SIMULADOR DE ALGORITMOS DE REEPLAZO DE PÁGINAS
MEMORIA VIRTUAL

El código base pertenece al curso: Sistemas operativos 1 de Ciencias de la Computación de SIMON FRASER UNIVERSITY (SFU), adaptado para el curso INF239 Sistemas Operativos

El objetivo del presente laboratorio consistirá en implementar y comparar algunos de los algoritmos de reemplazo de página (Memoria Virtual) estudiados en clase. El código se encuentra en el archivo *vmsim.tar*

Para lograr este objetivo usted deberá seguir lo siguiente:

Pasos iniciales

1) Descargue y compile *vmsim* en un directorio llamado *~/lab4SO*:

```
$ tar -xvf vmsim.tar
$ cd vmsim
$ make
$ ./vmsim -h   (mostrará cómo ejecutar el simulador y los argumentos/opciones que toma)
```

2) Analice el código cuidadosamente:

a) *vmsim.c* – función principal, contiene algún código de inicialización, y `simulate()`. `simulate()` lee una referencia de memoria a la vez desde un archivo de trazo (***trace file***) que lo toma como entrada, busca la referencia en la tabla de página e invoca un manejador de fallos de página si este no es encontrado, y actualiza algunas estadísticas.

b) *fault.c* – manejador de fallos de página. Mucho del código a implementar se encontrará en este archivo. Un algoritmo de reemplazo "aleatorio" (`random`) se ha implementado como ejemplo.

c) *stats.c* – funciones que recolectan estadísticas.

d) *pagetable.c* – implementación de una tabla de página con múltiples niveles.

e) *physmem.c* – estructura para representar memoria física junto con algunas funciones, por ejemplo para cargar y desalojar páginas.

f) *options.c* – funciones para procesar argumentos de la línea de comandos.

g) *util.c* – algunas funciones útiles.

3) Lleve a cabo algunas ejecuciones de prueba con el algoritmo "aleatorio" (`random`) con algunos trazos simples, por ejemplo el archivo *example_trace.txt.zip* (desempaquételo para poder usarlo)

4) Ahora, empiece a implementar y probar los algoritmos de reemplazo uno por uno. Haga los cambios necesarios para cualquier archivo del código.

¿Cómo trabaja el simulador?

En el archivo *vmsim.c* la función principal (`main`) simula el trabajo de la unidad de memoria virtual: lee referencias de memoria desde un archivo de trazo (**trace file**) y chequea si esta referencia se encuentra en la memoria. Si está en memoria, algunas estadísticas se actualizan. Si no se encuentra en memoria, ocurre un fallo de página y el manejador de fallos de páginas es invocado. El manejador de fallos de página usa el algoritmo para desalojar una página de la memoria y hace espacio para la página solicitada. Muchos algoritmos de reemplazo de página pueden ser usados con el simulador. Usted debe especificar el algoritmo de reemplazo a ser usado cuando ejecuta el simulador (observe la salida de `./vmsim -h`).

El siguiente troza de código es tomado de *vmsim.c*

```
pte = pagetable_lookup_vaddr(vaddr_to_vfn(vaddr), type);
if (!pte->valid) {              /* Fault */
    stats_miss(type);           /* update some statistics */
    handler(pte, type);         /* call the page fault handler*/
}
```

Invocando a `handler()` se transfiere el control al manejador de fallos de página definido en ese momento. Por ejemplo, si se invocó `./vmsim random`, el control se transferirá a la función `fault_random()` en `fault.c`. Mucho (pero no todo!) del código a implementar estará en *fault.c*. Por ejemplo, para implementar el algoritmo LRU, usted debe añadir una nueva función llamada `fault_lru()` y actualizar el arreglo `fault_handlers[]`. Luego escriba el código correspondiente a `fault_lru()`. Observe que usted puede necesitar actualizar algunos campos que son necesarios por el algoritmo de reemplazo en el lazo `while` en la función `main` (*vmsim.c*). Usted necesita decidir qué campos actualizar y cuándo.

Otra ayuda que se le proporciona. La variable más importante que usted usará mucho es `physmem`, que es un arreglo de entradas de tabla de página (tipo `structure pte_t`). Cada entrada de `pte_t` tiene muchos campos. Tales como contador, modificado y referencia. Por ejemplo, `physmem[2]→reference = 0;` limpiará el bit de referencia de la página 2 en la memoria.

Métricas para medir el desempeño

a) Tasa de fallos de página : número de fallos de página/total de referencias de memoria
b) I/O overhead : número de páginas leídas del disco a memoria y número de páginas que han sido escritas de regreso al disco, porque fueron modificadas. Donde el número de páginas leídas ya es capturada por la tasa de fallos de página, se considera solo el número de páginas escritas al disco.
c) Memory overhead: memoria necesaria para mantener la tabla de páginas y la estructura del algoritmo de reemplazo.
d) Computation overhead: ciclos de CPU para ejecutar el algoritmo de reemplazo

Parámetros de entradas

a) Tamaño de la página: varía desde 128 bytes a 128 Kbytes
b) Algoritmo de reemplazo de páginas: Random, FIFO, LRU, etc (estudiados en clase)
c) Tamaño de memoria física: varía desde 1 Kbyte a 1 Mbyte
d) Archivo de trazo (**trace file**): archivo conteniendo las referencias de memoria hechas durante la ejecución de algunas aplicaciones (tales como *web browsers*).