**PROGRAMACIÓN PARALELA**

En muchas aplicaciones actuales, el software necesita tomar decisiones rápidamente. Y la mejor manera de hacerlo es a través de la programación paralela en C/C++ y multihilo (programación multihilo).

En la clase de hoy vamos a estar explicando qué es la programación en paralelo, subprocesos múltiples (programación multiproceso), concurrente frente a paralelo, y cómo evitar los defectos de programación en paralelo de C/C++.

**¿Qué es la programación paralela?**

*La programación paralela es el proceso de utilizar un conjunto de recursos para resolver un problema en menos tiempo dividiendo el trabajo.*

El uso de la programación paralela en C es importante para aumentar el rendimiento del software.

**Concurrente vs Paralelo: ¿En qué se diferencia la programación paralela de la programación multiproceso?**

La programación paralela es un concepto amplio. Puede describir muchos tipos de procesos que se ejecutan en la misma máquina o en diferentes máquinas.

Multithreading se refiere específicamente a la ejecución simultánea de más de un conjunto secuencial (hilo) de instrucciones.

*La programación de subprocesos múltiples es la programación de múltiples subprocesos de ejecución concurrentes. Estos subprocesos podrían ejecutarse en un solo procesador. O podría haber varios subprocesos ejecutándose en varios núcleos de procesador.*

**Concurrente vs Paralelo: Programación multiproceso en un solo procesador**

Multithreading en un solo procesador da la ilusión de ejecutarse en paralelo. En realidad, el procesador está cambiando usando un algoritmo de programación. O bien, está cambiando en función de una combinación de entradas externas (interrupciones) y cómo se han priorizado los subprocesos.

**Concurrente vs Paralelo: Programación multiproceso en múltiples procesadores**

Los subprocesos múltiples en múltiples núcleos de procesador son verdaderamente paralelos. Los microprocesadores individuales trabajan juntos para lograr el resultado de manera más eficiente. Hay múltiples tareas paralelas y concurrentes que suceden a la vez.

**¿Por qué es importante la programación multiproceso?**

Los subprocesos múltiples son importantes para los equipos de desarrollo en la actualidad. Y seguirá siendo importante a medida que la tecnología evolucione.

Este es el por qué:

**Los procesadores están a la máxima velocidad de reloj**

Los procesadores han alcanzado la velocidad de reloj máxima. La única forma de sacar más provecho de las CPU es con el paralelismo.

Los subprocesos múltiples permiten que un solo procesador genere múltiples subprocesos simultáneos. Cada subproceso ejecuta su propia secuencia de instrucciones. Todos acceden al mismo espacio de memoria compartida y se comunican entre sí si es necesario. Los subprocesos se pueden administrar cuidadosamente para optimizar el rendimiento.

**El paralelismo es importante para la IA**

A medida que alcanzamos los límites de lo que se puede hacer en un solo procesador, se ejecutan más tareas en múltiples núcleos de procesador. Esto es particularmente importante para la IA.

Un ejemplo de esto es la conducción autónoma . En un automóvil tradicional, se confía en los humanos para tomar decisiones rápidas. Y el tiempo de reacción promedio para los humanos es de 0,25 segundos .

Entonces, dentro de los vehículos autónomos, la IA debe tomar estas decisiones muy rápidamente, en décimas de segundo.

El uso de subprocesos múltiples en C y la programación paralela en C es la mejor manera de garantizar que estas decisiones se tomen en el marco de tiempo requerido.

**Los lenguajes C/C++ ahora incluyen bibliotecas multiproceso**

Pasar de programas de subproceso único a multiproceso aumenta la complejidad. Los lenguajes de programación, como C y C++, han evolucionado para facilitar el uso de múltiples subprocesos y manejar esta complejidad. Tanto C como C++ ahora incluyen bibliotecas de subprocesos.

El C++ moderno , en particular, ha recorrido un largo camino para facilitar la programación paralela. C ++ 11 incluía una biblioteca de subprocesos estándar. C++17 agregó algoritmos paralelos e implementaciones paralelas de muchos algoritmos estándar.

Se espera soporte adicional para el paralelismo en futuras versiones de C++ .