UNIVERSIDAD TECNOLOGICA DE SANTIAGO

SISTEMA CORPORATIVO

FACULTAD DE INGENIERÍA Y ARQUITECTURA

CARRERA DE SISTEMAS COMPUTACIONALES

ALGORITMOS PARALELOS

GRUPO 3

Presentado a:

Iván Mendoza

Presentado por:

Sugeiri Torres	**1-16-0736**
Dinnibel Azcona	**1-16-0788**
Claudia Báez	**2-16-1116**

Santiago de los Caballeros,

República Dominicana

INDICE **PAG.**

1. Introducción

El presente proyecto tiene como finalidad presentar los diferentes algoritmos de búsqueda y ordenamiento, explicar que son y como se implementa. También explicaremos acerca de que es algoritmo paralelo, sus etapas, técnicas y modelos.

Presentaremos de forma practica varios métodos como el quick sort y el método burbuja. Analizando la duración de los mismos hasta lograr determinar cual es el mas eficiente.

2. Descripción del Proyecto

El proyecto se basa en una aplicación que permite ordenar un arreglo y comparar una serie de algoritmos de búsquedas a fin de estudiar su eficiencia y determinar, en base a los tiempos estudiados, cual es el mas rápido.

3. Objetivos

3. 1. Objetivo General

Analizar los diferentes algoritmos de búsqueda y ordenamiento para determinar su eficiencia.

3. 2. Objetivos Específicos

- Implementar varios algoritmos de búsqueda y ordenamiento
- Estudiar el comportamiento de los diferentes algoritmos con respecto a un mismo vector
- Determinar, mediante análisis, cual de los algoritmos es el mas eficiente

4. Definición de Algoritmos Paralelos

Es un algoritmo que puede ejecutar multiples instrucciones de forma simultanea en diferentes dispositivos de procesamiento, luego combinar las salidas individuales para producir el resultado final.

5. Etapas de los Algoritmos paralelos

5. 1. Partición

El cómputo y los datos sobre los cuales se opera se descomponen en tareas. Se ignoran aspectos como el número de procesadores de la máquina a usar y se concentra la atención en explotar oportunidades de paralelismo.

En esta etapa se buscan oportunidades de paralelismo y se trata de subdividir el problema lo más finamente posible, es decir; que la granuralidad sea fina.

Un grano es una medida del trabajo Computacional a realizar.

Al particionar se deben tener en cuenta los siguientes aspectos:

- El número de tareas debe ser por lo menos un orden de magnitud superior al número de procesadores disponibles para tener flexibilidad en las etapas siguientes.
- Hay que evitar cómputos y almacenamientos redundantes; de lo contrario el algoritmo puede ser no extensible a problemas más grandes.
- Hay que tratar de que las tareas sean de tamaños equivalentes ya que facilita el balanceo de la carga de los procesadores.
- Considere alternativas de paralelismo en esta etapa ya que pueden flexibilizar etapas subsecuentes.
- El número de tareas debe ser proporcional al tamaño del problema. Así, se podrá resolver problemas más grandes cuando se tenga más procesadores. Es decir, que el algoritmo sea escalable.

5.2. Comunicación

En esta etapa se determina la comunicación requerida para coordinar las tareas y se definen estructuras y algoritmos de comunicación.

La comunicación requerida por un algoritmo puede ser definida en dos fases.

- Primero se definen los canales que conectan las tareas que requieren datos con las que los poseen.
- Segundo se especifica la información o mensajes que deben ser enviado y recibidos en estos canales.

En la etapa de comunicación hay que tener en cuenta los siguientes aspectos:

- Todas las tareas deben efectuar aproximadamente el mismo número de operaciones de comunicación. Si esto no se da, es muy probable que el algoritmo no sea extensible a problemas mayores ya que habrán cuellos de botella.
- La comunicación entre tareas debe ser tan pequeña como sea posible.
- Las operaciones de comunicación deben poder proceder concurrentemente.
- Los cómputos de diferentes tareas deben poder proceder concurrentemente.

5.3. Agrupamiento

El resultado de las dos etapas anteriores es evaluado en términos de eficiencia y costos de implementación. De ser necesario, se agrupan tareas pequeñas en tareas más grandes.

Esta etapa va de lo abstracto a lo concreto y se revisa el algoritmo obtenido tratando de considerar si es útil agrupar tareas y si vale la pena replicar datos y/o cómputos de acuerdo a la plataforma seleccionada.

Mediante la agrupación de tareas se puede reducir la cantidad de datos a enviar y así reducir el número de mensajes a transmitir y por ende el costo de comunicación.

