UNIVERSIDAD NACIONAL DEL ALTIPLANO
"UNA PUNO"

ESCUELA PROFESIONAL DE INGENIERIA
ESTADISTICA E INFORMATICA

OPTIMIZACION DE ANALISIS DE MEDIO MILLON DE DATOS DE HECHOS VITALES UTILIZANDO THREADS

Est: Coasaca Callacondo Rubi melania
Gmail: rcoasacac@est.unap.edu.pe
Est: Chata Iscarra Grilia Yaneth
Gmail: gchatai@est.unap.edu.pe

28 noviembre 2022

Resumen

Objetivo: Optimizar el tiempo del análisis de medio millón de datos de Hechos Vitales (nacimientos, defunciones y matrimonios) utilizando hilos(Threads).

Metodología: Se creó un código en python sin threads y otro con threads, después se compilo en ambos códigos la misma cantidad de datos, para saber si el codigo en python con threads es optimo en el análisis de los 500000 hechos vitales.

Resultados: El código en python con Threads resulto ser el mas optimo en el análisis de los 500000 datos de Hechos Vitales en un 19.7%.

Conclución: El código en python con threads es mas óptimo que el código sin threads por lo cual se concluye que los threads es recomendable en análisis de datos en trabajos similares.

Abstract

Objetive: Optimize the analysis time of half a million Vital Events data (births, deaths and marriages) using threads.

Methodology: A python code without threads and another with threads were created, then the same amount of data was compiled in both codes, to know if the python code with threads is optimal in the analysis of the 500000 vital events .

Result: The python code with Threads turned out to be the most optimal in the analysis of the 500000 Vital Facts data at 19.7%.

Conclución: The python code with threads is more optimal than the code without threads, therefore it is concluded that threads are recommended in data analysis in similar works.

Introducción:

El Perú genera millones de registros de datos a diario, uno de esos registros es los Hechos Vitales que a diario va en aumento debido a que a diario se generan sucesos de actas de (nacimientos, defunciones y matrimonios) actualmente la última fecha que se actualizó la data de este registro fue el 19- 06-2022 (INEI, 2022).Al existir tal cantidad de datos lo que se pretende en este estudio es optimizar el análisis de los 500000 datos de los Hechos Vitales implementando un código en python en donde se optimizara el análisis de datos mediante la computación paralela, [8] implementando hilos (threads).

Metodología:

Computación paralela: En este estudio se utililizara la computación paralela, este metodo de programación utiliza multiples procesadores, [3]una tarea lo divides en pequeñas tareas para que el tiempo de compilación sea óptimo,La arquitectura Paralela da un enfoque constructivo del conocimiento del lenguaje basado en la computación paralela que es el uso de múltiples recursos computacionales, los cuales requieren de una gran capacidad de procesamiento y de espacio de memoria, debido a complejas operaciones que se deben realizar, [9]

Hilos(Threads): Se aprovecha las tecnologias de multicore y multithreading para realizar la paralelización, Se crean threads que comparten parte de memoria para compartir resultados globales entre los threads. [4]

Datos: Los datos corresponden a la base de datos obtenida de la Plataforma Nacional de Datos Abiertos en el Opción Gobernanza Hechos Vitales `bit.ly/3udD9Wh` esta información es proporcionada por el Instituto Nacional de Estadística e Informática - INEI.

Descripción de datos: Se esta trabajando con 500000 datos los cuales estan en un formato CSV, esta data presenta registros de hechos vitales registrados hasta mayo del 2022.La data contiene inscripciones de nacimientos, defunciones y matrimonios en las oficinas registrales y oficinas de registro del estado civil a nivel nacional, desagregado por año de inscripción, mes de inscripción, sexo, tipo de hecho vital, tipo de inscripción y lugar donde se efectuó la inscripción.

