

1. Un programa tarda 40 segundos en ejecutarse en un SMP (multiprocesador), durante el 20% del tiempo se ha ejecutado en 4 procesadores, el 60% del tiempo en 3 y el 20% restante en 1. Consideramos que la carga se ha distribuido por igual. ¿Cuándo tardaría el programa en ejecutarse en un solo procesador? Hallar la ganancia y la eficiencia.

20% 4 procs.                       $40 = 40*0.2 + 40*0.6 + 40*0.2$   
 60% 3 procs.                       $T_{\text{ptotal}} = T_{4\text{procs}} + T_{3\text{procs}} + T_{1\text{procs}}$   
 20% 1 proc.  
 Tsec?  
 Eficiencia?  
 SpeedUp?

#### TIEMPO PARALELIZADO

$$T_{4\text{procs}} = 40*0.2 = 8 \text{ s}$$

$$T_{3\text{procs}} = 40*0.6 = 24 \text{ s}$$

$$T_{1\text{procs}} = 40*0.2 = 8 \text{ s}$$

#### TIEMPO SIN PARALELIZAR

$$T_{\text{sec}(4 \text{ procs})} = 8*4 = 32 \text{ s}$$

$$T_{\text{sec}(3 \text{ procs})} = 24*3 = 72 \text{ s}$$

$$T_{\text{sec}(1 \text{ proc})} = 8*1 = 8 \text{ s}$$

$$T_{\text{sec}} = 32+72+8 = 112 \text{ s}$$

#### SPEED UP (GANACIA)

$$SP = \frac{T_s}{T_p}$$

$$SP = \frac{112}{40} = 2.8 \leq 4 \text{ :)}$$

#### EFICIENCIA

$$E = \frac{SP}{N}$$

$$E = \frac{2.8}{4} = 0.7 \leq 1 \text{ :)}$$

2. Un programa tarda 20 segundos en ejecutarse en un procesador p1 y 30 en otro p2. Quiero utilizar P1 y P2 para ejecutar el mismo programa en paralelo. ¿Qué tiempo tarda en ejecutarse el programa si la carga de trabajo se distribuye por igual entre los dos procesadores? [No se tiene en cuenta la sobrecarga]. Calcular el Speed-Up y la eficiencia.

Si repartimos de forma equivalente el trabajo cada sección paralela tardara:

- $T_{pp1} = 1/2 * 20 = 10 \text{ s}$
- $T_{pp2} = 1/2 * 30 = 15 \text{ s}$

Para calcular cuánto tarda el programa en ejecutarse en total tendremos que hacer lo siguiente:

$$T_{N=2} = \text{MAX}(T_{pp1}, T_{pp2}) = 15 \text{ s}$$

Lo cual se puede explicar diciendo que si repartimos el trabajo de forma equivalente sin tener en cuenta las especificaciones de cada procesador, el más lento será nuestro factor condicionante.

A continuación calculamos el Speed-Up y la eficiencia, en este caso para el tiempo secuencial cogeremos el más rápido de los dos.

#### GANANCIA

$$SP = \frac{T_s}{T_p}$$

$$SP = \frac{20}{15} = 1.33 \leq 2 \text{ :)}$$

#### EFICIENCIA

$$E = \frac{SP}{N}$$

$$E = \frac{1.33}{2} = 0.66 \leq 1 \text{ :)}$$

*Nota: Ganancia máxima teórica == numero de nodos*

- b) Que distribución de carga entre los dos procesadores P1 y P2 permite el menor tiempo de ejecución utilizando los dos procesadores en paralelo. Calcula el tiempo.

$$x_1 = x \quad t_{pp1}(x) = t_{pp2}(1-x)$$

$$x_2 = 1 - x \quad 20(x) = 30(1-x)$$

$$5x = 3 \rightarrow x = 3/5$$

A  $t_{pp1}$  se le dan 3/5 del problema y a  $t_{pp2}$  se le dan 2/5 del problema.  $t_{pp1}$  procesa mas parte del problema ya que es más rápido. Al haber distribuido la carga, ambos procesadores tardarán lo mismo a la hora de resolver el problema:

$$3/5 * 20 = 12 \text{ s}$$

$$2/5 * 30 = 12 \text{ s}$$

3. Cuál es la fracción de código paralelo de un programa secuencial que, ejecutado en paralelo en 8 procesadores tarda un tiempo de 100 ns, asumiendo que durante 50 ns utiliza un único procesador y durante los otros 50 los 8 procesadores. Todos los procesadores son iguales. Hallar el tiempo secuencial.