En la etapa de agrupación se debe tener en cuenta los siguientes aspectos:

- Chequear si la agrupación redujo los costos de comunicación.
- Si se han replicado cómputos y/o datos, se debe verificar que los beneficios son superiores a los costos.
- Se debe verificar que las tareas resultantes tengan costos de computo y comunicación similares.
- Hay que revisar si el número de tareas es extensible con el tamaño del problema.
- Si el agrupamiento ha reducido las oportunidades de ejecución concurrente, se debe verificar que aun hay suficiente concurrencia y posiblemente considerar diseños alternativos.
- Analizar si es posible reducir aun más el número de tareas sin introducir des balances de cargas o reducir la extensibilidad.

5.4. Asignación

Cada tarea es asignada a un procesador tratando de maximizar la utilización de los procesadores y de reducir el costo de comunicación.

La asignación puede ser:

- Estática:una tarea es asignada a un procesador desde su inicio hasta su fin
- Dinámica: una tarea puede ser migrada durante su ejecución. Esto puede agregar un costo adicional

Aquí se trata de establecer el mayor número posible de tareas con la intensión de explorar al máximo las oportunidades de paralelismo.

En esta etapa se determina en que procesador se ejecutará cada tarea

6. Técnicas Algorítmicas Paralelas

6. 1. Divide y Conquistarás

El algoritmo divide un problema a resolver en subproblemas que son más fáciles de resolver que el problema original, resuelve subproblemas, combina soluciones de subproblemas y encuentra soluciones a las causas del problema.

El modelo divide y vencerás mejora la modularidad del programa y, a menudo, da como resultado algoritmos simples y eficientes. Por lo tanto, ha demostrado ser una herramienta poderosa para los diseñadores de algoritmos secuenciales.

6. 2. Aleatorización

Los números aleatorios se utilizan en algoritmos paralelos para permitir que el procesador tome decisiones locales. Esto tiene una alta probabilidad de que la decisión general sea adecuada. Aquí analizamos tres usos del caso.

6. 3. Técnicas de Puntero Paralelo

6. 3. 1. 1. Recorrido de Euler

es un camino a través de un gráfico en el que cada arco se recorre solo una vez. En los gráficos no dirigidos, cada borde suele sustituirse por dos bordes opuestos. El camino del árbol no se dirige alrededor del árbol, moviendo cada borde dos veces, una hacia abajo y otra hacia arriba. Al mantener una estructura conectada que representa la ruta de Euler a través del árbol, puede calcular muchas funciones en el árbol, como el tamaño de cada subárbol. Esta técnica utiliza trabajo lineal y profundidad paralela y es independiente de la profundidad del árbol. Las instrucciones de Euler se utilizan a menudo para reemplazar las instrucciones de árbol estándar, como las instrucciones de profundidad.

6. 3. 2. Contracción del grafo

el tamaño de un gráfico conservando parte de la estructura original. Por lo general, después de realizar una operación de reducción en un gráfico, el problema se resuelve de forma recursiva en el gráfico reducido. La solución al problema del gráfico de contrato se utiliza para formar la solución final. Por ejemplo, una forma de dividir un gráfico en componentes conectados es combinar primero varios vértices en vértices adyacentes para contraer el gráfico y luego encontrar los componentes relevantes del gráfico. Contrae y

eventualmente cancela la operación de contracción. Se pueden solucionar muchos problemas encogiendo el eje. En ese caso, esta técnica se conoce como contracción del eje.

6.3.3. Descomposición del oído

en dividir los bordes del gráfico en un conjunto ordenado de caminos. El primer segmento es el bucle y los otros segmentos se denominan orejas. El punto final de cada oreja se fija en la primera línea. Una vez que se encuentra el decaimiento del gráfico en el oído, no es difícil determinar si los dos bordes se encuentran en un ciclo común. Esta información se puede utilizar en algoritmos para determinar dos conexiones, tres conexiones, cuatro conexiones y planitud. La caries auricular paralela se puede encontrar utilizando trabajo lineal y profundidad de registro, independientemente de la estructura del trazado. Por lo tanto, esta técnica se puede utilizar para reemplazar la técnica de secuenciación estándar resolviendo estos problemas y buscando la profundidad primero.

7. Modelos de Algoritmos Paralelos

7.1. Paralelismo de datos

Aquí Las tareas son mapeadas de manera cuasi-estática. Las operaciones paralelas son muy similares. Ademas de esto, necesita poca sincronización y asignación.

7.2. Grafo de tareas

Generalmente se usa cuando tenemos grandes volúmenes de datos. Y Usualmente se mapean estáticamente

7.3. Conjunto de trabajadores (work pool)

Mapeo dinámico de tareas a procesos.Es muy útil para el balance de carga. En este el mapeo puede ser centralizado o descentralizado

7.4. Maestro-esclavo (master-slave)

Un maestro genera trabajo y los esclavos ejecutan. Ayuda a resolver problemas de cuellos de botella

7. 5. Productor-consumidor (pipeline producer-consumer)

Una secuencia de datos pasa a través de una serie de procesos. Aquí cada proceso realiza una tarea particular y todos los procesos actúan a la vez formando un pipeline