CONCURRENCIA Y SECUENCIABILIDAD

14/NOV/24

1. Pruebas de escritorio para los Algoritmos Comunes
2. Revisar como la cola trabaja en cada Algoritmo.
3. Revisar como se va comportando la pila

ALGORITMO DE DEKKER

EJEMPLO DEL ALGORITMO:

→ Variables:

- flag[i]: Indica si un proceso esta interesado en entrar en ejecución.
- turn: Determina de quién es el turno

Iteración	Proceso	flag[0]	flag[1]	turn	
1	P0	true	false	0	P0 intenta estar en ejecución
2	P1	true	true	0	P1 intenta estar en ejecución
3	P1	true	true	0	P1 cede el paso a P0
4	P0	true	true	0	P0 se ejecuta y manda mensaje
5	P0	false	true	1	P0 cede el paso a P1
6	P1	false	true	1	P1 se ejecuta y manda mensaje
7	P1	false	false	0	P1 cede el paso a otro proceso

Cola: El productor inserta nodos al inicio de la lista (cabeza). El consumidor elimina el primer nodo disponible.

Pila de proceso: Cada función utiliza su propia pila para las llamadas y las variables locales. La pila se consume principalmente durante la creación de nodos y la llamada a funciones.

PRODUCTOR - CONSUMIDOR:

→ Variables:

- interesado[i]: Indica si el productor o consumidor quieren ejecutarse
- turno: Controla que proceso puede ejecutarse
- cabeza: Representa donde el productor inserta elementos y el consumidor los atiende.

Iteración	Acción	Interesado[0]	Interesado[1]	Turno	Cola
1	Productor interesa	1	0	0	Vacía
2	Productor interesa 1	1	0	1	1
3	Productor cede turno	0	0	1	1
4	Consumidor interesa	0	1	1	1
5	Consumidor atiende 1	0	1	0	Vacía
6	Consumidor cede turno	0	0	0	Vacía
7	Productor interesa	1	0	0	Vacía
8	Productor interesa 2	1	0	1	2
9	Productor cede turno	0	0	1	2
10	Consumidor interesa	0	1	1	2
11	Consumidor atiende 2	0	1	0	Vacía
12	Consumidor cede turno	0	0	0	Vacía

ALGORITMO DE PETERSON

EJEMPLO DEL ALGORITMO:

→Variables:

- flag[i]: Indica si un proceso desea entrar en ejecución.
- turn: Indica el turno del proceso a ejecutarse.

Iteración	Proceso	flag[Ø]	flag[1]	turn	
1	PØ	true	false	1	PØ quiere entrar, cede turno a P1
2	P1	true	true	1	P1 también quiere entrar, verifica turno
3	P1	true	true	1	P1 entra en ejecución y muestra mensaje
4	P1	true	false	Ø	P1 libera bandera y cede turno a PØ
5	PØ	true	false	Ø	PØ entra a ejecución y muestra mensaje
6	PØ	false	false	1	PØ libera bandera y cede turno a P1

PRODUCTOR- CONSUMIDOR:

→Variables:

- interesado[i]: Indica si el productor (Ø) o consumidor(1) quieren ejecutarse.
- turno: Controla de quién es el turno para ejecutarse.
- cabeza: Representa la cola de procesos.

→Inicialización:

- id_productor = 1
- id_consumidor = 2
- La prioridad se genera aleatoriamente entre 1 y 4.

Iteración	Acción	interesado[Ø]	interesado[1]	Turno	Cola
1	Productor interesa	1	Ø	1	Vacía
2	Productor genera proceso 10	1	Ø	1	10(p=2)
3	Productor cede turno	Ø	Ø	1	10(p=2)
4	Consumidor interesa	Ø	1	Ø	10(p=2)
5	Consumidor atiende proceso 10	Ø	1	Ø	Vacía
6	Consumidor cede turno	Ø	Ø	Ø	Vacía
7	Productor interesa	1	Ø	Ø	Vacío
8	Productor genera proceso 11	1	Ø	1	11(p=3)
9	Productor cede turno	Ø	Ø	1	11(p=3)
10	Consumidor interesa	Ø	1	1	11(p=3)
11	Consumidor atiende proceso 11	Ø	1	Ø	Vacía
12	Consumidor cede turno	Ø	Ø	Ø	Vacía

Cola: Implementada como una lista enlazada. El productor inserta nodos al inicio de la lista (cabeza). El consumidor elimina el primer nodo disponible.

Pila de proceso: Cada función (productor y consumidor) utiliza su propia pila para almacenar variables locales y llamadas a funciones (genera_proceso, free).

