

Universidad Autónoma Del Estado
De México

Unidad Académica Profesional de Tlanguistenco

INGENIERÍA DE SOFTWARE

Programación paralela

José Armando López Alvarez

Problemática.

El problema de la suma de subconjuntos es un problema importante en la teoría de la complejidad y la criptografía.

El problema que intenta resolver este programa es que dado un conjunto de enteros, ¿existe algún subconjunto cuya suma sea exactamente cero?

El problema presentado se clasifica como un NP-completo.

Análisis

*Requerimientos funcionales

- Conjunto de números aleatorios, que se encuentren entre un rango.
- Adaptado a programación paralela
- Que haga uso de la mayor parte de la potencia del PC
- Disminuir el tiempo que tarda en realizar la ejecución comparado con el algoritmo estructurado sin programación paralela

Requerimientos no funcionales

- Interfaz gráfica
- Mostrar el tiempo transcurrido
- Mostrar los subconjuntos creados
- Mostrar la suma de cada subconjunto
- Programa sencillo.

Tipos de datos

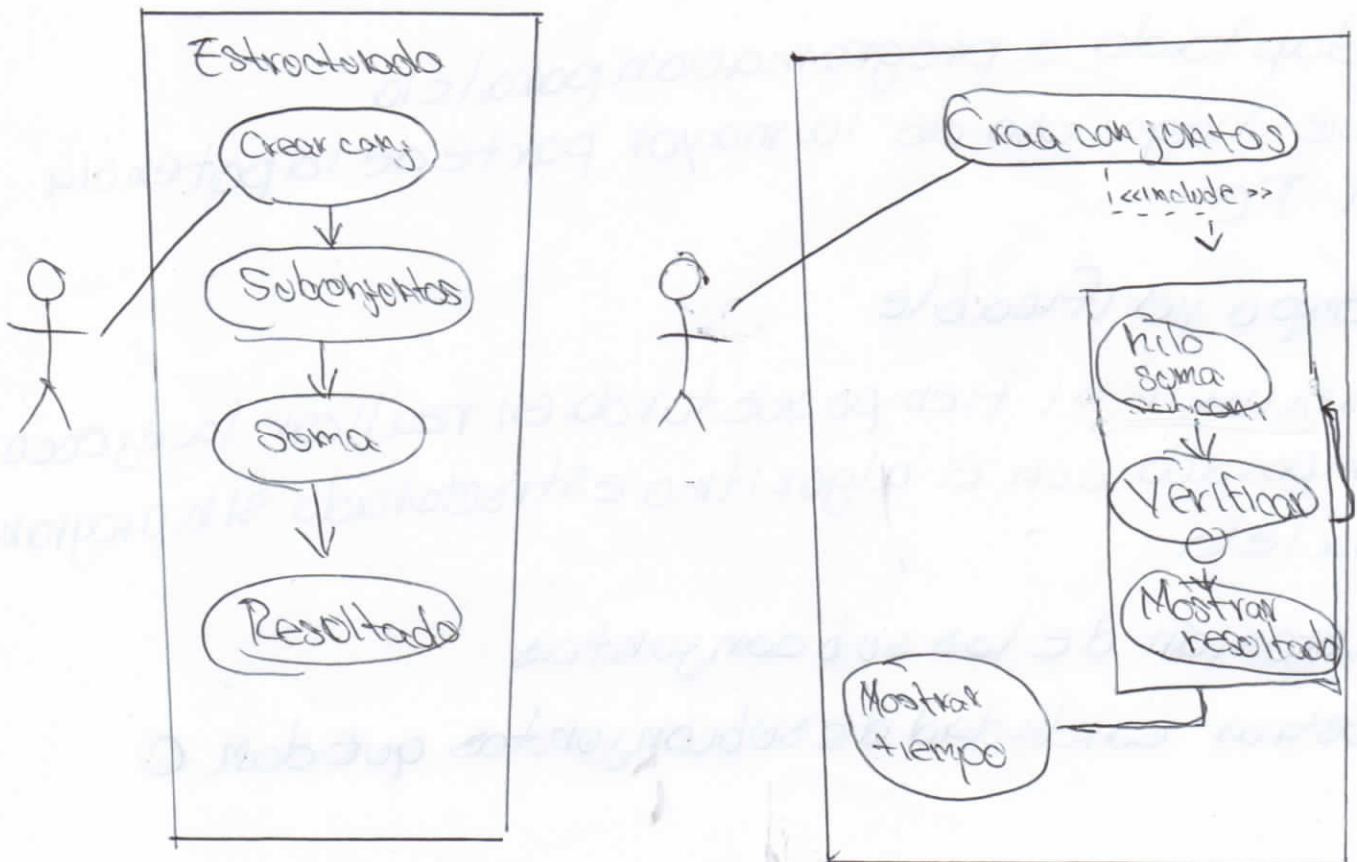
if

for

in range

list

Caso de uso










Metodología

Cascada, debido a 1 paso + tiempo para el desarrollo y porque permite volver a etapas anteriores y corregir errores.

Problemas que podrían presentarse

- Mala optimización del algoritmo
- Ciclos infinitos al crear los hilos
- Tiempo mayor al algoritmo incremental

Cronograma

Actividad	29 Nov - 5 Dic	6 - 12 Dic	13 - 18 Dic
Planeación			
Id. de req.			
Implementación			
Pruebas			
Id. de req.			
Implementación			
Pruebas			

Diseño

- Programación Paralela

Lenguaje utilizado

- Python 3.9.0 64bit.

Otras herramientas

- Visual Studio Code

Estructuras de control

if (Condición):

 instrucciones

for i in range(x, y):

 instrucciones

for element in elements:

 instrucciones

for _ in range(x):

 instrucciones

Interfaz

