

Ejercicio 1

5 trabajos (A-E) llegan al centro de cómputo casi al mismo tiempo. Los mismos tienen tiempos estimados de ejecución de 10, 6, 2, 4 y 8 minutos respectivamente. Sus prioridades (determinadas estáticamente) son 3, 5, 2, 1 y 4 respectivamente, siendo 5 la máxima prioridad. Para cada uno de los siguientes algoritmos de scheduling, determine el turnaround time medio.

1. Round Robin
2. Priority scheduling
3. First-come, First-served (considerar el orden A B C D E)
4. Shortest job first

Para 1. asuma que el sistema es multiprogramado y que cada trabajo obtiene la misma proporción de CPU. Para el resto asuma que solo 1 trabajo ejecuta hasta que termina sin interrupciones.

Todos los trabajos son CPU-bound
Ignore el tiempo necesario para el context switch

1. **Round robin** (cada uno la misma proporción de CPU) el orden es ABCDE con tiempos de ejecución A 10 B 6 C 2 D 4 E 8. Ignora prioridades (es el preemptive del FIFO, o sea orden de llegada pero los va interrumpiendo).

Como no está aclarado el tiempo de quantum lo tomo como el mínimo entre los tiempos que tardan los procesos listados, 2 mins en este caso.

A = 10 B = 6 C = 2 D = 4 E = 8 (tiempos restantes de ejecución)

A 2 B 2 C 2 | D 2 E 2 (tiempo corriendo)

A = 8 B = 4 C = 0 D = 2 E = 6

A 2 B 2 D 2 | E 2

A = 6 B = 2 C = 0 D = 0 E = 4

A 2 B 2 | E 2

A = 4 B = 0 C = 0 D = 0 E = 2

A 2 E 2 |

A = 2 B = 0 C = 0 D = 0 E = 0

A 2

El turnaround de A es la suma de todos los 2 $\rightarrow 2 \cdot 5 = 10$

El turnaround de B es la suma de todos los 2 hasta que B termina $\rightarrow 2 \cdot 3 = 6$

El turnaround de C es la suma de todos los 2 hasta que C termina $\rightarrow 2 \cdot 1 = 2$

El turnaround de D es la suma de todos los 2 hasta que D termina $\rightarrow 2 \cdot 2 = 4$

El turnaround de E es la suma de todos los 2 hasta que E termina $\rightarrow 2 \cdot 4 = 8$

El promedio es $(10 + 6 + 2 + 4 + 8)/5 = 20$ minutos

2. **Priority Scheduling**, en lugar de ver el orden de llegada respeto la prioridad asignada. El nuevo orden es: BEACD

Todos tienen prioridades diferentes \rightarrow Se ejecuta cada uno hasta que termine, porque se interrumpe pero se vuelve a elegir pues es el único con la prioridad más alta. Sigo considerando el quantum = 2 mins pues la consigna dice que uno solo corre hasta que termina sin interrupciones (con este criterio el quantum podría ser 3 también pero no 4).

(Si dos o más tuviesen la misma prioridad aplico round-robin entre ellos)

B 6 E 8 A 10 C 2 D 4

Turnaround de B es 6

Turnaround de E es $6 + 8 = 14$

Turnaround de A es $6 + 8 + 10 = 24$

Turnaround de C es $6 + 8 + 10 + 2 = 26$

Turnaround de D es $6 + 8 + 10 + 2 + 4 = 30$

El promedio es $(6+14+24+26+30)/5 = 20$

3. First Come First Served ABCDE

Ejecuto en orden de llegada cada uno hasta que termine (no lo interrumpo)

A 10 B 6 C 2 D 4 E 8

Turnaround time de A es 10

Turnaround time de B es $10 + 6 = 16$

Turnaround time de C es $10 + 6 + 2 = 18$

Turnaround time de D es $10 + 6 + 2 + 4 = 22$

Turnaround time de E es $10 + 6 + 2 + 4 + 8 = 30$

El promedio es $(10+16+18+22+30)/5 = 19.2$

4. Shortest job first → el orden es C 2 D 4 B 6 E 8 A 10

Ejecuto en ese orden a cada uno hasta que termine (no lo interrumpo)

