

Programación Concurrente

La programación concurrente es un paradigma esencial en la creación de aplicaciones eficientes y reactivas en sistemas modernos, permitiéndole que múltiples tareas se ejecuten de manera simultánea, mejorando tanto la capacidad de respuesta como el rendimiento del sistema.

Se abordará la diferencia entre conceptos estrechamente relacionados: Computación concurrente y la Computación paralela. Aunque ambas implican la ejecución de diferentes tareas o procesos, abordan esta ejecución de manera distinta, con objetivos y enfoques específicos.

Proceso en Programación Concurrente

En programación concurrente, un proceso es una unidad ejecutable independiente que representa la ejecución de un programa o una parte de este. Cada proceso tiene su propia copia de memoria, recursos y contexto de ejecución. Los procesos pueden ejecutarse en paralelo, lo que significa que múltiples procesos pueden estar actuando al mismo tiempo y ser ejecutados por múltiples núcleos del CPU o hilos.

Cada proceso tiene su propio ciclo de vida, que generalmente es una secuencia de varios estados como: creado, listo, en ejecución, bloqueado y terminado. La comunicación de procesos es la acción de compartir la CPU de un proceso a otro. Los sistemas operativos administran las operaciones de procesos asegurándose de que los recursos se asignen adecuadamente y que los procesos se ejecuten de manera justa y eficiente.

Ejemplo de Proceso

Imagina que ejecutas un navegador web y un reproductor de música como procesos separados (S.O.). Ambos procesos pueden estar activos al mismo tiempo permitiéndote navegar por internet mientras que escuchas música. Cada proceso tiene su propia asignación de recursos y se ejecuta en su propia espacio de memoria, lo que evita que un proceso afecte directamente a otro.

Concurrencia en Programación Concurrente

La concurrencia se refiere a la capacidad de un sistema para manejar múltiples tareas de manera simultánea o en rápida sucesión. En programación concurrente, la concurrencia se logra mediante la ejecución de múltiples procesos o hilos. Los procesos o hilos pueden ejecutarse en paralelo (en sistemas con múltiples núcleos) o mediante la alternancia rápida entre ellos (en sistemas con solo un núcleo), lo que crea la ilusión de ejecución simultánea.

La concurrencia permite que múltiples tareas se ejecuten al mismo tiempo, lo que puede aumentar la eficiencia y la capacidad de respuesta del sistema. Sin embargo, también introduce desafíos como la gestión de recursos compartidos y la sincronización para evitar problemas como las condiciones de carrera y bloqueos.

Ejemplo de Concurrencia

Imagina que estás desarrollando una aplicación de edición de imágenes que permite aplicar filtros y ajustes en tiempo real, mientras que se carga una imagen. Puedes utilizar la concurrencia para que la aplicación cargue la imagen de un hilo, mientras otro hilo se encarga de aplicar los filtros en paralelo. De esta manera, el usuario experimenta una interfaz más fluida y receptiva, ya que la carga y el procesamiento no se bloquean entre sí. Sin embargo, es importante sincronizar correctamente los hilos para evitar problemas al dar al hilo e interfaz acceso a la imagen que todavía está cargando.

Computación Concurrente

La programación concurrente se refiere a la ejecución de múltiples tareas o procesos en un sistema, donde estos procesos pueden comenzar, ejecutarse y finalizar en momentos diferentes. El objetivo principal de la programación concurrente es mejorar la capacidad de respuesta y la eficiencia de una aplicación al permitir que diferentes tareas se ejecuten de manera intercalada, aunque estas tareas no se ejecuten necesariamente al mismo tiempo, se alternan rápidamente en la ejecución, creando la ilusión de una ejecución simultánea.

En la computación concurrente, los procesos a menudo interactúan entre sí. Pueden compartir recursos o comunicarse mediante mecanismos de sincronización. Sin embargo, esta interacción puede dar lugar a desafíos como condiciones de carrera y bloqueos, que requieren una gestión cuidadosa.

