UASLP
Universidad Autónoma
de San Luis Potosí

FACULTAD DE
INGENIERÍA

Práctica 3 . Paginación por demanda pura con algoritmo de reemplazo FIFO.

Sistemas operativos B.
Profesora: M.I. Ortiz Hernández Marcela.
grupo: 2402-03.
Semestre: 2019-2020/II.

Cantu Olivares Pedro de Jesus..

21/Mayo/2020.

1.- Resumen:

Objetivo:
Poder ejecutar el programa halt, matmult y sort en nachos implementando memoria virtual con paginación por demanda pura utilizando el algoritmo de reemplazo FIFO.

Características algoritmo FIFO (First In First Out):

- Algoritmo sencillo de reemplazo.
- Se reemplaza la página que haya llegado primero (la primer página en ser cargada en memoria es la página víctima).
- Presenta Anomalía de Belady.

2.- Instrucciones:

1. Tomando como base la práctica 2 implementar el algoritmo de reemplazo FIFO, siguiendo el procedimiento Pasos para el reemplazo de páginas
a) Atender la excepción de fallo de página y encontrar la ubicación de la página deseada en disco (archivo de intercambio)
b) Encontrar un marco libre: • Si hay un marco libre, usarlo • Si no hay un marco libre, usar el algoritmo de reemplazo FIFO para elegir la página víctima • Escribir la página víctima en disco(solo si fue modificada), actualizar tabla de páginas y de marcos c) Traer la página deseada al (nuevo) marco libre; actualizar las tablas de páginas y de marcos d) Continuar con la ejecución del proceso

2. Cuando se haya elegido una víctima con el algoritmo de reemplazo se deberá verificar si esta página fue modificada para respaldarla en su archivo de intercambio(swap out), en caso contrario, no se hace nada debido a que la información que está en memoria es idéntica a la del archivo de intercambio.

3. Se debe identificar la variable del SO que tiene la cuenta de las lecturas a disco duro e incrementarla cada que se realice un intercambio hacia adentro

4. Se debe imprimir en el resumen final de la máquina el número total de fallos de páginas, el número de lecturas y escrituras en disco al finalizar la

ejecución del programa(como se muestra), donde, a) el número de lecturas tiene que ser igual al número de fallos y, b) el número de escrituras tiene que ser menor al número de fallos, ya que solamente se deben escribir en disco aquellas páginas que hayan sido modificadas. matmult y sort deberán ejecutarse y finalizar correctamente, mostrando el resumen de la máquina como se muestra en el punto anterior.

5. La salida que tendrá que imprimirse en terminal es el vpn de cada página que causó el fallo de página separadas por comas.

3.- Liste todos los archivos que se modificaron para esta práctica y una breve descripción de modificación:

1.- archivo modificado: ./nachos/userprog/stats.h:
Descripción:
Definición de la variable para contar el marco a asignar.
Código:

```
  /*********************************
    Practica 3
    *********************************/
    int contadorMarco;
```

2.-archivo modificado: ./nachos/userprog/stats.cc:
Descripción:
Se agrego una variable para controlar el número de marco se se le asigna a una página que entra con fallo.
Código:

```
Statistics::Statistics()
{
    totalTicks = idleTicks = systemTicks = userTicks = 0;
    numDiskReads = numDiskWrites = 0;
    numConsoleCharsRead = numConsoleCharsWritten = 0;
    numPageFaults = numPacketsSent = numPacketsRecvd = 0;
    /****************************
    Practica 3
    ***********************/
    contadorMarco = 0;
}
```

3.-archivo modificado: ./nachos/userprog/addSpace.h:
Descripción:
Declaración del método para hacer swapOut.
Código:

```
/*****************************+
    Practica 3:
    *****************************/
    bool swapOut();
```

