# Rendimiento de sistemas paralelos

Arquitectura de Computadoras II
Fac. Cs. Exactas
UNCPBA
Prof. Marcelo Tosini
2015

# Rendimiento de un sistema paralelo

Rendimiento en un sistema con un procesador:

$$T_{cpu} = RI \times CPI \times t_{ciclo}$$

Con RI : recuento de instrucciones
CPI: ciclos promedio por instrucción
$t_{ciclo}$: duración del ciclo de reloj

Rendimiento en un sistema con P procesadores:

$$T_{CPU} = \frac{RI}{P} * \frac{1}{IPC} * t_{CICLO}$$

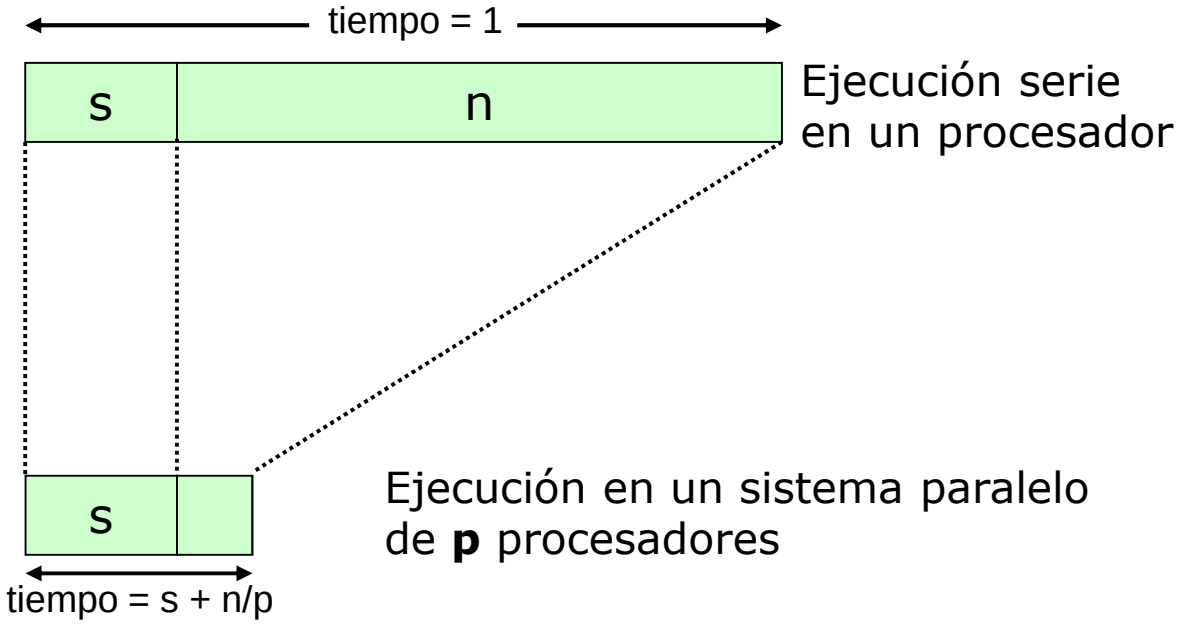
Con P : número de procesadores
IPC: número de instrucciones por ciclo de reloj
$t_{ciclo}$: duración del ciclo de reloj

# Rendimiento de un sistema paralelo

**Según Amdahl:**

"Cuando la fracción de un trabajo serie de un problema es pequeña, y la denominamos s, la máxima aceleración o Speedup alcanzable (incluso para un número infinito de procesadores) es sólo 1/s"

