Práctica 4: Memoria virtual

2019 – Sistemas Operativos 2

Licenciatura en ciencias de la computación

Entrega: fecha a determinar

Nota: debe utilizar el Subversion de la materia creando un subdirectorio por alumno/grupo en:
`https://svn.dcc.fceia.unr.edu.ar/svn-no-anon/lcc/R-412/Alumnos/2019/`

1. Introducción

Para esta práctica no hay nuevos archivos, sin embargo de acuerdo a la constante `USE_TLB` definida en el `Makefile` del nuevo directorio de trabajo (`vmem`), se comienza a utilizar la TLB en lugar de las tablas de paginación manejadas por el procesador. Los programas de usuario `matmult` y `sort` son dos ejemplos de programas que hacen uso extensivo de la memoria virtual y se recomienda utilizarlos para probar el correcto funcionamiento de las implementaciones (además crear otros propios).

Algunos ejercicios requieren modificar clases del directorio `machine` (para obtener estadísticas por ejemplo). Se debe tener cuidado de no alterar la funcionalidad de esas clases.

2. Ejercicios

1. Implemente TLB. Detalles: cada espacio de direcciones tiene una tabla de páginas que ya no se actualiza automáticamente con el hardware. La única asistencia del hardware ahora es que cuando una entrada no figura en la TLB se dispara un `PageFaultException`, además si la entrada figura en la TLB pero es de solo lectura, se genera una `ReadOnlyException`. Debe implementar una rutina para solucionar estos inconvenientes.

   Para probar los programas de ejemplo debe ampliar el tamaño de la memoria física.

2. Calcule el porcentaje de aciertos ("hit ratio") de la TLB (al menos para dos programas razonablemente grandes). Vea cómo se modifica este porcentaje al agregar más entradas a la TLB (32 entradas = MIPS R2000, 64 = MIPS R4000). ¿Qué tamaño sugeriría para la TLB?

3. Implemente carga por demanda ("demand loading") pura. Suponga por ahora que los programas entran en memoria (agrándela de no ser así). Para esto, el constructor del espacio de direcciones no debe leer ninguna página de código o datos del ejecutable sino que estas se cargarán cuando se requiera acceder a esa página (fallo de página).

4. Implemente paginación utilizando la política de reemplazo más sencilla posible (FIFO o algo similar). Ahora se necesita un archivo para guardar las páginas que no entren en memoria y de esa forma proveer la ilusión de una memoria mucho más grande que la física. Cree un archivo con nombre `SWAP.asid` para cada espacio de direcciones, en

donde la página N del espacio de direcciones se pueda guardar en el bloque N del archivo. Debe mantener un mapa de la memoria (o "coremap") que indique a qué página virtual de qué proceso corresponde una página física.

Pruebe la implementación ejecutando un solo proceso, luego con procesos sucesivos y luego con procesos concurrentes (opción `-rs`).

5. Mejore la política de paginación (se recomienda ir introduciendo mejoras sucesivas). Implemente LRU o el algoritmo mejorado del reloj (o alguno que considere similar).

6. **Opcional pero interesante**
   Compare el rendimiento (cantidad de accesos a disco, fallos de páginas totales, etc.) entre los dos puntos anteriores y también contra el algoritmo óptimo (hágalo al menos con dos programas).