4.-archivo modificado: ./nachos/userprog/addSpace.cc:
Descripción:
Definicion del metodo para hacer swapOut, para a clase AddrSpace.
Código:

```
bool
AddrSpace::swapOut()
{
    int indicePagina = -1;

    for(int i = 0 ; i < numPages ; i++)
    {
        if(pageTable[i].physicalPage ==
stats->contadorMarco && pageTable[i].valid == TRUE)
        {
            indicePagina = i ;
            break;
        }
    }

    if(indicePagina <= -1)
    {
        printf("Pagina no encontrada\n");
    }
    else
    {
        printf("Haciendo swapOut a
%s\n",machine->swapFileName );
        if(pageTable[indicePagina].dirty)//la pagina esta
sucia.
        {
```

```
            OpenFile *swp =
fileSystem->Open(machine->swapFileName);   //abrimos el
archivo
            int direccionBaseDeMarco =
stats->contadorMarco * PageSize;
            if(swp == NULL)
            {
                printf("\nswabFile no abrio\n");
                return false;
            }
            else
            {
                printf("Escribiendo en la direccion %d de
la memoria principal\nTamaño de escritura: %d\nDesde la
direccion %d del archivo de
intercambio.\n",direccionBaseDeMarco,PageSize,indicePagin
a* PageSize);

swp->WriteAt(&(machine->mainMemory[direccionBaseDeMarco])
, PageSize, indicePagina* PageSize);
                stats->numDiskWrites++;
            }
            delete swp; //cerramos el archivo

        }

        pageTable[indicePagina].valid = FALSE;
        pageTable[indicePagina].dirty = FALSE;

        return true;
    }
    return false;
}
```

4. Tablas de resultados:

Halt/Marcos	Fallos	Lecturas	Escrituras
64	3	3	0
32	3	3	0
16	3	3	0
8	3	3	0
4	3	3	0

Matmult/Marcos	Fallos	Lecturas	Escrituras
64	47	47	0
32	110	110	48
16	8,960	8,960	1,090
8	23,375	23,375	5,208
4	82,981	82,981	17,611

Sort/Marcos	Fallos	Lecturas	Escrituras
64	40	40	0
32	3,309	3,309	2,879
16	10,954	10,954	8,367
8	19,467	19,467	10,821
4	1,589,336	1,589,336	540,722

5. Conclusiones

El algoritmo de remplazo FIFO es muy simple pero se puede complicar a medida que se reduce el número de marcos de la arquitectura.

UASLP
Universidad Autónoma
de San Luis Potosí

FACULTAD DE
INGENIERÍA

Práctica 4. Banderas de visualización.

Sistemas operativos B.
Profesora: M.I. Ortiz Hernández Marcela.
grupo: 2402-03.
Semestre: 2019-2020/II.

Cantu Olivares Pedro de Jesus..

21/Mayo/2020.

1.- Resumen:

Realizar la modificación de la práctica 3, Paginación por demanda pura con algoritmo de reemplazo de tal forma que cuando se ejecute un programa se pueda elegir la salida a visualizar en la terminal. Puntos a implementar

2.- Instrucciones:

1. Modificar la orden de ejecución(línea de comandos en terminal), agregando un argumento que especificará la salida a visualizar utilizar cuando se ejecute un programa, se muestra a continuación: La orden de ejecución de Nachos en su forma original es de la siguiente manera: $./nachos -x ../test/nombre_programa Se deberá realizar la siguiente modificación $./nachos -[C ó M ó F -o R] -x ../test/nombre_programa donde nombre_programa es halt, matmult o sort. Las opciones deben especificarse en ese orden. De esta manera, el usuario es capaz de seleccionar la salida que deberá mostrarse en la terminal.
2. Descripción de las nuevas opciones El argumento adicional puede ser la letra C, ó M, ó F, ó R mayúscula, depediendo del argumento que se especifique será la salida que se va a visualizar de la práctica. a) La opción -C indica que solo se va a visualizar como salida el tamaño del proceso y la cantidad de páginas que lo forman, además, del vpn de cada dirección lógica separadas por comas (es decir, se visualiza la cadena de referencia). b) La opción -M indica que se va a visualizar el mapeo de cada dirección lógica a física que generó fallo de página (dirección lógica, vpn, offset, dirección física) y el número de fallo. c) La opción -F indica que se va a visualizar como salida el tamaño del proceso, la cantidad de páginas que lo forman, además, del vpn de las direcciones lógicas que generaron fallo de página. d) La opción -R indica que solo mostrará el resumen de la máquina.
3. Es necesario validar los argumentos de entrada y marcar los errores en caso de que se presenten. Es decir, errores de sintaxis.