$$A_g = \frac{s+n}{s+\frac{n}{p}} = \frac{1}{s+\frac{n}{p}}$$



# Límites del paralelismo

**Según la ley de Amdahl:**

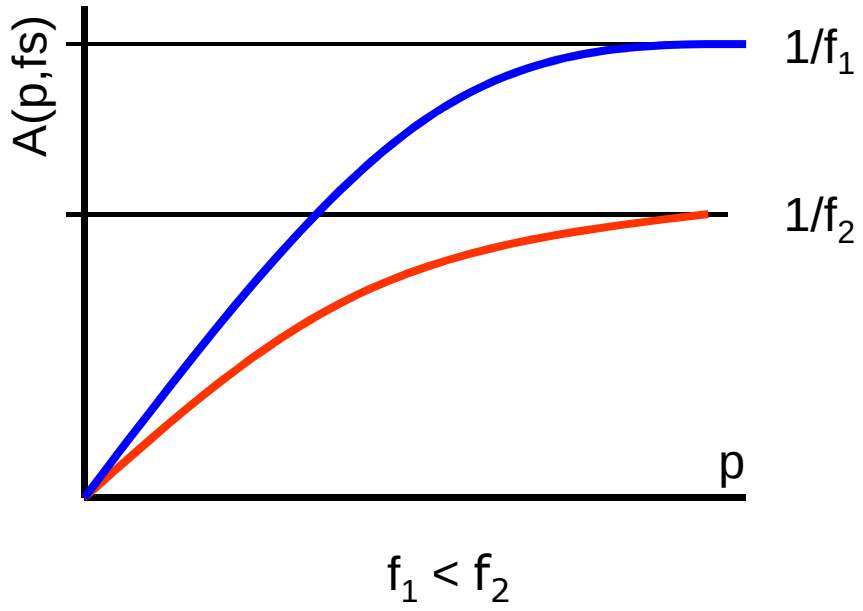
En todo problema hay una parte secuencial, fs, en la que no es posible utilizar la potencia de los p procesadores de un sistema. De este modo, la ganancia de velocidad se ve limitada como se muestra a continuación

$$A = \frac{1}{(1 - f_m) + \frac{f_m}{a_m}}$$

Con : factor de mejora am = p (procesadores) y
(1-fm) = fs (parte no paralelizable)

$$A(p, fs) = \frac{p}{p(fs + \frac{1 - fs}{p})} = \frac{p}{p * fs + 1 - fs}$$

$$A(p, fs) = \frac{p}{1 + fs * (p - 1)}$$



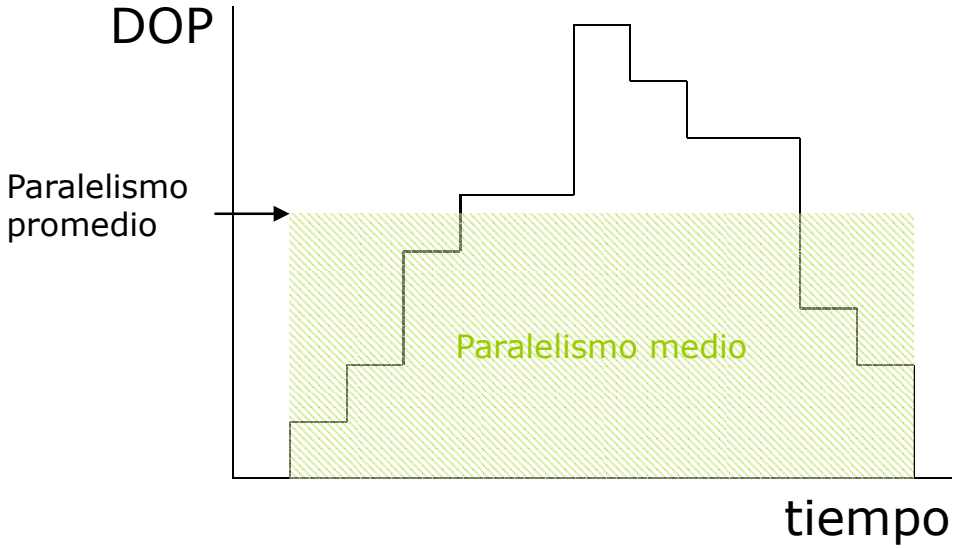
# Grado de paralelismo (DOP)

**Representa el grado de mapeo entre el paralelismo del software y el grado de procesamiento paralelo del hardware.**

Se mide en intervalos de tiempo cuantos procesadores se están utilizando para un algoritmo dado

Si, P = número de procesadores del sistema; m = máximo paralelismo posible de un algoritmo dado y DOP = número de procesos paralelos en los que se puede dividir un programa, entonces

