Productor-Consumidor

Tacos de canasta

Descripción

Este problema es uno de los problemas clásicos de sincronización. El problema describe dos procesos, uno llamado productor y el otro consumidor, ambos comparten un recurso de tamaño finito.

La tarea del productor es generar un producto, almacenarlo y comenzar nuevamente; mientras que el consumidor toma (al mismo tiempo) productos uno a uno.

El problema ocurre cuando debido a que existe un recurso compartido y además este posee una cota superior en el número de elementos, y también puede tener ningún elemento. Por lo tanto, tanto el productor como el consumidor necesitan una manera de saber cuándo realizar su labor, y cuándo esperar.

Aplicación: tacos de canasta

- Hay un taquero y muchos clientes esperando a comer tacos.
- El taquero prepara tacos y los va poniendo en una canasta.
- Los clientes pasan a la barra y toman un taco cada uno.

Consideraciones

- Los tacos tienen el mismo tamaño
- La canasta tiene un número máximo de tacos.
- Para que no estorbe, un cliente solo puede pasar a la barra si hay al menos un taco en la canasta
- Para que los tacos no se enfríen, los clientes deben de tomarlos en el orden en que el taquero los fue metiendo.
- **Para que los clientes no se quemen, si el taquero está metiendo tacos a la canasta el cliente no puede intentar tomar uno, y viceversa.**
- El taquero es muy olvidadizo, así que no sabe cuál es el número máximo de tacos que caben en la canasta.

Solución

En dos displays controlados por botones llevamos dos cuentas:

- El número de tacos que hay en la canasta (A)
- Cuántos tacos más caben antes de que la canasta está llena (B)

Tanto el taquero como los clientes pueden controlarlos.

TACOS

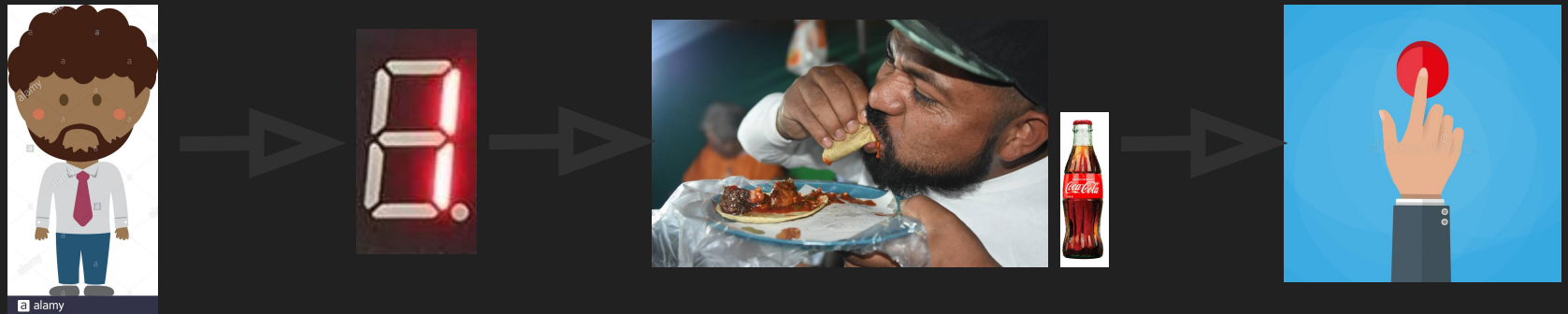
ESPACIO

Solución: cliente

Si el display A indica que hay al menos un taco, el cliente hace lo siguiente:

1. Pasa a la barra
2. **Toma un taco de la canasta (si el taquero no está metiendo tacos y otro cliente no está sacando)**
3. Presiona el botón que reduce el display A, para indicar que hay un taco menos en la canasta.
4. Presiona el botón que incrementa el display B, para indicar que hay un espacio más en la canasta.

Si el display indica que no hay tacos, el cliente espera hasta que haya uno.



Solución: taquero

Si el display B indica que hay al menos un espacio disponible en la canasta, el taquero hace lo siguiente:

1. **Pone un taco en la canasta (si el cliente no está tomando un taco)**
2. Presiona el botón que incrementa el display A, para indicar que hay un taco más en la canasta.
3. Presiona el botón que reduce el display B, para indicar que hay un espacio menos en la canasta.

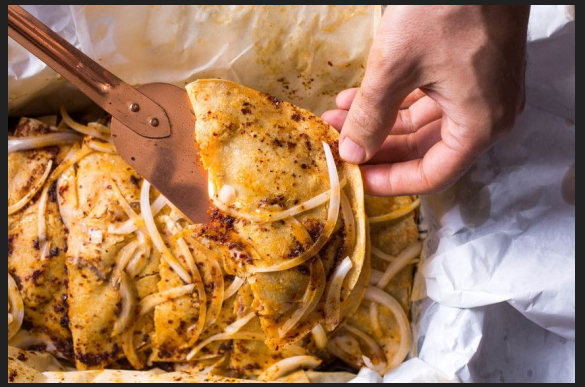
Si el display indica que no hay espacio para más tacos en la canasta, el taquero espera hasta que haya espacio.

Solución: análisis

Recurso compartido: canasta

Secciones críticas:

- Cuando el taquero mete un taco.
- Cuando el cliente toma un taco.



Método de sincronización: Semáforos

Variables enteras no negativas

Tipos:

- Binario: Toma valores en el conjunto {0,1}
- General: Toma cualquier valor entero no negativo.

Operaciones (atómicas):

- P: Intenta restar uno al semáforo, y espera hasta lograrlo
- V: Suma uno al semáforo
    - Para los semáforos binarios, V(1) = 1

Semáforo binario vs semáforo general

- El semáforo general no es esencial.
- Puede ser reemplazado por dos semáforos binarios y un contador

General	Binario

`Semáforo S, inicialmente K`	`val = K`
	`espera = Semaphore(min(1,val))`
	`mutex = Semaphore(1)`

- Todos los procesos esperan en el semáforo **`espera`**
- Si **`val<=0`** entonces **`espera=0`**. En otro caso **`espera=1`**.
- **`mutex`** garantiza que solo un proceso modifique **`val`**.

Semáforo binario vs semáforo general

General	Binario

`P(S)`	`P(S):`
	`  P(espera)`
	`  P(mutex)`
	`  val = val - 1`
	`  if val > 0:`
	`  V(espera)`
	`  V(mutex)`

`V(S)`	`V(S):`
	`  P(mutex)`
	`  val = val + 1`
	`  si val == 1:`
	`  V(espera)`
	`  V(mutex)`

Pseudocódigo

```
canasta = cola()
num_tacos = Semaforo(0)
espacio_libre = Semaforo(espacio)
tomando_canasta = Semaforo(1)

código taquero:
    repite infinitamente:
        1. taco = haz_taco()
        2. P(espacio_libre)
        3. P(tomando_canasta)
        4. pon_en_canasta(taco)
        5. V(tomando_canasta)
        6. V(num_tacos)
```

```
código cliente:
    repite infinitamente:
        1. P(num_tacos)
        2. P(tomando_canasta)
        3. taco = toma_de_canasta()
        4. V(tomando_canasta)
        5. V(espacio_libre)
```