Programación de Microprocesadores

Juan Salguero
24374

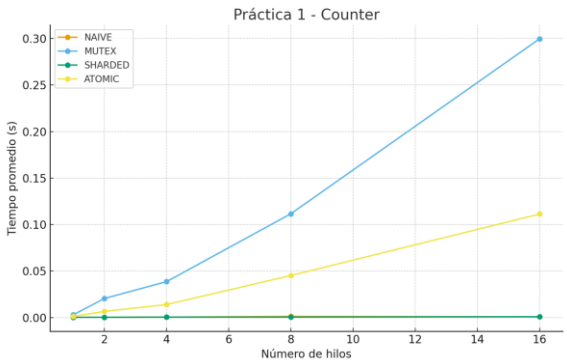
Laboratorio #6

Practica 1:

El programa implementa un contador global compartido entre varios hilos con cuatro variantes distintas.

```
pablownski@MSI:~/Lab6$ make p1
./bin/p1_counter 4 1000000
NAIVE total=1328320 (esperado=4000000), tiempo=0.001208 s
MUTEX total=4000000 (esperado=4000000), tiempo=0.282036 s
SHARDED total=4000000 (esperado=4000000), tiempo=0.000424 s
ATOMIC total=4000000 (esperado=4000000), tiempo=0.079734 s
```

Gráfico:



Análisis:

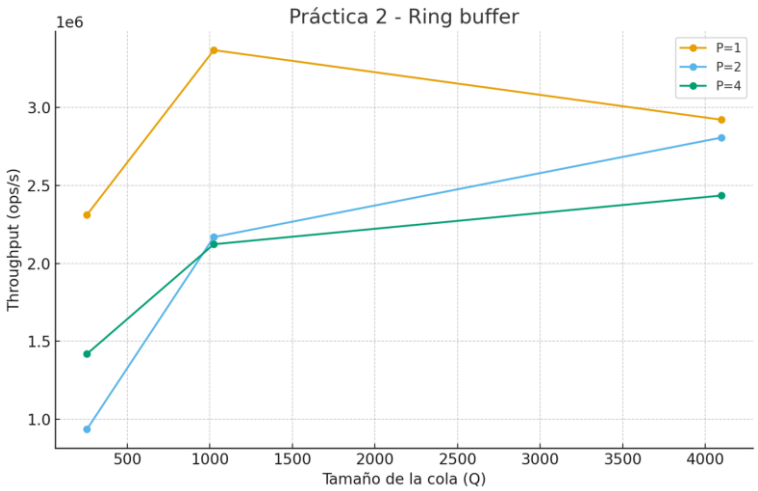
Se observa que la variante naive es la más rápida pero incorrecta por las condiciones de carrera, mientras que el uso de mutex asegura correctitud a costa de un tiempo que crece casi linealmente con los hilos. La estrategia sharded es la más eficiente, ya que reduce la

contención y mantiene tiempos estables, y la opción atomic ofrece un balance entre seguridad y rendimiento, aunque con un costo mayor que sharded.

Práctica 2:

```
pablownski@MSI:~/Lab6$ make p2
./bin/p2_ring 2 2 1024 100000
Consumidor terminó: 101088 items
Consumidor terminó: 98912 items
P=2, C=2, Q=1024, items=200000 -> tiempo=0.052619 s, throughput=3800878.22 ops/s
pablownski@MSI:~/Lab6$
```

Gráfico:



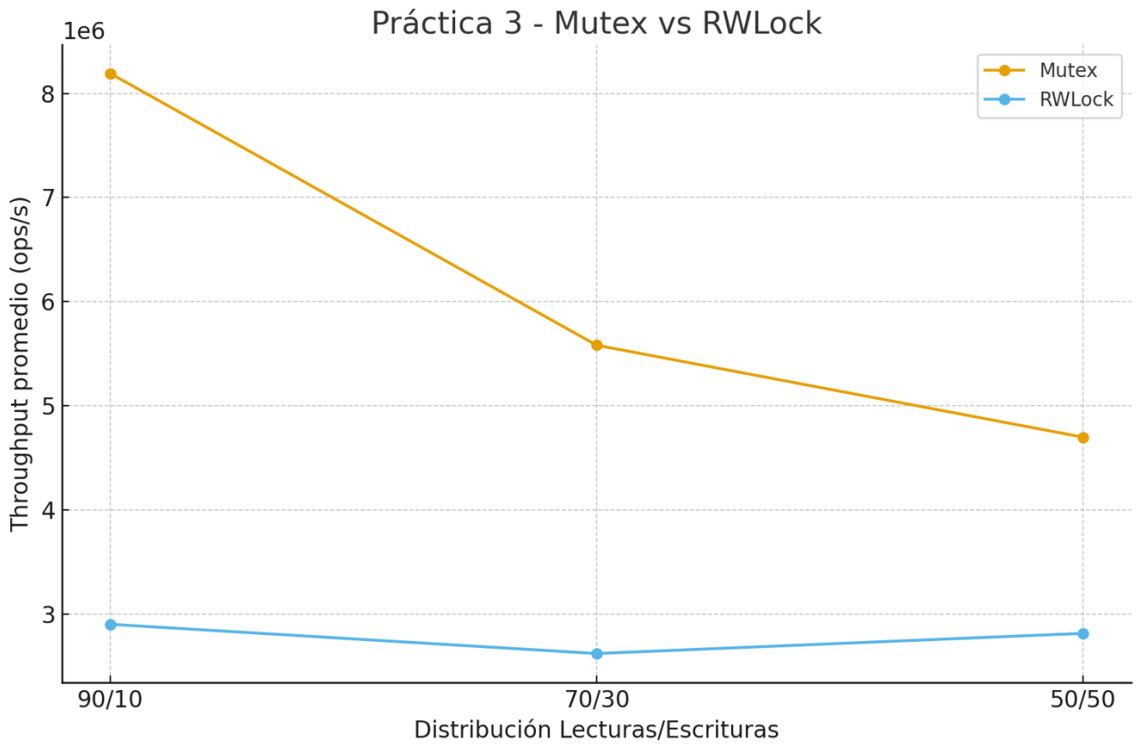
Análisis:

Se observa que el throughput aumenta al crecer el tamaño de la cola hasta un punto óptimo, ya que se reducen los bloqueos entre productores y consumidores. Con un productor se alcanza el mayor rendimiento, mientras que al incrementar el número de productores aparece mayor contención y el throughput se estabiliza en valores menores, mostrando que más productores no siempre implican mejor desempeño.