Computación Paralela

Se centra en la ejecución de simultánea real de múltiples tareas o procesos en diferentes unidades de procesamiento. El objetivo principal radica en acelerar la ejecución de una tarea al dividirla en partes más pequeñas que pueden ejecutarse simultáneamente en diferentes núcleos del CPU o en máquinas separadas en una red. La computación paralela requiere de hardware específico, como procesadores multicore o sistemas cluster. En la computación paralela, cada tarea se divide en subprocesos o subproblemas independientes, los procesos se ejecutan en paralelo lo que puede resultar en una mejora en velocidad.

de procesamiento y rendimiento. Sin embargo, la computación paralela también presenta desafíos como necesidad de configurar y sincronizar los subprocesos para garantizar resultados correctos y precisos.

Diferencias clave entre C. Concurrente y C. Paralela

- Enfoque:** La computación concurrente se centra en administrar de manera eficiente múltiples tareas intercaladas, mientras que la computación paralela se enfoca en la ejecución simultánea real de tareas divididas en subprocesos independientes.
- Ejecución:** La computación concurrente crea la ilusión de ejecución alterando rápidamente entre tareas, mientras que la computación paralela implica la ejecución real en el hardware paralelo.
- Objetivo:** La computación concurrente tiene como objetivo mejorar la eficiencia y la capacidad de respuesta del sistema. La computación paralela busca aumentar la velocidad de procesamiento y el rendimiento general de las tareas.
- Interacción:** La computación concurrente implica la interacción y la comunicación entre procesos concurrentes, en la computación paralela los subprocesos pueden estar más aislados y requieren menos comunicación.

Características de la Programación Concurrente

► **Ejecución Simultánea:** La programación concurrente permite la ejecución simultánea de múltiples tareas o procesos. Estos procesos pueden ejecutarse en diferentes núcleos del CPU o hilos, aprovechando al máximo de los recursos disponibles.

► **Mejora de la eficiencia:** Una de las principales ventajas de la P.C. es mejorar la eficiencia y el rendimiento del sistema. Al ejecutar tareas en paralelo se reduce el tiempo total de la ejecución.

► **Capacidad de Respuesta:** La computación concurrente aumenta la capacidad de respuesta de las aplicaciones al permitir que los procesos se ejecuten de manera intercalada. Esto significa que una aplicación puede responder a las solicitudes del usuario mientras se realizan otras tareas en segundo plano.

► **Descomposición de Problemas:** Mediante la P.E.A. con los problemas se pueden dividir en partes más pequeñas que se pueden abordar de manera independiente. Cada parte puede resolverse en un proceso o hilo separado, facilitando la solución de problemas complejos.

► **Comunicación y Sincronización:** Los procesos concurrentes a menudo necesitan comunicarse y sincronizarse entre sí. Las técnicas de comunicación, como los colas y los canales, permiten compartir datos y coordinar actividades.

► **Recursos Compartidos:** Los procesos concurrentes pueden compartir recursos como memoria y archivos. Sin embargo, el acceso simultáneo de recursos compartidos puede dar lugar a problemas como condiciones de carrera, por lo que se requiere sincronización adecuada.

► **Paralelismo y Distribución:** La programación concurrente abreca tanto para sistemas multi-core como la distribución de sistemas distribuidos. Pueden abarcar desde aplicaciones en una sola máquina hasta aplicaciones distribuidas en múltiples servidores.

► **Gestión de la Concurrencia:** La programación concurrente implica lidiar con desafíos específicos de concurrencia, como condiciones de carrera, bloques y dead lock. Se requieren técnicas cuidadosas para evitar estos problemas.

► **Modelos de Programación:** Se utilizan diferentes modelos de programación concurrente, como hilos, procesos ligeros, actores basados en eventos y modelo de actores. Cada modelo tiene sus propias ventajas y desventajas.

► **Dificultad y Complejidad:** La programación concurrente puede ser más compleja que la programación secuencial. Los errores pueden ser difíciles de depurar y reproducir debido a la naturaleza intermitente de la concurrencia.

► **Planificación de Procesos:** Los sistemas operativos desempeñan un papel crucial en la programación concurrente al administrar la planificación y asignación de recursos de los procesos. La gestión de prioridades y la contención de procesos son aspectos fundamentales.