Lo primero que haremos será hallar el tiempo secuencial.

$$t_{\text{sec}} = 8 * 50 + 50 = 450 \text{ ns}$$

Una vez tenemos el tiempo secuencial total podemos hacer lo mismo que hicimos en el ejercicio anterior:

$$T_p = 100 \text{ ns} = x * 450 + ((1-x) * 450) / 8$$

La x es el valor que queremos hallar por lo tanto despejamos:

$$x = 1/9$$

Ahora bien, ese 1/9 sería la fracción de código sin paralelizar, pues para hallar la fracción de código paralelizado haremos:

$$1 - 1/9 = 8/9$$

b) Calcula la ganancia y la eficiencia en paralelo.

$$SP = T_{\text{sec}} / T_p$$

$$SP = 450 / 100 = 4.5 \leq 8 \text{ :)}$$

$$E = SP / N$$

$$E = 4.5 / 8 = 0.5625 \leq 1 \text{ :)}$$

4. El 25% de un programa no se puede paralelizar y el resto se puede distribuir por igual entre cualquier número de procesadores.

a) ¿Cuál es el máximo valor de ganancia en velocidad que se podría conseguir al paralelizarlo en  $p$  procesadores?

b) ¿Y con infinitos?

c) ¿A partir de que número de procesadores podríamos conseguir ganancias superiores o iguales a 2?

d) Calcula la eficiencia

e) Máxima eficiencia que puedo obtener cuando  $p$  tiende a  $\infty$

$$\begin{aligned} \text{a) } SP &= t_{\text{sec}}/t_{\text{pp}} = t_{\text{sec}} / ((0.25)*t_{\text{sec}} + (0.75*(1/p))*t_{\text{sec}}) = \\ &= 4p/(3+p) \end{aligned}$$

Para 4 nodos por ejemplo:  $4*4/(3+4) = 16/7 = 2.28 \geq 2$  :)

$$\text{b) } \lim_{p \rightarrow \infty} SP = 4p/(3+p) = 4\infty/\infty = 4$$

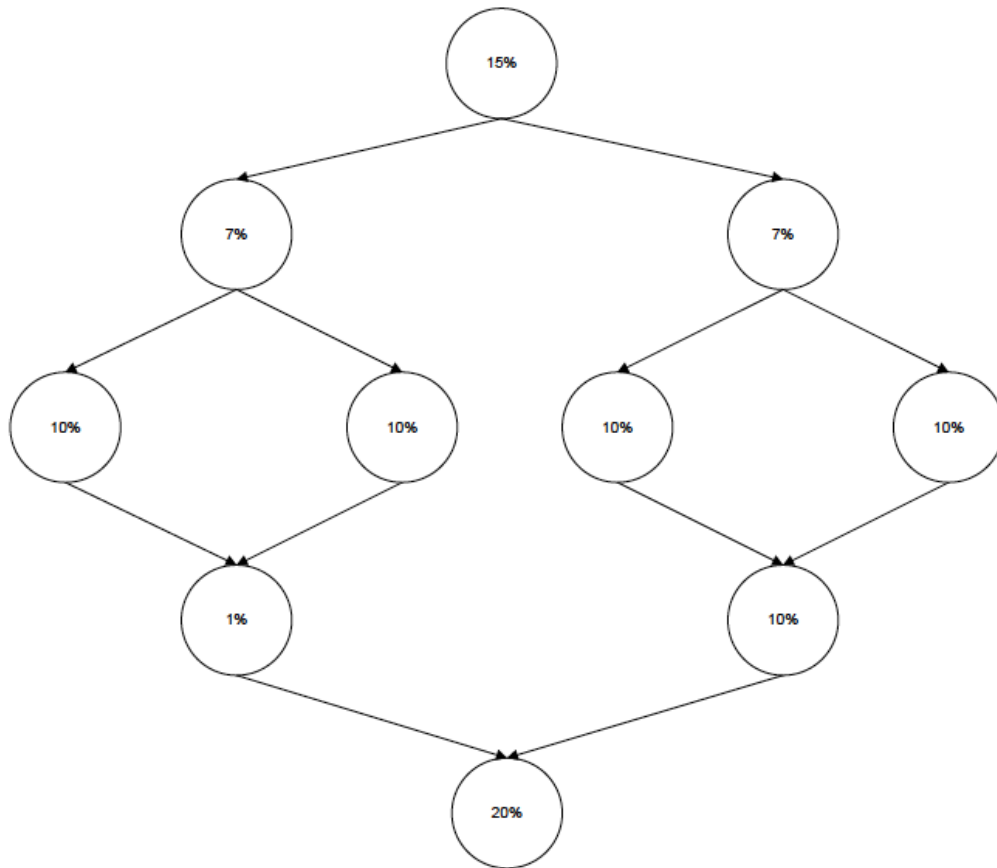
$$\text{c) } 4p/(3+p) = 2 \rightarrow \text{Despejamos} \rightarrow p = 3$$