Atributo	Descripción
AÑO_INSCRIPCION	Mes en que se realizó la inscripción
MES_INSCRIPCION	Mes de registro del hecho
COD_SEXO	Sexo del titular del acta
COD_HECHO	Hecho vital (Nacimiento, matrimonio o defunción)
COD_INSCRIPCION_1	Tipo de inscripción: Ordinaria, extemporánea, judicial, extranjero, etc)
COD_INSCRIPCION_2	Acta de inscripción o acta generada
UBIGEO_INEI_L	Código de Ubicación geográfica INEI
DEPA_CONT_L	Departamento donde se realizó la inscripción
PROV_PAIS_L	Provincia donde se realizó la inscripción
DIST_CIUD_L	Distrito donde se realizó la inscripción
CANTIDAD	Cantidad de casos

Figure 1: Fuente: INEI

Lógica de Implementación: Se realizará un código en python donde se aplicara la computacion paralela mediante hilos(threads) [3]

- Se utilizó el lenguaje de programacion Python, donde se creo 2 codigos el primero sin threads y el segundo con threas.

- Los codigos realizan un conteo y lectura del medio millón de datos de la cantidad de atributos que se le indica.

- El programa inicia contabilizanzo los datos hasta que termine todo el recorrido de la data CSV.

- El programa tiene como fin calcular el tiempo que demoro lecturando la cantidad de datos que se le solicito leer y contar.

Implementación: Los threads representan una plataforma que utiliza recursos de la CPU para resolver problemas. Esta plataforma puede usar un solo thread para resolver un problema paso a paso con un solo procesador cuando el proceso no tiene secciones que se puedan paraleliza, [5]. Lo cual nos ayudara a optimizar el tiempo de análisis de datos:

-la librería pandas se utilza para la exportacion y el procesamiento de los 500000 datos.

-La libreria time es quien se encarga de controlar el tiempo de ejecución de los datos.

-La libreria Numpy sirve para realizar matrices con mayor velocidad y guardar gran cantidad de información. [1]

-Los datos son exportados, se continua con la extracción de los atributos de los datos; y selecciona los atributos con los que trabjará.
-El contador de tiempo se inicia una vez que se asigno los atributos.

-La función contar datos se encarga de contabilizar todo los datos que le pertenescan al atributo solicitado.

-Se comienza con el primer atributo que se asigno y su lectura y conteo respectivo.

-De manera sucesiva se pasa de atributo a atributo con su respectivo conteo segun el atributo solicitado.

-Imprime la cantidad de datos y el tiempo que se demoró en contabilizarlos y lecturarlos.

Pseudocodigo: El Pseudocódigo es uno de los métodos más utilizados para el diseño de algoritmos, y es independiente del lenguaje de programación que se vaya a utilizar, [7].

Pseudocodigo con threads: [2]

```
inicio
exportar base de datos
leer base de datos
bucle base de datos
        extraer atributos
quitar atributos no ingresados
función contar(atributo)
        contador
        bucle dato in base de datos
                si dato == atributo
                        contador+1
        imprimir contador
bucle atributo in atributo
        iniciar hilo (contar, atributo)
```

Pseudocodigo sin threads:

```
inicio
exportar base de datos
leer base de datos
bucle base de datos
        extraer atributos
quitar atributos no ingresados
función contar(atributo)
        contador
        bucle dato in base de datos
                si dato == atributo
                        contador+1
        imprimir contador
```

Codigo

Librerias:

```
from numpy import var
import pandas as pd
import time
```

Figure 2: librerias python

Código sin Threads:

```
url = "Hechos_Vitales.csv"
data = pd.read_csv(url)

atributos=[]
for i in data:
    atributos.append(i)

#atributos.pop(1)
#atributos.pop(2)
atributos.pop(3)
del atributos[1:3]

start = time.time()
for v in atributos:
    clase=[]
    cont=[]
    for i in data.index:
        if data[v][i] not in clase:
            clase.append(data[v][i])
            cont.insert(clase.index(data[v][i]),1)
        else:
            cont[clase.index(data[v][i])]+=1

end = time.time()
print("Done, time taken", end - start)
```