La primera idea fue haber hecho una interfaz pero debido a que lo que se busca es incrementar la eficacia del algoritmo, más no hacer que el programa se vea bonito, se deshecha esta idea, la consola de visual studio code tiene un límite de muestra de los datos, no muestra todos los subconjuntos creados, pero creo que para la tarea que se pide en el problema es más que suficiente la información que muestra la consola, en futuras actualizaciones podrá hacerse uso de la interfaz de Tkinter en python. Otra razón fue porque desde el inicio se maneja una programación sin uso de objetos, y pasar lo todo a ese enfoque hubiera sido algo tardado. Y afecta mucho al rendimiento del programa.

Implementación

Para este proyecto hice uso de las herramientas que ofrece el lenguaje de programación Python en específico el módulo multiprocessing, el cual sirve para gestionar los procesos como hilos, podría decirse que python ofrece este módulo como un reemplazo casi directo al módulo threading. y tal como lo dice cierta documentación del módulo, este puede ser usado para hacer uso de los múltiples núcleos de la CPU, y evitar cuellos de botella asociados al bloqueo global del intérprete.

La función Process puede ser llamada con los mismos argumentos que un hilo en Python, contiene procesos daemon, permite observar que proceso se está ejecutando etc.

En el caso específico de este proyecto se creó una función main, que iniciará la ejecución del programa, y que en este caso sirve para llamar al Process, iniciarlo, además de crear las subconjuntos la función target de process es ~~subconjunto~~ subconjunto, que es la que se encarga de sumar los subconjuntos.

Las ejecuciones mostraron resultados favorables, en relación con los resultados mostrados por el algoritmo estructurado, mostrando una mejora en cuanto al tiempo que tarda el método en realizarse.

2a información va mostrándose en consola, y muestra el subconjunto y debajo el resultado de la suma, y al terminar la ejecución muestra el tiempo transcurrido.

También se decidió quitar del método principal la generación de aleatorias ya que tomaba demasiado tiempo, y no se notaba mucha mejora al compararse con el algoritmo estructurado.

Pruebas

Subconjuntos gen	Tiempo	Long. de conjunto.
1000	0.91s	3 Estructurado
10000	9.03s	
100000	91.033s	
1000	0.78s	3 Paralelo
10000	4.66s	
100000	48.40s	