A. (2024). *Búsqueda binaria.* Descargado de https://www.datacamp.com/es/tutorial/binary-search-python

Page, B. (2018). *Búsqueda lineal.* Descargado de https://pier.guillen.com.mx/algorithms/03-ordenacion/03.6-busqueda_lineal.htm

Tejeda, R. (2023). *Jump search.* Descargado de https://blog.aviada.mx/es/jump-search/