ALGORITMO DE LAMPORT'S BAKERY

EJEMPLO DEL ALGORITMO:

→Variables:

- choosing[i]: Indica si el proceso i esta eligiendo un número. (f=false, t=true)
- number[i]: Contiene el numero asignado al proceso i.
- NUM_PROCESOS: Numero total de procesos

Iteración	Proceso	choosing	number	
1	PØ	[t, f, f]	[1, Ø, Ø]	PØ elige su ticket: 1
2	P1	[t, f, f]	[1, 1, Ø]	P1 elige su ticket: 1
3	PØ	[f, t, f]	[1, 2, Ø]	P1 aumenta su ticket a 2, PØ finaliza acción
4	P2	[f, f, t]	[1, 2, 1]	P2 elige su ticket: 1
5	PØ	[f, f, f]	[1, 2, 1]	PØ entra a sección crítica
6	PØ	[f, f, f]	[Ø, 2, 1]	PØ libera su ticket
7	P2	[f, f, f]	[Ø, 2, 1]	P2 entra a sección crítica
8	P2	[f, f, f]	[Ø, 2, Ø]	P2 libera su ticket
9	P1	[f, f, f]	[Ø, 2, Ø]	P1 entra a sección crítica
10	P1	[f, f, f]	[Ø, Ø, Ø]	P1 libera su ticket

Cola: Implementada como una lista enlazada. El productor inserta nodos al inicio de la lista (cabeza). El consumidor elimina el primer nodo disponible.

Pila de proceso: Se utiliza para almacenar variables locales y llamadas a funciones. Se libera memoria con cada eliminación de proceso.

PRODUCTOR CONSUMIDOR:

→Variables:

- turno[i]: Almacena los tickets de productor y consumidor.
- eligiendo[i]: Indica si el productor o consumidor están eligiendo su ticket.
- cabeza: Representa la cola de procesos.

→Inicialización:

- id_productor = 1
- id_consumidor = 2

Iteración	Acción	eligiendo	turno	Cola
1	Productor elige ticket	[1, Ø]	[1, Ø]	Vacía
2	Productor genera proceso 10	[Ø, Ø]	[Ø, Ø]	10(p=3)
3	Consumidor elige ticket	[Ø, 1]	[Ø, 1]	10(p=3)
4	Consumidor atiende proceso 10	[Ø, Ø]	[Ø, Ø]	Vacía
5	Productor elige ticket	[1, Ø]	[1, Ø]	Vacía
6	Productor genera proceso 11	[Ø, Ø]	[Ø, Ø]	11(p=2)
7	Consumidor elige ticket	[Ø, 1]	[Ø, 1]	11(p=2)
8	Consumidor atiende proceso 11	[Ø, Ø]	[Ø, Ø]	Vacía

ALGORITMO DE SEMÁFOROS Y MUTEXES

EJEMPLO DEL ALGORITMO (SECCIÓN CRÍTICA):

→Inicialización:

- semaforo = 1
- Los hilos t1 y t2 ejecutan la función proceso, que utiliza sem_wait para entrar a la sección crítica y sem_post para salir de ella.

Iteración	Acción	Semáforo	Salida
1	t1 y t2 se crean	1	
2	t1 ejecuta sem_wait()	ø	
3	t1 entra en la sección crítica	ø	"Proceso en sección crítica"
4	t1 ejecuta sem_post()	1	
5	t2 ejecuta sem_wait()	ø	
6	t2 entra en la sección crítica	ø	"Proceso en sección crítica"
7	t2 ejecuta sem_post()	1	

Cola: Es una lista enlazada donde los procesos se insertan al inicio. Los semáforos controlan la disponibilidad de espacios y elementos en la cola.

Pila de procesos: Contiene llamadas a funciones del hilo productor y consumidor. El mutex asegura que solo un hilo pueda modificar la lista a la vez, evitando condiciones de carrera.

PRODUCTOR CONSUMIDOR:

→Variables:

- espacios: Indica el número de espacios disponibles en la cola.
- elementos: Indica el número de procesos disponibles para ser consumidos.

} Semáforos

- Productores: Generan procesos y los insertan en la cola.
- Consumidores: Buscan el proceso con la prioridad más alta y lo eliminan.

} MUTEX: Protege el acceso a la lista enlazada de procesos.

→Inicialización:

- 3 productores (P1, P2, P3)
- 2 consumidores (C1, C2)

Iteración	Acción	espacios	elementos	Salida
1	P1 genera proceso 10 (Prioridad 2)	9	1	"Productor 1 generó proceso 10 con prioridad 2"
2	P2 genera proceso 20 (Prioridad 4)	8	2	"Productor 2 generó proceso 20 con prioridad 4"
3	C1 consume proceso 20	9	1	"Consumidor 1 atendiendo proceso 20 con prioridad 4"
4	P3 genera proceso 30 (Prioridad 3)	8	2	"Productor 3 generó proceso 30 con prioridad 3"
5	C2 consume proceso 30	9	1	"Consumidor 2 atendiendo proceso 30 con prioridad 3"
6	C1 consume proceso 10	10	ø	"Consumidor 1 atendiendo proceso 10 con prioridad 2"