**DOP <= m < P**



# Grado de paralelismo (DOP) (cont.)

El trabajo total es proporcional a la curva bajo de DOP

$$W = \Delta \sum_{i=t1}^{t2} i * t_i = \Delta \sum_{i=t1}^{t2} W_i$$

con t1: intervalo inferior de tiempo
t2: intervalo superior de tiempo
i: número de procesos paralelos en el intervalo ti
ti: sumatoria de todos los intervalos en que se usan i procesadores

Paralelismo promedio:

$$A = \frac{\Delta \sum_{i=t1}^{t2} i * t_i}{\Delta \sum_{i=t1}^{t2} t_i}$$

con **Δ** la *capacidad de procesamiento* de un procesador, expresada en MIPS o Mflops, sin considerar las penalizaciones debidas al acceso a memoria, latencia de las comunicaciones, o sobrecarga del sistema.

# Grado de paralelismo (ejemplo)

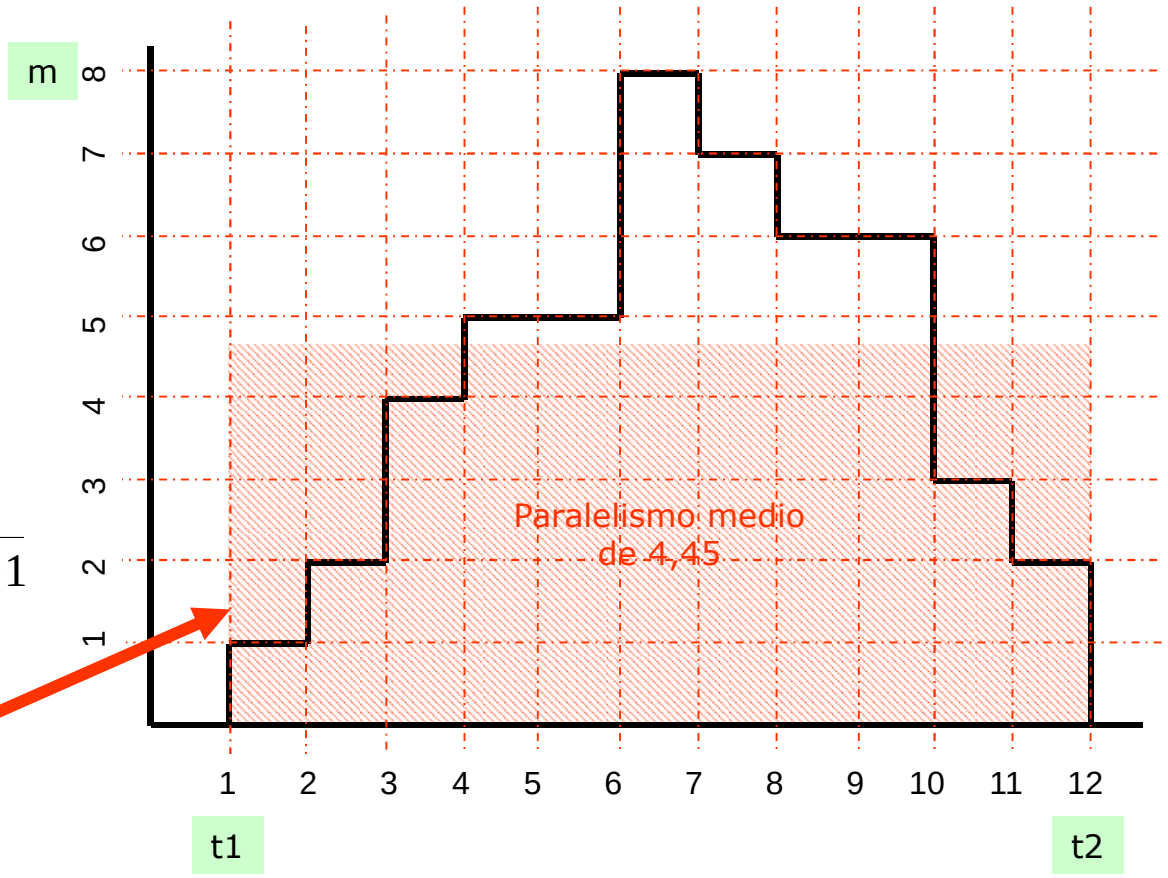
$$W = \sum_{i=t1}^{t2} i * t_i$$

*W = 1.1+2.2+3.1+4.1+5.2+6.2+7.1+8.1*

*W = **49***

$$A = \frac{W}{\sum_{i=t1}^{t2} t_i} = \frac{49}{1+2+1+1+2+2+1+1}$$

*A = 49/11 = **4,45***



# Métricas de rendimiento

**Speedup asintótico**

Sea $W_i = i\Delta t_i$ el trabajo realizado cuando DOP = *i, entonces* $W = \sum_{i=1}^{m} W_i$

El tiempo de ejecución de $W_i$ para un solo procesador es $t_i(1) = W_i/\Delta$.

El tiempo de ejecución de $W_i$ para *k* procesadores es $t_i(k) = W_i/k\Delta$.

El tiempo de ejecución de $W_i$ para ∞ procesadores es $t_i(\infty) = W_i/i\Delta$ *con* $1 \leq i \leq m$

# Speedup asintótico

Los tiempos de respuesta para 1 procesador y para ∞ procesadores son:

$$T(1) = \sum_{i=1}^{m} t_i(1) = \sum_{i=1}^{M} \frac{W_i}{\Delta}$$

$$T(\infty) = \sum_{i=1}^{m} t_i(\infty) = \sum_{i=1}^{M} \frac{W_i}{i\Delta}$$

Entonces, el speedup asintótico será el cociente de T(1) y T(∞), o sea, un parámetro que mide la aceleración del tiempo de cálculo por el hecho de poder paralelizar al máximo la aplicación:

