Práctica 4: Memoria virtual

2021 – Sistemas Operativos 2

Licenciatura en ciencias de la computación

Entrega: viernes 28 de mayo

1. Introducción

Véase el documento *Notas para la Plancha 4*.

2. Ejercicios

1. Implemente TLB. Para ello:

   a) Invalide el estado de la TLB en cada cambio de contexto, ya que la información que allí se encuentra deja de ser correcta en este caso.

   b) Reintente las lecturas y escrituras en caso de fallo, pues en un primer intento las traducciones no estarán disponibles en la TLB; y luego de que el sistema operativo maneje la excepción adecuada, podrán llevarse a cabo.

   c) Implemente el manejador de `PageFaultException` para que cada vez que la MMU no pueda realizar una traducción, el sistema operativo cargue allí la información necesaria para tal efecto. Para decidir qué entrada de la TLB reemplazar, puede utilizar un índice circular.

   d) Implemente el manejador de `ReadOnlyException`.

   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 cuando esté habilitada la bandera `DEMAND_LOADING`. 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 dé el primer acceso (que provocará un fallo de página).

4. Implemente intercambio de páginas a disco (*swap*) cuando esté habilitada la bandera `SWAP`. Ahora se usará un espacio en disco 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. Cuando la memoria se llene, el sistema operativo guardará algún marco en el disco y luego lo desalojará de la memoria. Nótese que una vez tomada la decisión de qué marco desalojar, será necesario conocer a qué página virtual de qué proceso corresponde.

a) Reemplace el mapa de bits de la memoria física por un *coremap*, una especie de tabla de paginación invertida. Con él no solo se podrá saber qué marcos físicos están libres, sino también a qué espacio de direcciones corresponde un marco ocupado y cuáles de sus páginas virtuales aloja.

b) Defina el espacio de intercambio en el disco. Cree un archivo con nombre `SWAP.`*`pid`* para cada proceso, donde *pid* sea un identificador del mismo. En próximos apartados, la página N del proceso se podrá guardar en el bloque N del archivo.

c) Como política de reemplazo, seleccione un número al azar. Implemente la política en una función `PickVictim`:

```
int PickVictim();
```

que devuelva un número de marco.

d) Implemente reemplazo de páginas. Cuando tenga que elegir qué marco físico desalojar, llame a la función del apartado anterior.

Tenga en cuenta que la MMU actualiza los bits de uso y modificación en la TLB. Cada vez que una entrada de la TLB sea reemplazada, se debe copiar dicha información a la tabla de página correspondiente. En particular esto ocurre en cada cambio de contexto y cuando se produce un fallo de página.

e) Agregue al núcleo estadísticas relevantes sobre el intercambio de páginas.

Pruebe la implementación ejecutando un solo proceso, luego con procesos sucesivos y luego con procesos concurrentes (opción `-rs`). Verifique que los valores de retorno sean correctos.

No debe modificar el contenido del directorio ***`machine`*** *excepto para probar distintos tamaños de RAM o TLB.*

5. Mejore la política de reemplazo de página establecida en la función `PickVictim`:

   a) Implemente FIFO cuando esté habilitada la bandera `PRPOLICY_FIFO`.

   b) Implemente LRU, el algoritmo mejorado del reloj o alguno que considere similar cuando esté habilitada alguna bandera correspondiente, como `PRPOLICY_LRU` o `PRPOLICY_CLOCK`.

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).