SISTEMAS OPERATIVOS

Práctica 3. Concurrencia

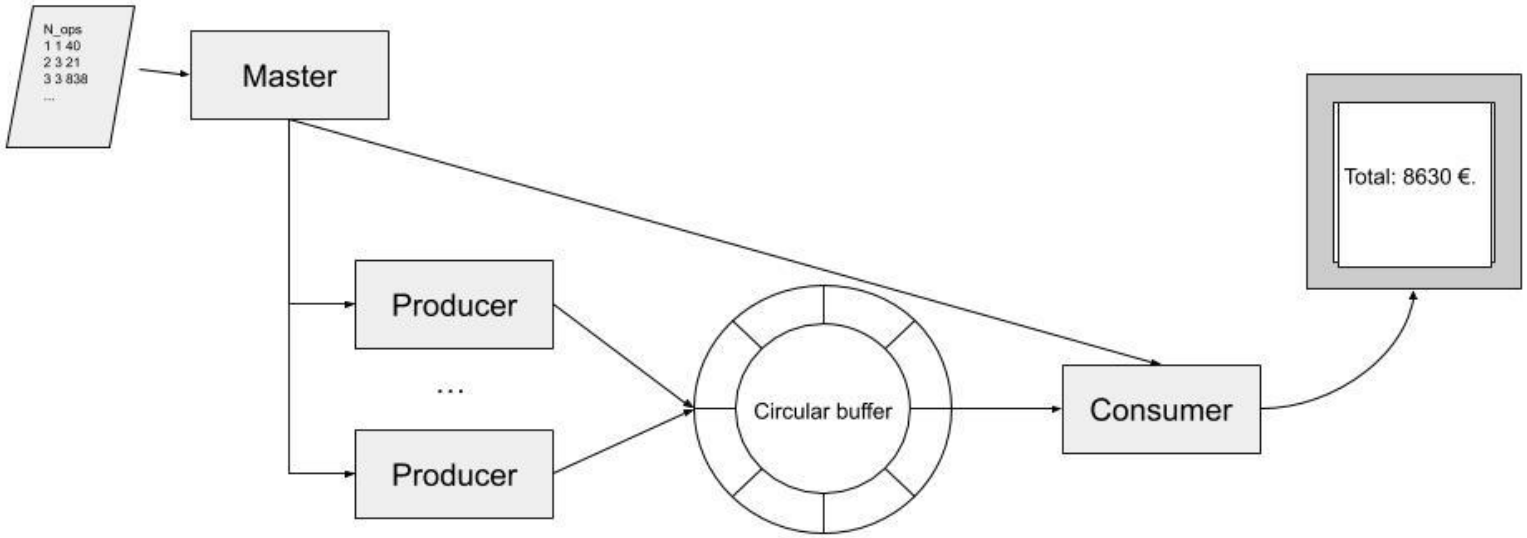
Presentación

- Descripción de la práctica
- Roles
- Buffer circular
- Concurrencia
- Entrada & Salida
- Corrector
- Entrega

Descripción

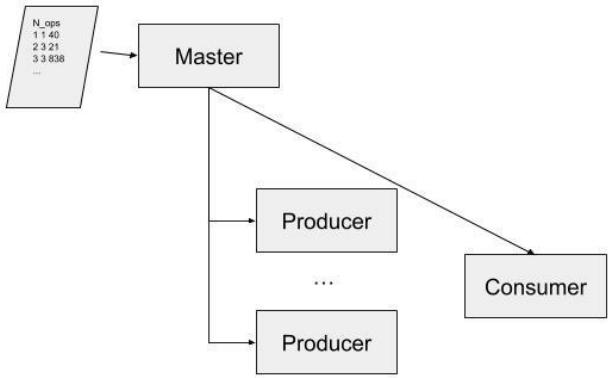
- Desarrollo de un sistema de cálculo de costes por el uso de máquinas en un centro de computación.
  - El usuario da un fichero con los tipos de máquinas que quiere utilizar, y el tiempo de uso de cada una (**fichero de entrada**).
  - El sistema debe proporcionar al usuario el presupuesto de la utilización de esos recursos (**salida**).
- Para el cálculo de costes:
  1. Cargar los datos del fichero en memoria.
  2. Iniciar un sistema N-productores – 1-consumidor
     1. Los productores insertan los datos de memoria al buffer circular compartido.
     2. El consumidor extrae los datos y calcula el coste de cada operación (acumulándolos para obtener el total).

