Sistemas en Tiempo Real

Grado en Ingeniería Electrónica

UNIDAD 10: Planificación de sistemas de tiempo real

Capítulo 11 excepto 11.14, 11.15 y 11.16 de la edición Inglés (2009)

Capítulo 13, excepto punto 13.14 de la edición en Español (2003)

Tutoría

1. Modelo simple de procesamiento y enfoque de ciclo ejecutivo
2. Planificación basada en tareas
3. Test de planificabilidad basados en la utilización
4. Análisis del tiempo de respuesta para FPS
5. Tiempo de ejecución del peor caso
6. Tareas esporádicas y aperiódicas
7. Sistemas de tareas con D < T
8. Interacciones y bloqueos entre tareas
9. Protocolos de acotación de la prioridad
10. Un modelo extensible de procesamiento
11. Planificación EDF (Early Deadline First)
12. Sistemas dinámicos y análisis en línea

Introducción

En un programa concurrente, no es necesario especificar el orden exacto en el cual se ejecutan las tareas.

Sin embargo, un sistema de Tiempo Real necesita un esquema de planificación que elimine el no determinismo.

Este esquema implica un algoritmo para ordenar el uso de los recursos (sobre todo la CPU) y un test de planificabilidad que confirme que se pueden satisfacer los requisitos temporales del sistema.

Modelo simple de procesamiento

Es un modelo básico que permite describir algunos esquemas de planificación estándar:

- La aplicación está compuesta por un conjunto fijo de tareas.
- Las tareas son periódicas, con periodos conocidos.
- Las tareas son independientes entre sí.
- El instante crítico se produce cuando todas las tareas son ejecutadas a la vez.
- Las sobrecargas del sistema, tiempos de cambio de contexto y demás, se suponen con coste cero.
- Los tiempos límite de las tareas son iguales a sus periodos.
- Las tareas tienen tiempo de ejecución constante en el peor caso.

Modelo simple de procesamiento

Para definir este modelo se utilizan las siguientes variables:

Notación	Descripción
B	Tiempo de bloqueo en el peor caso
C	Tiempo de ejecución de la tarea en el peor caso (WCET)
D	Tiempo límite de la tarea
I	Tiempo de interferencia de la tarea
J	Fluctuación en la ejecución de la tarea
N	Número de tareas en el sistema
P	Prioridad asignada a la tarea
T	Tiempo mínimo entre ejecuciones de la tarea (periodo)
Y	Utilización de cada tarea (C/T)
a-z	Nombre de la tarea

Ciclo ejecutivo

Una forma común de la implementación de sistemas de tiempo real es el uso de un ciclo ejecutivo, que es apropiado para sistemas periódicos simples.

El ciclo ejecutivo es una **tabla de llamadas a procedimientos**, donde cada procedimiento representa parte del código de un proceso, con un ciclo principal y ciclos secundarios de duración fija.

Propiedades:

- En tiempo de ejecución no existen tareas reales, sino llamadas a procedimientos.
- Los procedimientos comparten un espacio de direcciones, pueden compartir datos sin necesidad de protección porque no se permite acceso concurrente.
- Los periodos deben ser múltiplos del tiempo de ciclo secundario.

Inconvenientes:

- Dificultad para incorporar tareas esporádicas.
- Dificultad para incorporar tareas con periodos grandes, el tiempo del ciclo principal es el único que se puede usar sin replanificación.
- Actividades con tiempo notable tendrán que ser divididas en procedimientos de tamaño fijo, por lo que es propenso a errores.
- Dificultad para construir el ciclo.

Si se puede construir un ciclo ejecutivo no será necesario ningún test de planificabilidad.

Planificación basada en tareas

Con el enfoque del ciclo ejecutivo, la ejecución consiste en una secuencia de invocaciones a procedimientos.

La noción de tarea no se preserva durante la ejecución. Una alternativa a este enfoque consiste en soportar de forma directa la ejecución de tareas y determinar cuál es la tarea que deberá ejecutarse en cada instante de tiempo mediante uno o más atributos de planificación.

Una tarea está limitada a estar en uno de los posibles estados siguientes:

- Ejecutable.
- Suspendida en espera de un evento temporizado (apropiado para tareas periódicas).
- Suspendida en espera de un evento no temporizado (apropiado para tareas esporádicas).

Alternativas de planificación:

- **FPS**: prioridad **estática**. Se asigna una prioridad fija a cada tarea antes de la ejecución. Las tareas se organizan por prioridad.
- **EDF**: primero la tarea que tiene el tiempo límite más cercano. Los tiempos límite se calculan en ejecución (**esquema dinámico**).
- **VBS**: basada en el valor. Asigna valores a las tareas para decidir cuál es la siguiente (**esquema adaptable**).

Planificación basada en tareas

En un esquema basado en **prioridad (FPS)**, es posible que una tarea de alta prioridad tenga que ser ejecutada mientras se está ejecutando otra de menor prioridad.