3.- Liste todos los archivos que se modificaron para esta práctica y una breve descripción de modificación:

1.- archivo modificado: ./nachos/userprog/main.cc:
Descripción:
Declaración de variable para la bandera que se usará para imprimir el flujo del programa.
Código:

```
/*****************************
Practica 4
******************/
char *comando_Fifo;//guarda el tipo de impresion segun el comando
```

2.-archivo modificado: ./nachos/threads/main.cc:
Descripción:
Se agrega una variable(comando_Fifo) externa que representa a la cadena para la bandera.
Código:

```
/***********************************
Prsctica 4 variable externa para los comandos agregados en la practica.
*******************************/
extern char* comando_Fifo;
```

3.-archivo modificado: ./nachos/threads/main.cc:
Descripción:
Entradas para las banderas(C,F,M,R) junto con el comando -x, para ejecutar.
Código:

```
#ifdef USER_PROGRAM
        if(!strcmp(*argv, "-C"))
        {
          ASSERT(argc > 2);
          if(!strcmp(*(argv+1), "-x"))
          {
               comando_Fifo = "-C";
               StartProcess(*(argv + 2));
               argCount = 3;

          }
          else
```

```
          {
               printf("\nSintaxis incorrecta, intenta
con:\n\t./nachos -C -x nombre_Programa.\n");
               interrupt->Halt();
          }

        }else if(!strcmp(*argv, "-M"))
        {
          ASSERT(argc > 2);
          if(!strcmp(*(argv+1), "-x"))
          {
               comando_Fifo = "-M";
               StartProcess(*(argv + 2));
               argCount = 3;
          }
          else
          {
               printf("\nSintaxis incorrecta, intenta
con:\n\t./nachos -M -x nombre_Programa.\n");
               interrupt->Halt();
          }

        }
        else if(!strcmp(*argv, "-F"))
        {
          ASSERT(argc > 2);
          if(!strcmp(*(argv+1), "-x"))
          {
               comando_Fifo = "-F";
               StartProcess(*(argv + 2));
               argCount = 3;
          }
          else
          {
               printf("\nSintaxis incorrecta, intenta
con:\n\t./nachos -F -x nombre_Programa.\n");
               interrupt->Halt();
          }

        }
        else if(!strcmp(*argv, "-R"))
        {
```

```
          ASSERT(argc > 2);
          if(!strcmp(*(argv+1), "-x"))
          {
               comando_Fifo = "-R";
               StartProcess(*(argv + 2));
               argCount = 3;
          }
          else
          {
               printf("\nSintaxis incorrecta, intenta
con:\n\t./nachos -R -x nombre_Programa.\n");
               interrupt->Halt();
          }
        }
        else if (!strcmp(*argv, "-x")) {          // run
a user program
         ASSERT(argc > 1);
          comando_Fifo = "-x";
            StartProcess(*(argv + 1));
            argCount = 2;
        } else if (!strcmp(*argv, "-c")) {      // test
the console
         if (argc == 1)
             ConsoleTest(NULL, NULL);
         else {
          ASSERT(argc > 2);
             ConsoleTest(*(argv + 1), *(argv + 2));
             argCount = 3;
         }
         interrupt->Halt();        // once we start the
console, then
                         // Nachos will loop forever
waiting
                         // for console input
     }
#endif // USER_PROGRAM
```

4.-archivo modificado: ./nachos/machine/translate.cc:
Descripción:
Definicion de la impresion segun la bandera en translate.cc en la clase Machine, en su método Translate(Machine::Translate()).

Código:

```
/*****************************
               Practica 4
               *****************************/
               if(!strcmp(comando_Fifo, "-M"))
              {
               printf("\nFallo #
%d\n",stats->numPageFaults +1);
                    printf("Direccion Virtual(Logica):
%d\n", virtAddr);
                    printf("offset(Desplazamiento)
calculado: %d\n",offset);
                    printf("vpn calculado: %d\n",vpn);
               }
               if(!strcmp(comando_Fifo, "-F"))
               {
                    printf("vpn -> Fallo# %d:
%d\n",stats->numPageFaults +1,vpn );
               }
```

5.-archivo modificado: ./nachos/userprog/addrSpace.cc:

Descripción:

Definicion de la impresion segun la bandera en addrSpace.cc en la clase AddrSpace y su constructor.

Código:

```
/************************************************
    Practica 0 :
    ************************************************/
    if(!strcmp(comando_Fifo, "-C") ||
!strcmp(comando_Fifo, "-F") )//Practica 4
    {
    printf("\nTamaño del proceso: %d
Bytes.\n",size);//imprime el tamaño del proceso.
    printf("\nNumero de paginas para el proceso:
%d.",numPages);//imprime el numero de paginas necesarias
para cargar el proceso.
    }
```

4. Conclusiones

Las variables externas funcionan para usarlas en varios archivos incluso si estos están en otro directorio. Es una forma inteligente el ejecutar los programas con banderas para especificar el modo que funcionara el programa y además tener mayor control sobre el mismo.