UNIVERSIDAD NACIONAL
COSTA RICA

Facultad de Ciencias Exactas y Naturales

Asignatura:

Sistemas Operativos

PROYECTO 1: MEMORIA

Profesor:

Eddy Miguel Ramírez

Estudiantes:

José Isaac Zeledón Jiménez
Jonathan Estrada Vargas

I CICLO

2019

Índice

1. Descripción del documento y del problema

En este documento se describe como se desarrolló el proyecto 1 de Sistemas Operativos en el que se refozaron los conceptos concernientes a la memoria y a la asignación de esta.
En este proyecto el problema inicia con el generador de calendarización de procesos el cual debe de recibir por consola

- Número de procesos
- Pid inicial
- Memoria mínima
- Memoria máxima
- Duración mínima
- Duración máxima
- Hora llegada inicial
- lambda media de Poisson

De estos datos se debía de generar los procesos que posteriormente serían utilizados para la calendarización en "memoria", todo esto se lograría con un programa en C linux, el cual como salida en la consola daría:

- Número o identificador único
- Unidad de tiempo que inicia
- Duración en unidades de tiempo
- Un tamaño en bytes

Con estos datos y utilizando el operador ">"de la consola de linux se debía de generar un archivo que guardara todos los procesos generados, para la simulación, este archivo es el que leería el módulo encargado de la calendarización, que además de poseer los datos de todos los procesos, debe leer un archivo de configuración con el cual podrá comportarse de la manera deseada ya que esta parte del proyecto requería que la simulación pudiera hacer la calendarización del recurso de la memoria en diferentes enfoques, más que todo para poseer una visualización del comportamiento de los diferentes algoritmos y formas de asignar memoria a procesos.

En este documento se describirá la manera en que abordamos el problema, los problemas que se afrontaron durante la realización del mismo, los algoritmos utilizados para cada uno de los modos de ejecución de la simulación y las conclusiones, las cuales van en función de los aprendizajes adquiridos a la hora del desarrollo de la simulación y además de los comportamientos observados durante la pruebas de las misma simulación. Con esto se refuerza de una forma visual los conceptos relacionados a la asignación de la memoria y la calendarización de procesos en esta.

2. Especificacion de la solución

Para la primera parte del proyecto se debía de utilizar una distribución de probabilidad la cual iban a seguir los procesos a la hora de su creación, esta distribución de probabilidad es la de Poisson.
Para esto se utilizó el cálculo de esta distribución de la probabilidad de Poisson que sigue esta fórmula:

$$\frac{e^{-lambda} lambda^{k}}{k!}$$

Esta fórmula se utilizó para que los procesos en el generador, se crearan, pero con esta distribución se debía de utilizar un concepto llamado tabla acumulada debido a que para calcular las horas de creación de los procesos se necesitaban valores en el eje de las abcisas osea enteros positivos, que estuvieran relacionados a una probabilidad de esta distribución. Para esto se utilizo el seguiente codigo brindado por el profesor del curso, mas adenlante en problemas encontrados se describirá como se intentó resolver pero no se tuvo éxito. Tabla acumulada:

```
long double * tablaPoisson;

void creaTabla(int lambda){
    tablaPoisson = calloc(sizeof(long double), 3*lambda);
    tablaPoisson[0] = exp(-1*lambda);
    int i =0; int ifactorial = 1;
    while(i++ <3*lambda){
        ifactorial *=i;
        tablaPoisson[i] = tablaPoisson[i-1]+
                          tablaPoisson[0]*
                          pow(lambda,i)/ifactorial;
    }
    tablaPoisson[i-1] =1.0;
}
```

Con el código anterior se generaba la tabla, que para un lambda n generaba una tabla de probabilidades desde $0 * lambda$ hasta $3 * lambda$ que era el rango máximo que el profesor especifico en el documento del proyecto.
Luego se utilizaba un método el cual para un número random cualquiera, se devolvía un entero que correspondía a un valor en en las abcisas de la distribución.
Codigo que retorna un valor en la distribución de Poisson:

```
int valorPoisson(long double r){
    int respuesta =0;
    while(r>tablaPoisson[respuesta++]);
    return respuesta-1;
}
```

2.1. Descripción de los algoritmos utilizados

1. 44

3. Descripción del manejo de la simulación

3.1. Modo Secuencial

3.1.1. FIFO

3.1.2. Best-fit

3.1.3. Worst-fit

3.2. Modo Paginación

4. Problemas encontrados

5. Conclusiones

- 22