Hay tres posibilidades:

- Esquema **apropiativo**: se hace el cambio inmediato para la ejecución de la tarea prioritaria.
- Esquema **no apropiativo**: la tarea de menor prioridad puede finalizar.
- Esquema de **apropiación diferida o distribución cooperativa**: la tarea de menor prioridad puede continuar un tiempo limitado.

EDF y **VBS** también pueden tomar formas apropiativas y no apropiativas.

A partir del modelo simple de procesamiento, el esquema óptimo de asignación de prioridades se denomina **tasa monotónica**.

Básicamente, se asignan mayores prioridades a las tareas de periodos menores.

Test de planificabilidad basado en la utilización para FPS

Es muy simple, aunque no es exacto.

El test supone una condición suficiente pero no necesaria.

Si un conjunto de procesos pasa el test, entonces cumpliría todos los tiempos límite; si lo falla, puede o no fallar en tiempo de ejecución.

Este test basado en la utilización sólo aporta una respuesta del tipo sí/no. No da ninguna indicación sobre los tiempos de respuesta de las tareas.

Unidad de tiempo: pulso
N: número de tareas
T: Periodo
C: tiempo de ejecución (en el peor caso)

$$\sum_{i=1}^{N} \frac{C_i}{T_i} <= N(2^{\frac{1}{N}} - 1)$$

Planificación EDF (Primero el tiempo límite más temprano)

Test de utilización para EDF

- Primero el proceso con tiempo límite más cercano.
- Es más **simple**: Si la utilización del conjunto de tareas es menor que la capacidad total del procesador, entonces todos los tiempos límite se cumplirán.

$$\sum_{i=1}^{N} \frac{C_i}{T_i} <= 1$$

Unidad de tiempo: pulso
N: número de tareas
T: Periodo
C: tiempo de ejecución (en el peor caso)

Desventajas:

- Es más sencillo incorporar tareas sin tiempos límites en **FPS**.
- Es más fácil trabajar con prioridades (**FPS**) que con tiempos límite (deadlines) (**EDF**).
- **FPS** es más predecible en situaciones de sobrecargas.

Análisis del tiempo de respuesta para FPS

Los test de planificabilidad basados en la utilización para FPS tienen dos inconvenientes significativos:

- No son exactos.
- No son realmente aplicables a un modelo más general.

El test de tiempo de respuesta tiene dos etapas:

- Primero, se realiza una aproximación analítica **para predecir el tiempo de respuesta en el peor caso** para cada tarea.
- Después, estos valores son comparados con los tiempos límites de las tareas.

- Para la tarea de mayor prioridad, el tiempo de respuesta en el peor caso será igual a su tiempo de ejecución.
- El resto sufrirán cierta interferencia por parte de las tareas con mayor prioridad.

El cálculo del tiempo de respuesta es **suficiente y necesario** para la planificabilidad.

Por este motivo, este capítulo se centrará en este método.

Tiempo de ejecución en el peor caso

Para la planificación se supone que se sabe **el peor caso** del tiempo de ejecución de cada tarea, es decir, el máximo tiempo que puede requerir la invocación de esta.

La estimación de este tiempo se puede obtener por análisis o por medición.

El inconveniente del análisis es que se debe contar con un modelo efectivo del procesador.

El problema de la medición es demostrar que se ha tenido en cuenta el peor caso.

Procesos esporádicos y aperiódicos

Los tests anteriores se basan en un modelo simple con tareas periódicas.

Para poder incluir tareas esporádicas y aperiódicas hay que asignarles un valor T (período) y un tiempo límite (D).

El valor T se suele interpretar como el mínimo o la media de los periodos de las tareas periódicas.

El modelo simple supone que D = T, pero estas tareas esporádicas pueden requerir una respuesta inmediata, por lo que se debe permitir que D < T. (Lo que también será útil para procesos periódicos).

Con estos valores se puede aplicar el algoritmo del **tiempo de respuesta**.

Sistemas de tareas con D < T

En general, para cada tarea, debe ser posible definir un tiempo límite (D) menor que su periodo (T).

La ordenación de prioridad resulta óptima para un esquema de prioridad estática cuando D=T.

Para D < T se podía definir una ordenación de prioridades monotónica de tiempo límite (DMPO: deadline monotonic priority ordering), donde la prioridad fija de un proceso es inversamente proporcional a su tiempo límite: (Di < Dj => Pi > Pj)

Cualquier conjunto de procesos que pase el test de planificabilidad en prioridad estática (FPS) siempre cumplirá con sus requisitos de temporización si se ejecuta con tiempo límite (EDF).

Interacciones y bloqueos entre tareas

En un mundo ideal, una tarea nunca debería esperar por otra de menor prioridad.

En la realidad, no siempre es posible eliminar totalmente este efecto, aunque se pueden minimizar sus efectos adversos.

Si una tarea está esperando por otra de menor prioridad se dice que está bloqueada. Para poder probar la planificabilidad, el bloqueo debe estar limitado y ser ponderable (además de ser pequeño).

Protocolos de acotación de la prioridad

Son protocolos diseñados para minimizar las situaciones de cadenas de bloqueo y eliminar condiciones de fallo.