$$\text{d) } E = SP/N = (4p/(3+p))/p$$

$$\text{e) } \lim_{p \rightarrow \infty} E = 0$$

Si metemos nodos infinitos la eficiencia acabará siendo 0.

5. Tenemos un programa que hemos dividido en 10 tareas, las cuales tardan en ejecutarse 5 segundos. En la siguiente figura podemos ver como se divide el cómputo entre las distintas tareas y el orden de precedencia de las mismas. Si disponemos de 4 procesadores, se pide calcular el tiempo de ejecución de la versión paralela del programa, así como la ganancia en velocidad obtenida al realizar la paralelización.



Tiempo de ejecución en paralelo:

$$T_p(n) = T_s \times (0.15 + \max(0.07, 0.07) + \max(0.1, 0.1, 0.1, 0.1) + \max(0.01, 0.1))$$

$$T_p(n) = T_s \times (0.15 + 0.07 + 0.1 + 0.1) = 5s \times 0.42 = 2.1s$$

Ganancia:

$$S_{(p,n)} = \frac{T_s}{T_p(n)} = \frac{5s}{2.1s} = 2.38$$

6. Se quiere paralelizar el siguiente trozo de código:

```
// {Cálculos antes del bucle} | t1
for(int i = 0; i < w; i++) {
    // iteración i | ti
}
// {Cálculos después del bucle} | t2
```

Los cálculos después del bucle supone  $t_2$ , los de antes del bucle  $t_1$ . Cada iteración del bucle supone un tiempo  $t_i$ . En la ejecución paralela, la inicialización de  $p$  procesos supone un tiempo con  $k_1$  constante ( $k_1 \cdot p$ ). La comunicación y sincronización supone un tiempo  $k_2$  ( $k_2 \cdot p$ ).

- a) Hallar  $tp(p)$
- b) Hallar el Speed Up de  $p$
- c) ¿Existe un mínimo de  $p$  para la ejecución en paralelo?

a) Llamamos  $t'$  al tiempo que no se puede paralelizar

$$t' = t_1 + t_2$$

Al tener overhead lo dejamos listo así.

$$k_1 p \rightarrow to(p) = (k_1 + k_2)p = k'p; \text{ siendo } k' = k_1 + k_2$$

Tenemos  $w$  iteraciones paralelizables por  $p$  procesadores, por lo tanto:

$$tp(p) = t' + k'p + \lceil w/p \rceil \cdot t_i$$

$tp(p) = t$ . no paralelizable + overhead + tiempo paralelizable.  
 $\lceil w/p \rceil$  implica coger el siguiente entero ya que no podemos tener números no enteros (ver OMP práctico).

$$b) SP(p) = tsec/tp = (t' + t_i \cdot W) / tp(p)$$

- c) Para simplificar  $w*ti = k''$   
 Si  $tp(p) = t' + k'p + k''/p$

Derivamos

$$tp'(p) = 0 + k' - k''/p^2$$

Igualamos la derivada a 0 para sacar el punto de inflexión.

$$k' = k''/p^2;$$

$$p = +\sqrt{(k''/k')} \rightarrow p = +\sqrt{(w*ti)/k'};$$

(Cogemos el resultado positivo  $+\sqrt{\phantom{x}}$  ya que no puede haber un número de procesadores negativo)

Hayamos la segunda derivada y comprobamos que efectivamente sí que existe un mínimo

$$tp''(p) = + k''/p^3 \rightarrow tp''(p) = + (w*ti)/p^3$$

Sustituimos el punto crítico dado en la 1ª derivada en la 2ª

$$tp''(p) = (w*ti)/(+\sqrt{(w*ti)/k'})^3$$

Como podemos observar no cabe posibilidad alguna que el resultado de la segunda derivada sea menor que 0. Por lo tanto obtendríamos un mínimo.

$X > 0 \rightarrow$  Minimo

$X < 0 \rightarrow$  Maximo

$X = 0 \rightarrow$  Punto de inflexión