Programación Concurrente y Distribuida (CC65)

Trabajo Parcial 2024-1

Secciones: Todas
Profesores: Carlos Alberto Jara García

Instrucciones

- La tarea se desarrolla de manera grupal (grupos de 3 integrantes).
- Desarrollar uno de los algoritmos según indicación de su profesor.
- Se usará software para detección de plagio.
- Debe elaborar un video de máximo 5 minutos presentando los puntos más resaltantes de su tarea y dedicar el máximo tiempo posible a demostrar su dominio de los temas presentados.

Contexto

La programación concurrente se refiere a la capacidad de un sistema para realizar múltiples tareas simultáneamente, lo que es esencial en el contexto actual de desarrollo de software, donde la demanda de sistemas eficientes y escalables es alta. La combinación de programación concurrente con machine learning ofrece varios beneficios en la actualidad.

Estos beneficios se centran principalmente en el aprovechamiento eficiente del uso de goroutines para la mejora del rendimiento de las aplicaciones, así como, semáforos para sincronización entre goroutines usando sync.Mutex, sync.WaitGroup especialmente cuando se trabaja con grandes conjuntos de datos.

Descripción del Trabajo

El trabajo parcial consiste en desarrollar una aplicación que usando GO que implemente uno de los algoritmos siguientes (estudiados en la tarea académica 2) sin usar librerías externas, para resolver problemas específicos de la manera más eficiente posible, haciendo uso de mecanismos de paralelización y sincronización como semáforos**:**

- **Regresión Lineal**
- **K-Means**
- **Clasificación con Árboles de Decisión**

Rúbrica de calificación:

- ✓ **(2 puntos)** Implementar la versión del algoritmo de manera secuencial tradicional usando el lenguaje Go.
- ✓ **(6 puntos)** Implementar el algoritmo seleccionado de manera concurrente usando el lenguaje Go y métodos de sincronización como semáforos (goroutines, sync.Mutex, sync.WaitGroup), se calificará con 0 puntos si hace uso de librerías de terceros).
- ✓ **(5 puntos)** Las pruebas deben realizarse un número de veces, por ejemplo 1000 veces con arreglos de 1 millón de elementos, luego se debe calcular la media recortada es decir omitiendo los 50 menores y los 50 mayores tiempos.
- ✓ **(3 puntos)** Simular el algoritmo de forma simplificada usando promela.
- ✓ **(2 puntos)** Analizar el cumplimiento de exclusión mutua en este algoritmo simulado con

promela, usando assert y el procedimiento de análisis de spin.

- ✓ **(2 punto)** Presentación de la documentación completa (es excluyente para la calificación de los puntos anteriores).

Documentación:

1.- Presentar un informe en documento word conteniendo:

a) Carátula
b) Índice
c) Introducción
d) Explicación de los algoritmos (secuencial y concurrente) y la justificación del uso de los mecanismos de paralelización y sincronización utilizados.
e) Explicación de las pruebas realizadas, resultados y pegar las imágenes de evidencia.
f) Explicación de la simulación realizado con promela, pegar las imágenes de evidencia.
g) Explicación del análisis usando spin, pegar las imágenes de evidencia.
h) Bibliografía de ayuda usando APA.
i) Anexos:
    a. Enlace de github donde subió su código fuente y se pueda descargar (Se bajará 5 puntos menos si se evidencia que hubo actualizaciones en el repositorio luego de la fecha de presentación)
    b. Enlace de video de máximo 5 min presentando el funcionamiento de la aplicación (publicarlo en un repositorio en la nube para acceder y visualizar a través de la url).

Presentación:

Colocar como nombre del documento word su código de alumno (ejemplo 201616054.docx) finalmente subir el archivo al aula virtual.

OJO: Cada integrante del grupo sube su archivo de forma individual.

Santiago de Surco, abril 2024