Universidad de Huelva

Universidad de Huelva

Grado en Ingeniería Informática

Tema 1. Introducción a la programación concurrente

Resumen

Autor: Alberto Fernández Merchán
Asignatura: Programación Concurrente y Distribuida

1. Introducción

Se llama concurrencia a el transcurso simultáneo de varias circunstancias. La programación concurrente surge con la multiprogramación.

La programación concurrente toma relevancia con tres hitos:

- Aparece el concepto de hilo.
- Aparece Java que da soporte a la concurrencia.
- Aparece Internet.

2. Conceptos de la programación concurrente

Dos procesos serán concurrentes cuando la primera instrucción de uno de ellos se ejecute después de la primera de otro y antes de la última.

Los procesos concurrentes pueden colaborar y competir por los recursos. Para ello hacen falta mecanismos de comunicación y sincronización entre los procesos (Programación Concurrente).

En la programación paralela, las instrucciones de dos o más procesos se ejecutan a la vez.

3. Ventajas y desventajas de la programación concurrente

Ventajas

- Diseño más simple de las aplicaciones.
- Mayor velocidad de ejecución.
- Soluciona problemas concurrentes.
- Mejor tiempo de respuesta de las aplicaciones.

Desventajas

- Diseño más complejo de las aplicaciones (en otros casos).
- Sobrecarga en los cambios de contexto.
- Incremento del consumo de recursos.

4. Arquitecturas Hardware para la concurrencia

4.1. Sistemas monoprocesador

Hacen uso de la multiprogramación o del tiempo compartido para la ejecución concurrente de procesos. Todos los procesos comparten la misma memoria. La comunicación puede hacerse mediante memoria compartida.

4.2. Sistemas multiprocesador

Puede haber paralelismo. Se pueden clasificar en:

4.2.1. Sistemas fuertemente acoplados

Los procesadores y otros dispositivos como la memoria están conectados a un bus. Esto permite que todos los procesadores puedan comartir la misma memoria.

4.2.2. Sistemas débilmente acoplados

Cada procesador tiene su propia memoria local y está conectado con otros procesadores mediante un enlace de comunicación. (Sistemas distribuidos).

Se utiliza el paso de mensajes como mecanismo de comunicación y sincronización.

5. Procesos vs. Hilos

5.1. Proceso

Un proceso es un programa en ejecución o una instancia de este. Es una entidad dinámica.
Son entidades pesadas. Los cambios de contexto son costosos.

5.2. Hilo

Un hilo puede definirse coo cada secuencia de control dentro de un proceso que ejecuta sus instrucciones de forma independiente.
Son entidades ligeras. Los cambios de contexto entre hilos consumen poco tiempo de procesador.

6. Especificaciones de ejecución concurrente

6.1. Condiciones de Bertstein

Sea $L(S_k) = \{a_1, a_2, ..., a_n\}$ El conjunto de lectura del conjunto de instrucciones S_k formado por las variables que son leídas durante la ejecución de S_k.

Sea $E(S_k) = \{b_1, b_2, ..., b_n\}$ El conjunto de escritura del conjunto de instrucciones S_k formado por las variables que son escritas durante la ejecución de S_k.

Para que dos conjuntos de instrucciones S_i y S_j puedan ejecutarse concurrentemente debe cumplirse:

- $L(S_i) \cap E(S_j) = \emptyset$
- $E(S_i) \cap L(S_j) = \emptyset$
- $E(S_i) \cap E(S_j) = \emptyset$

6.2. Grafos de precedencia

Es un grafo acíclico dirigido. Cada nodo representa un conjunto de instrucciones. Una flecha de A hacia B representa que A debe ejecutarse antes que B.

7. Características de los sistemas concurrentes

7.1. Orden de ejecución de las instrucciones

- Orden total: Dado un conjunto de datos de entrada, se conoce el flujo del programa.
- Orden parcial: No se puede saber el flujo de ejecución.

7.2. Indeterminismo

Debido al orden parcial, los programas pueden devolver diferentes resultados al ejecutarse sobre el mismo conjunto de datos.

8. Problemas inherentes a la programación concurrente

8.1. Exclusión Mutua

La ejecución concurrente se produce entre las instrucciones de bajo nivel. Estas pueden ejecutarse en cualquier orden y devolver un resultado erróneo.

La sección de código que no puede ejecutarse de forma concurrente es la sección crítica.

Estas secciones críticas se deben ejecutar en exclusión mutua.

8.2. Condición de sincronización

En ocasiones los procesos no pueden hacer uso de un recurso compartido hasta que no se encuentre en un determinado estado. La programación concurrente proporciona mecanismos para bloquear procesos que no pueden hacer algo hasta que un evento los desbloquee.

9. Corrección de programas concurrentes

Para que un programa concurrente sea correcto debe satisfacer unas propiedades inherentes a la concurrencia:

9.1. Propiedades de seguridad

Aseguran que nada malo va a pasar durante la ejecución del programa.

9.1.1. Exclusión mutua

Hay recursos del sistema que deben accederse en exclusión mutua. Hay que garantizar que si un proceso adquiere el recurso, otros procesos deberán esperar a que sea liberado.

9.1.2. Condiciones de sincronización

Cuando un proceso debe esperar a que ocurra un evento para seguir ejecutándose, hay que garantizar que el proceso no continua hasta que no se produzca dicho evento.

9.1.3. Interbloqueo

Se produce cuando los procesos esperan a que ocurra un evento que nunca se producirá.

9.2. Propiedades de viveza

Aseguran que algo bueno pasará eventualmente durante la ejecución del programa.

9.2.1. Interbloqueo activo

Se produce cuando un sistema ejecuta una serie de isntrucciones sin progresar en la ejecución del programa.

9.2.2. Inanición

Se produce cuando el sistema hace progresos, pero existe un grupo de procesos que nunca progresan porque no se les otorga tiempo de procesador para avanzar.