Figure 3: codigo python

Código con Threads:

```
def contar(v,data):
    clase=[]
    cont=[]
    for i in data.index:
        if data[v][i] not in clase:
            clase.append(data[v][i])
            cont.insert(clase.index(data[v][i]),1)
        else:
            cont[clase.index(data[v][i])]+=1
url = "Hechos_Vitales.csv"
data = pd.read_csv(url)

atributos=[]
for i in data:
    atributos.append(i)

#atributos.pop(1)
atributos.pop(2)
#atributos.pop(3)
del atributos[1:3]

start = time.time()
for v in atributos:
    #print("hilo "+str(variables.index(v)))
    t = Thread(target=contar,args = (v,data))
    t.start()
t.join()
end = time.time()
print("Done, time taken", end - start)
```

Figure 4: codigo python

Resultados:

la data de Hechos Vitales cuenta con 11 atributos y 500000 datos de los cuales se realizó la prueba 10 veces con 10 atributos. El siguiente cuadro muestra una comparación del tiempo que demoro el código en python con threads y el sin threads.

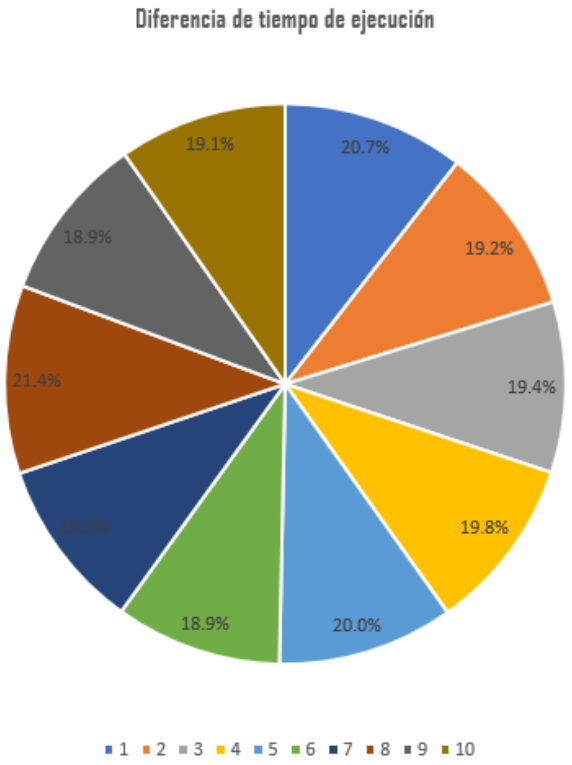
atributos	Tiempo de compilación		Diferencia de Tiempo (%)
	Sin threads	Con threads	
1	232.3663998	213.1211798	20.7%
2	225.3104243	205.8117213	19.2%
3	221.7641857	201.9206567	19.4%
4	226.0203583	205.9935672	19.8%
5	228.8250995	209.8369839	20%
6	255.7286234	236.1488495	18.9%
7	211.1879025	189.7333343	19.5%
8	241.0740626	222.0782423	21.4%
9	242.9323359	223.7528777	18.9%
10	232.3663998	213.1211798	19.1%

Interpretación En el cuadro comparativo se puede apreciar que el código en lenguaje python utilizando threads fue mas optimo

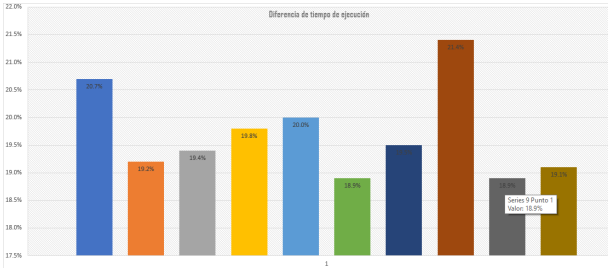
en tiempo de ejecución que el codigo en python sin threads.