El turnaround time de C es 2

El turnaround time de D es $2 + 4 = 6$

El turnaround time de B es $2 + 4 + 6 = 12$

El turnaround time de E es $2 + 4 + 6 + 8 = 20$

El turnaround time de A es $2 + 4 + 6 + 8 + 10 = 30$

El promedio es $(2+6+12+20+30)/5 = 14$

Ejercicio 2

Los procesos de un sistema particular tienen un tiempo de ejecución promedio T antes de bloquearse por I/O. El tiempo necesario para un context switch es S , el cual se considera desperdiciado. Para el algoritmo de scheduling Round-Robin con un quantum Q , provea una fórmula de la eficiencia del CPU para cada una de las siguientes condiciones.

1. $Q = \infty$
2. $Q > T$
3. $S < Q < T$
4. $Q = S$
5. Q cercano a 0

Tiempo de ejecución (uso de CPU) T o Q

Tiempo de context switch (usa el CPU) S

1. $Q = \infty$

Con un quantum ∞ no se interrumpe a los procesos, el tiempo de CPU que utiliza un proceso es T , y tiempo total de uso de CPU es $T+S \rightarrow$ la eficiencia (%) es

$$\frac{T}{T+S} * 100$$

2. $Q > T$

Con un quantum mayor al tiempo de ejecución T vale la fórmula anterior pues tampoco se interrumpen los procesos.

3. $S < Q < T$

Ahora el Quantum es menor a T y los procesos son interrumpidos, por lo tanto los procesos corren un tiempo Q . El tiempo de CPU que utiliza un proceso es Q y el tiempo total de uso de CPU es $Q+S \rightarrow$ la eficiencia (%) es

$$\frac{Q}{Q+S} * 100$$

4. $Q = S$ (asumo $S < T$)

Como el quantum es igual al tiempo del context switch, usando la fórmula del punto anterior, reemplazando y simplificando \rightarrow la eficiencia (%) es

$$\frac{S}{S+S} * 100 = \frac{1}{2} * 100 = 50$$

5. Q cercano a cero

Si el quantum tiende a cero, utilizando la fórmula del punto 3, el numerador tiende a cero y el denominador a S, entonces la eficiencia tiende a cero.

Ejercicio 3

5 trabajos están esperando para ser ejecutados. Sus tiempos de ejecución esperados son 9, 6, 3, 5 y X. ¿En qué orden deben ejecutarse para minimizar el turnaround time?

De menor a mayor (así figura en el cálculo del promedio el tiempo más corto)

En el orden 3 5 6 9 X (Como X no lo conozco por las dudas lo pongo al final)

El turnaround time promedio es: $3 + (3+5) + (3+5+6) + (3+5+6+9) + (3+5+6+9+X) / 5$

Ejercicio 3

La técnica de envejecimiento (aging) con $a = 1/2$ es usada para estimar los tiempos de ejecución de un proceso. Las últimas 4 ejecuciones registraron los siguientes tiempos (de más antiguo a más reciente): 40, 20, 40 y 15 ms. ¿Cuál es la estimación para la siguiente ejecución?

$$a = 1/2$$

$$T_0 = 40$$

$$T_1 = 20$$

$$T_2 = 40$$

$$T_3 = 15$$

S_i estima la ejecución (i+1)-ésima (la 4-ésima es la 5ta porque cuento desde el cero)

$$S_0 = T_0$$

$$S_1 = aT_0 + (1-a)T_1 = 1/2 * 40 + 1/2 * 20 = 30$$

$$S_2 = aS_1 + (1-a)T_2 = 1/2 * 30 + 1/2 * 40 = 35$$

$$S_3 = aS_2 + (1-a)T_3 = 1/2 * 35 + 1/2 * 15 = 25$$

La estimación para la próxima ejecución es de 25 ms