Bisoños Usuarios de GNU/Linux de Mallorca y Alrededores | Bergantells Usuaris de GNU/Linux de Mallorca i Afegitons

Inversión de Prioridades, SCHED_IDLE y Linux (10673 lectures)

Per **Ricardo Galli Granada**, ***gallir*** (http://mnm.uib.es/gallir/)

Creado el 20/03/2001 00:11 modificado el 20/03/2001 00:11

*Hasta ahora Linus Torvalds era muy reacio a implementar el mecanismo de planificación SCHED_IDLE en Linux. Un proceso de este tipo sólo es ejecutado cuando no hay ninguna otra tarea o proceso en la cola de listos para ejecutar. Las razones para no implementarlo en Linux era el tratamiento de la **inversión de prioridades** que se pueden dar en estos sistemas. Parece que finalmente el mecanismo será implementado en el 2.5.*

Las inversión de prioridades provocan efectos indeseables en sistemas de multiprogramación cuando esos procesosde muy baja prioridad acceden a recursos compartidos con otros procesos de mayor prioridad.

La solución es hacer que el proceso que está en una sección crítica y bloqueando a otros procesos mediante un semáforo de exclusión mutua (*mutex*), tenga la prioridad igual a la mayor prioridad de los procesos bloqueados en ese semáforo.

El Problema de Inversión de Prioridades

Tenemos un sistema de multiprogramación con tres procesos activos:

- **A**: tiempo real, alta prioridad.
- **B**: calculos de media prioridad
- **C**: la tarea *IDLE* que sólo se ejecuta si no hay ningún proceso en la cola.

A y **C** acceden al mismo recurso por lo que se usa a un semáforo **mutex** para asegurar la exclusión mutua.

En un momento dado, **C** se está ejecutando y entra en la sección crítica mediante un *wait(mutex)*. Mientras **C** está en la sección crítica, ocurre un evento que hace que **A** sea asignado al procesador (*scheduled*), como **A** hace también un *wait(mutex)*, queda bloqueado esperando que **C** haga el *signal(mutex)*, pero resulta que antes que se ejecute **C** nuevamente, **B** se activa y como no hay otro proceso (recuerda que **C** sólo se ejecuta si no hay procesos en cola) se le asigna el procesador.

Bueno, ahora imagina que **B** hace varios segundos de cálculo (o sea tiene un *burst* largo). ¿Cual es el estado?

C: bloqueado esperando que no haya ningún proceso.
A: bloqueado esperando que **C** libere al *mutex*
B: ejecutándose tranquilamente como si **fuese el proceso de mayor prioridad**.

Esto es justamente **inversión de prioridades**. **B** está teniendo mas prioridad que **A** por efectos del bloqueo. Este efecto es muy malo para sistemas interactivos o de tiempo real. En el primer caso el usuario tendrá la sensación que el sistema es muy lento, en el segundo caso invalida toda la ejecución del proceso **A**.

En un sistema sin *SCHED_IDLE* (como Linux), **B** hubiera bajado de prioridad muy rápidamente hasta igualar a **C** y por lo tanto **C** hubiera podido obtener ciclos de procesador y liberar (señalizar) a *mutex*. Esto no ocurre con el mecanismo *SCHED_IDLE*.

El problema mencionado anteriormente ocurrió[1] en el Mars Pathfinder, aquél pequeño vehículo que paseaba por Marte sacando fotos y enviandolas a la Tierra. Menos mal que había dejado habilitado el debug en el sistema operativo

de forma que pudieron cambiar el comportamiento de los semáforos de exclusión mutua para que permita heredar la prioridad de otro procesos bloqueados en los semáforos.

Mas discusión del tema en Bulmailing[2].

--ricardo

Lista de enlaces de este artículo:

1. http://catless.ncl.ac.uk/Risks/19.49.html#subj1
2. http://bulma.net/pipermail/bulmailing/2001-March/000628.html

---

E-mail del autor: gallir _ARROBA_ uib.es