Cuando se utilizan estos protocolos en un sistema monoprocesador se cumplen las siguientes propiedades:

- Una tarea de alta prioridad puede ser bloqueada por tareas de prioridad baja una vez como mucho.
- Se previenen los bloqueos mutuos y transitivos.
- Se aseguran los accesos mutuamente excluyentes a recursos.

Un modelo extensible de procesamiento

El modelo visto hasta ahora es demasiado simple. Utiliza FPS con un enfoque apropiativo.

A continuación, se muestra cómo hacerlo más realista eliminando alguna de sus restricciones.

Planificación cooperativa

- El código de la aplicación se divide en bloques no desalojables (o apropiables), cuyos tiempos de ejecución están limitados. Al final de cada uno de estos bloques, se realiza una petición al núcleo del sistema para “desplanificar”. Si una tarea de alta prioridad está en estado ejecutable, el núcleo iniciará un cambio de contexto; si no, la tarea continuará con el siguiente bloque no desalojable.
- Es decir, una tarea continuará su ejecución hasta que se ofrezca como desplanificable. Por ello, se asegura la exclusión mutua siempre que cualquier sección crítica esté totalmente contenida entre dos invocaciones de desplanificación.

Un modelo extensible de procesamiento

Fluctuaciones en la activación

En el modelo simple, se supone que todos los procesos son periódicos y que van a ser activados con periodicidad perfecta (T).

Los procesos esporádicos se incorporan al modelo suponiendo que su intervalo mínimo entre llegadas es T. Sin embargo, esto no es siempre una suposición realista.

Para capturar correctamente la interferencia de las tareas esporádicas sobre las demás, se les debe asignar un valor máximo de variación en su activación (**fluctuación o jitter**).

Tiempos límite arbitrarios

Cuando el tiempo límite es menor (o igual) que el periodo, se considera una única activación de cada tarea. El instante crítico, cuando todas las tareas de alta prioridad se activen al mismo tiempo, representa la máxima interferencia y, por lo tanto, el peor caso del tiempo de respuesta debe darse en el instante crítico.

Sin embargo, cuando el tiempo límite es mayor que el periodo, se pueden considerar varias activaciones. Se supone que la activación de una tarea será retardada hasta que haya acabado cualquier ejecución anterior de la misma tarea.

Si el tiempo de respuesta en el peor caso (R) es mayor que el tiempo límite de la tarea (D), esa tarea no será planificable.

Un modelo extensible de procesamiento

Tolerancia a fallos

- La tolerancia a fallos mediante recuperación de errores implica cálculos extra.
- En sistemas de tiempo real tolerantes a fallos, los tiempos límite deberían cumplirse incluso ante ciertos niveles de fallo. Este nivel de tolerancia a fallos se conoce como modelo de fallos.
- Este modelo considera un máximo de un fallo, y supone que un proceso ejecutará su acción de recuperación con la misma prioridad que sus tareas normales.

Introducción de desplazamientos (overheads)

Existen grupos de tareas que se pueden beneficiar de la elección explícita de sus tiempos de activación, de forma que no compartan un instante crítico. (Asignar un desplazamiento con respecto a otras tareas).

El problema es que elegir los desplazamientos de forma que un conjunto de tareas sea óptimamente panificable es muy complejo.

MODULO 3: TEMPORIZACIÓN DE LAS TAREAS DE UN SISTEMA EN TIEMPO REAL

Este módulo explica los mecanismos de temporización que se pueden aplicar en un sistema de tiempo real, tomando como base el reloj del sistema en tiempo real. Una vez definidos los conceptos necesarios (timeouts, ámbitos temporales, etc.) se explica la forma de categorizar las tareas para su inclusión y priorización en la programación temporal del sistema. Una vez hecho esto, se pueden aplicar varias estrategias de planificación basadas en prioridades o tiempos de ejecución. Se termina el módulo mostrando los detalles específicos de programación sobre entornos de ejecución asociados a los mecanismos de entrada/salida de dichos entornos.

Foro de dudas del módulo 3 — Marcar como hecha

Dudas PED3

Dudas PED4

Autoevaluaciones de las unidades

Autoevaluación de la Unidad 9: Capacidades de tiempo real

Autoevaluación de la Unidad 10: Planificación de sistemas de tiempo real

Autoevaluación de la Unidad 11: Programación de bajo nivel

Ejercicios resueltos sobre el planificador de ciclo ejecutivo

ejercicio1.pdf

ejercicio2.pdf

Prácticas

PED3: Desarrollo de código RT

Prueba de evaluación a distancia (PED3): Desarrollo de código RT

En el fichero zip, están tanto el enunciado (PED3.pdf) como el ejemplo de implementación del PID (PID.pdf)

PED4: Real time sensor y AWS IoT (integración de código)

Prueba de evaluación a distancia (PED4): Integración del código RT

Leer el enunciado adjunto

Sistemas en Tiempo Real

Grado en Ingeniería Electrónica

UNIDAD 10: Planificación de sistemas de tiempo real

UNED

Nieves Pedreira Souto