Estadística descriptiva: promedio:

Gráfico pastel o circular: [10]



Interpretació: El gráfico de pastel nos muestra que las 10 pruebas que se realizaron a 10 atributos de 500000 datos de hechos vitales (matrimonio,nacimientos,defunciones) la variación entre cada prueba es mínima.



Interpretació: De igual forma gráfico de barras nos muestra que las 10 pruebas que se realizaron a 10 atributos de 500000 datos de hechos vitales (matrimonio,nacimientos,defunciones) la variación entre cada prueba es mínima.

promedio: Sumaremos todos los porcentajes de diferencia de tiempo de ejecución para conocer el porcentaje de obtimización del cdigo de python con hilos(threads). [6]

Para ello utilizaremos la siguiente fórmula:

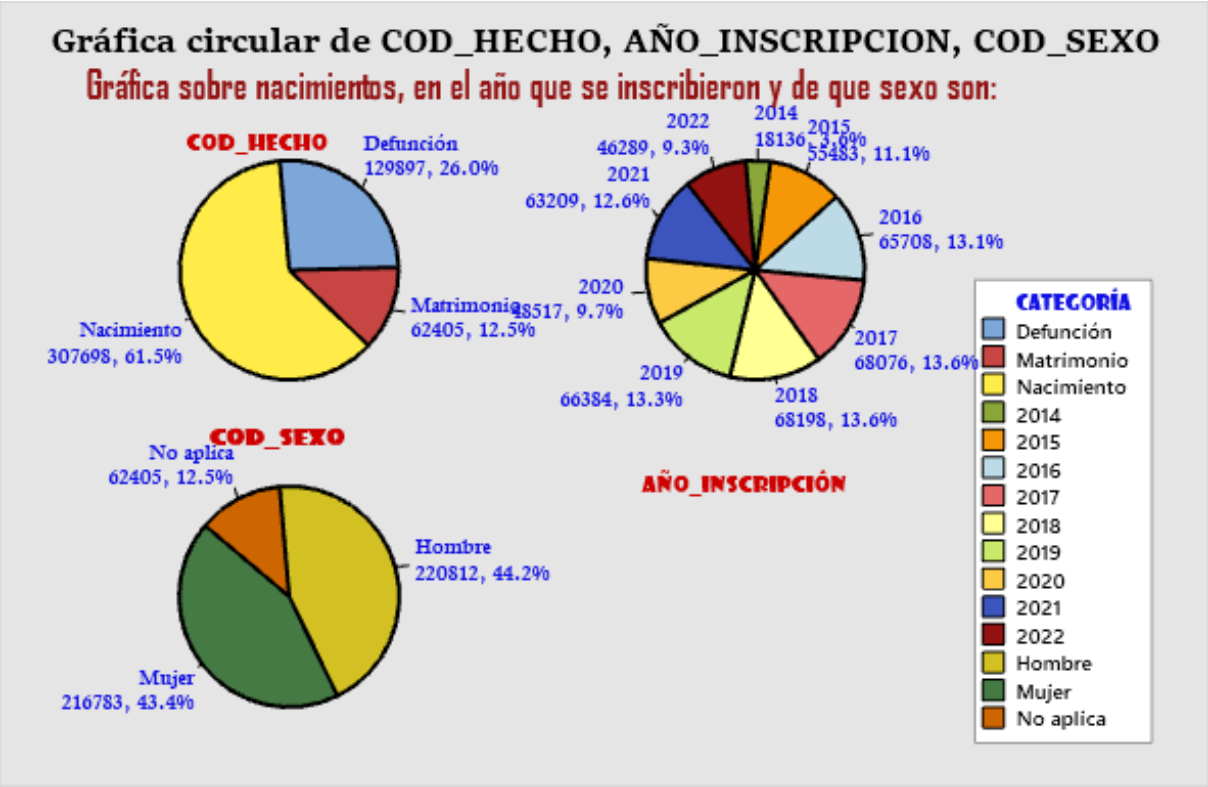
$$\bar{X} = \frac{X_1 + X_2 \ldots . + X_n}{n}$$

$$\bar{X} = \frac{20.7+19.2+19.4+19.8+20.0+18.9+19.5+21.4+18.9+19.1}{10}$$

$$\bar{X} = 19.7$$

Interpretació: El promedio de la diferencias entre el código en python con threads y él código en python sin threads es de 19.7%.

