Práctica

2

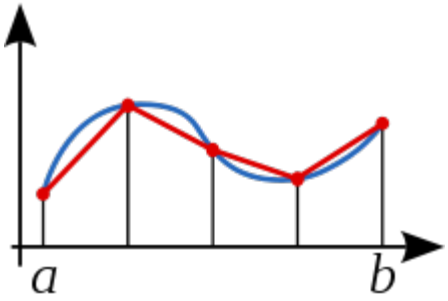
Sistemas Operativos

Procesos: creación, sincronización y comunicación

Enunciado

En esta práctica se pretende realizar un sistema capaz de calcular el área de una función (integral) en un intervalo determinado mediante métodos de cálculo numérico. Para ello, se va a dividir el problema en una serie de pasos y cada una de las partes será asignado a un proceso concreto, de forma que el resultado final se obtenga en base a lo que cada uno de ellos aporta a la solución final.

El método empleado para calcular la integral de una función consiste en reducir el problema a la suma del área de los trapecios que de forma aproximada componen la función. Cuantos más trapecios tengamos, menos error cometeremos en el cálculo final. En el ejemplo de la figura, se han utilizado 5 puntos (que se corresponden a 4 trapecios).



El ancho de cada uno de ellos es $w = \frac{b-a}{N}$

Por ejemplo, para el primer trapecio su área queda determinada por la fórmula:

$$AT_1 = \frac{f(a) + f(a+w)}{2} \times w$$

Sumando todas las superficies obtenidas, tendremos una buena aproximación al resultado final buscado.

El ejercicio planteado consiste en dividir el problema de cálculo descrito entre varios procesos, de forma que cada uno de ellos analice una parte de la función, y el resultado final se obtenga a partir de los resultados parciales calculados por cada proceso. Para ello, se debe realizar lo siguiente:

1. Crear un programa que se encarga de generar N_WORKERS procesos hijo. También define una función que será el objeto del cálculo de la integral, y la denominaremos "`calc_function`", la cual recibe como parámetro el valor "x" y devuelve el f(x) correspondiente. Consideraremos la función:
$$f(x) = x \cdot \sin\frac{25}{x}$$
(para el valor x=0, se tomará el valor f(0) = 1.0)
2. El proceso padre tiene dos pipes de lectura/escritura con cada uno de los hijos que ha creado. A cada hijo le va a pasar los datos que definen el trozo de la función que va a analizar, en concreto el valor de inicial de la función (`x_ini`), el ancho de la función a analizar (`w`) y el número de pasos a considerar (`pasos`).
3. Cada proceso hijo, cuando es creado, espera a recibir por la pipe de lectura los datos que el padre le envía (`x_ini, w, pasos`). El proceso hijo, con esos datos calcula el área de la función (`calc_function`) en el intervalo indicado (para lo cual subdivide el problema en el nº de trapecios indicado por la variable `pasos`), y además chequea si en ese intervalo la función cruza con el eje "y" (es decir, detecta un cero de la función). A través de la pipe de escritura le indica al padre el área calculada y el contador de ceros que ha registrado. A continuación, termina con el código de retorno del punto 7.
4. El proceso hijo, cuando detecta un cero también lanza la señal `SIGUSR1` al proceso padre.
5. El padre, cada vez que recibe una señal `SIGUSR1` incrementa el contador de ceros detectados.
6. El padre lee de la pipe todos los datos recibidos de cada proceso hijo, y calcula el valor de la integral de la función sumando todos ellos.

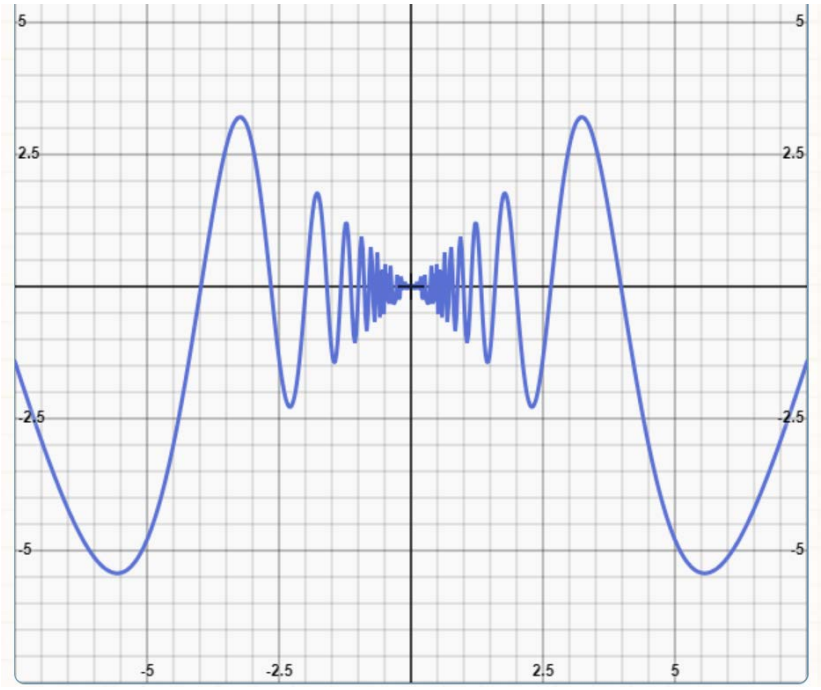
7. El padre espera la finalización correcta de cada uno de los procesos hijo e imprime la siguiente información: Nº de hijos, área de la función, nº de ceros (Los 3 valores separados por coma). El padre recibirá un -1, 0 ó 1 si el área calculada es negativa, cero o positiva.
8. Las constantes del programa deben ser establecidas con el fichero "`constantes.h`" como valores #define que faciliten su modificación.
9. El programa principal debe recibir como parámetros los valores `N_WORKERS` y los intervalos inicial y final del cálculo. Por ejemplo, se podría invocar el programa así:
   `%> calcula 100 -3.14159 3.14159`
   Para realizar el cálculo con 100 procesos hijo en el intervalo [-π, π].
10. Probar la ejecución del programa con diferentes parámetros. Considerar el cambio de la función por otra cuya integral conozcas de antemano para contrastar que el resultado obtenido es correcto.

Realización

A tener en cuenta:

1. Todo el trabajo debe estar en un directorio para esta práctica, denominado "`practica_2`".
2. Debes utilizar variables de tipo double en lugar de float para mejorar la precisión del cálculo realizado.
3. Debes construir un fichero makefile que facilite la compilación del programa.
4. Debes construir una macro que lance tu programa varias veces, modificando en cada ejecución el número de hijos que realizan el cálculo (empezando en 10 y acabando en 200, incrementando en 10 en 10). El resultado se debe ir añadiendo al fichero "`resultados.txt`". El intervalo de cálculo considerado será [-5,5].
5. Cuando lo tengas, debes entregar en un único fichero de formato `tar` lo realizado (explicación en comentarios, fuentes, datos, constantes, `make`, entrada, macro, …) a través de Moodle. Se te valorará la calidad del código, que todo funcione y la macro y `make` realizados, así como la documentación / comentarios y el fichero de resultados final "`resultados.txt`".

La función propuesta tiene este aspecto:



Plazo

Para esta práctica tienes una sesión de trabajo (dos horas presenciales y dos en casa), será parecida al examen posterior de esta parte y te servirá de referencia.