Descripción



Roles

- Proceso principal:
  - Es el encargado de:
    - Inicializar las estructuras pertinentes (mutex, variables condición, arrays, …).
    - Obtener las operaciones indicadas en el fichero de entrada y almacenarlas en memoria (para posterior procesamiento).
    - Lanzar los hilos que componen el sistema Nprod-1cons.
    - Esperar la finalización de los hilos.
    - Mostrar el resultado por pantalla.



· *Se recomienda definir una estructura para cada operación y Almacenarlas en un array de estructuras (SoA).*

Roles

- Productor:
  - Puede haber desde 1 hasta *n* (valor indicado por parámetro).
  - Su función es:
    - Obtener las operaciones que le correspondan(·) de la estructura en memoria creada por el proceso principal.
    - Insertarlas en el buffer circular compartido.
    - Una vez que ha terminado de insertar las operaciones, finaliza su ejecución con *pthread_exit()*.

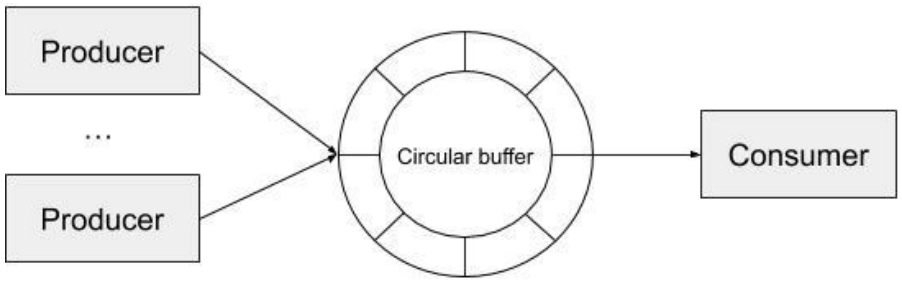
- *Se recomienda que el proceso principal realice un reparto de las operaciones a procesar.*

Roles

- Consumidor:
  - Sólo hay un consumidor.
  - Su función es:
    - Obtener las operaciones que hay insertadas en el buffer circular compartido.
    - Para cada una:
      - Obtener el coste de dicha operación.
      - Acumular ese valor obteniendo el total actualizado.
    - Una vez que ha terminado de procesar las operaciones, finaliza su ejecución con *pthread_exit()* **retornando el total calculado**.

Buffer circular

- Es la estructura que almacenará las operaciones que se quieren procesar.
- Será utilizada **al mismo tiempo** por productores y consumidor.
- Los accesos deben ser concurrentes.
- Funciones básicas definidas en *queue.c* y *queue.h*.
  - Completar estructura de operación y cola.



Concurrencia

- El control de la concurrencia debe realizarse utilizando **únicamente:**
  - Mutex
  - Variables condición
- Puede realizarse en dos lugares:
  - *Queue.c:* la concurrencia se gestiona directamente en el buffer.
  - *costCalculator.c:* la concurrencia es gestionada entre los threads que acceden a la cola
- Utilizar el método que resulte más sencillo para el grupo.
- Requisito:
  - *El resultado del cálculo con n-productores debe ser igual que el obtenido con 1-productor.*

Entrada & Salida

- Argumentos de entrada: *./cosCalculator <file> <prods> <bsize>*
  - *File*: fichero de entrada. Incluye número de operaciones a procesar y la lista de operaciones con el siguiente formato:

```
500          > num max operaciones que se deben procesar
1 1 4        > [id] [tipo máquina] [tiempo]
2 2 12
3 1 100
4 3 45
...          > puede haber más operaciones que num max operaciones
             > pero no menos
```

  - *Prods:* número de productores que se deben ejecutar.
  - *Bsize:* tamaño del buffer circular (número de elementos máximo que puede almacenar al mismo tiempo).
- Salida:

```
$> ./calculator input_file 5 10
Total: 13423 €.
$>
```

Corrector

- Se proporciona un corrector automático que ofrece una tentativa de nota sobre las pruebas funcionales.
- El corrector ejecuta pruebas básicas, luego el código pasará más ejemplos para su evaluación.
- Previo a la ejecución:
  - Comprimir en ZIP con el nombre correcto los ficheros solicitados.
- Ejecución:
  - *./corrector_ssoo_p3.sh ssoo_p3_<NIA>.zip*

Entrega

- **Viernes 8 de mayo 2020 (hasta las 23:55h).**
- Fichero zip con el nombre **ssoo_p3_AAA_BBB_CCC.zip** que contiene:
  - costCalculator.c
  - Queue.c y Queue.h
  - Makefile
  - Autores.txt
- Memoria: se entrega mediante turnitin (PDF).
- **Se deben seguir las normas incluidas en el enunciado. Leedlo cuidadosamente.**