Conclusión: Se ve una notoria difencia entre en el tiempo de ejecución del código con threads. Por lo tanto se concluye que utilizar threads es efectivo para casos similares en donde se requiere que sea mas optimo el análisis de datos.



Interpretaciones:

Interpretación para el gráfico 1 (Cod-hecho): El gráfico de pastel nos muestra que de un total de 500000 datos de hechos vitales nos dice que 307698 individuos nacieron esto representa un 61.5%, enseguida podemos observar que 129897 que es 26% fallecieron o dejaron de existir y finalmente 62405 personas que representa un 12.5% se casaron o tomaron matrimonio.

Interpretación para el gráfico 2(Año-inscripción): El gráfico pastel nos muestra que de un total de 500000 datos de hechos vittales nos dice que como máximo se inscribieron en el año 2018 un total de 68198 individuos que representa un 13.6%, enseguida observamos que en el 2017 fue inscrito 68076 individuos que representa el 13.6%, luego en el 2016 se incribieron 65708 individuos representando un 13.1%, después en el 2019 se inscribieron 66384 individuos que representaría un 13.3%, enseguida en el 2021 se inscribieron 63209 individuos representando un 12.6% , asi sucesivamente y finalmente podemos observar que como mínima parte fue en el 2014 se inscribieron 18136 individuos que representaría un 3.6%.

Interpretación para el gráfico 3 (Cod-sexo): El gráfico pastel nos muestra que un total de 62405 individuos no aplicaron esto representa la mínima parte con 12.4%, enseguida un total de 220812 son hombres esto representa como máximo un 44.2% y finalmente fueron mujeres un total de 216783 que representa un 43.4%.

References

[1] Alexander David Andrango Ayo. Implementación de un sistema de detección de monedas por visión artificial: Sistema de detección de patrones simples de monedas. B.S. thesis, Quito: EPN, 2022., 2022.

[2] Matías S Ávalos. 1. resolución de problemas.

[3] Héctor Rico García. *Aplicación de técnicas de computación paralela para la aceleración de algoritmos de ingeniería*. PhD thesis, Universitat d'Alacant-Universidad de Alicante, 2021.

[4] Mario Ibáñez Bolado et al. Paralelización de técnicas de toma de decisiones en empresas de acuicultura. 2022.

[5] Oriel Keba Manianga et al. Aplicación de cómputo en paralelo basado en threads en el cálculo de métricas usadas en el análisis armónico. 2021.

[6] Maryluz Largo Aguirre. Estadistica descriptiva con excel: Grado 9. 2017.

[7] Antonio López García and Jaime Urquiza-Fuentes. Experiencias de uso del pseudocódigo y java en la enseñanza de programación en ciclos formativos y bachillerato. 2022.

[8] Mateo José Moinelo Torres. *Simulación de ondas estacionarias mediante computación paralela y software estándar de computación científica*. PhD thesis, Universitat Politècnica de València, 2022.

[9] Erika Patricia Ponce Gallegos. Estudio de herramientas de desarrollo en entornos paralelos para aplicaciones de agricultura de precisión. B.S. thesis, 2022.

[10] Yusimí Guerra Véliz, Andier Aguilar García, and Julio Leyva Haza. Aprendizaje de la estadística descriptiva en secundaria básica con datos provenientes del consumo de energía. *Horizonte de la Ciencia*, 11(21):201–215, 2021.