Practica 3:

```
pablownski@MSI:~/Lab6$ make p3
./bin/p3_rw 4 100000 0.1 0
T=4, ops=100000, writes=0.10, mode=mutex, tiempo=0.049412 s, throughput=8095186.28 ops/s
./bin/p3_rw 4 100000 0.1 1
T=4, ops=100000, writes=0.10, mode=rwlock, tiempo=0.234863 s, throughput=1703122.77 ops/s
pablownski@MSI:~/Lab6$
```

Gráfico:

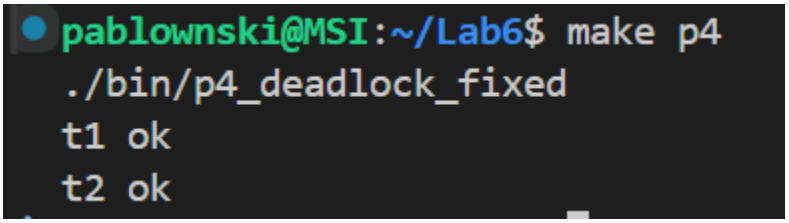


Análisis:

Los resultados muestran que el mutex mantiene un throughput mucho mayor en todas las distribuciones de lectura/escritura, disminuyendo gradualmente a medida que aumentan las escrituras. Por su parte, el rwlock se mantiene casi constante y con valores más bajos,

lo que indica que su mayor overhead no se ve compensado en estos escenarios, y que en cargas con pocas lecturas su ventaja teórica desaparece frente al mutex.

Practica 4:



Conclusión:

Se comprobó que el interbloqueo surge al adquirir recursos en distinto orden, y que imponer un orden global de locks elimina la espera circular, garantizando la correcta finalización de los hilos sin deadlocks.

Practica 5:

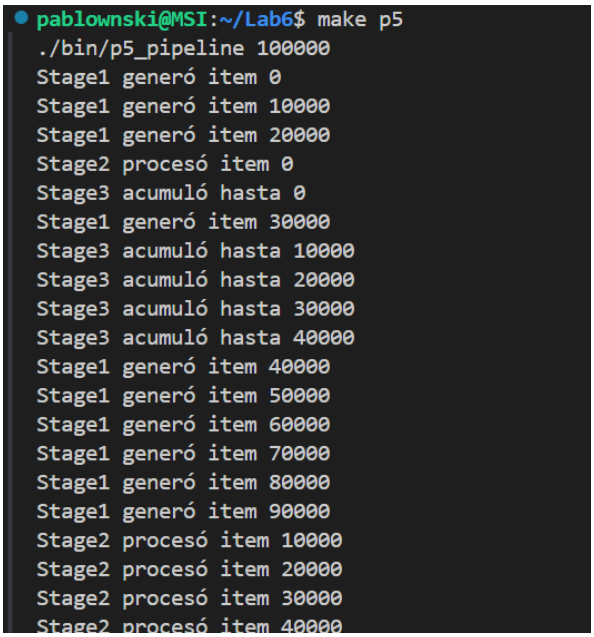
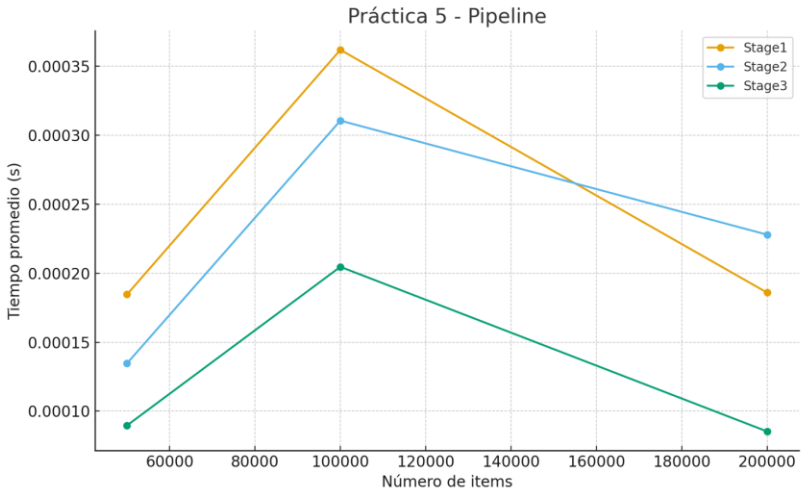


Gráfico:



Análisis:

La gráfica muestra que los tiempos promedio por ítem aumentan inicialmente con 100 000 elementos debido a la mayor carga de procesamiento, pero luego disminuyen al crecer hasta 200 000, lo que indica un mejor aprovechamiento del paralelismo del pipeline. Además, se observa que la Stage1 y Stage2 concentran la mayor parte del costo, mientras que la Stage3 es más ligera y no constituye un cuello de botella.

Link del video:

https://www.youtube.com/watch?v=C7SuiwXQGVg