$$S_\infty = \frac{T(1)}{T(\infty)} = \frac{\sum_{i=1}^{m} W_i}{\sum_{i=1}^{m} \frac{W_i}{i}}$$

Con lo que $S_\infty$ = *A en el caso ideal y*

$S_\infty \leq$ *A en el caso real debido a las latencias de comunicaciones y sobrecarga del sistema*

# Medidas de rendimiento

Si se tiene p procesadores ejecutando N programas con diferentes grados de paralelismo (i)

Entonces el rendimiento medio se define como:

- **Rendimiento basado en media aritmética (smith,88)**
  - **Simple**
  - **Ponderada**
- **Rendimiento basado en media geométrica (smith,88)**
  - **Normalizada**
- **Rendimiento basado en media armónica**

# Rendimiento basado en media aritmética (Simple)

**Rendimiento inversamente proporcional a la suma de los tiempos de ejecución**

$$\frac{1}{R_a} = \frac{\sum_{i=1}^{n} tiempo_i}{n}$$

Con $tiempo_i$, tiempo de ejecución del programa i-esimo de un total de $n$ de la carga de trabajo

# Ejemplo

La tabla muestra los tiempos de ejecución de 2 programas en tres máquinas distintas

| Segundos | Máquina A | Máquina B | Máquina C |
|---|---|---|---|
| Programa 1 | 1 | 10 | 20 |
| Programa 2 | 1000 | 100 | 20 |
| total | 1001 | 110 | 40 |

Se ve que…

- A es 1900% más rápido que C para el programa 1
- B es 100% más rápido que C para el programa 1
- B es 900% más rápido que A para el programa 2
- ……

- B es 810% más rápido que A para los programas 1 y 2
- C es 2400% más rápido que A para los programas 1 y 2
- B es 175% más rápido que B para los programas 1 y 2

# Ejemplo…

La **media aritmética** para las 3 máquinas A, B y C será:

$$T_A = \frac{1+1000}{2} = 500{,}5seg.$$

$\rightarrow R_A = 1/T_A =$ **0,002**

$$T_B = \frac{10+100}{2} = 55seg.$$

$\rightarrow R_B = 1/T_B =$ **0,018**

$$T_C = \frac{20+20}{2} = 20seg.$$

$\rightarrow R_C = 1/T_C =$ **0,05**

$\rightarrow R_C > R_B > R_A$

# Rendimiento basado en media aritmética (Ponderado)

**Rendimiento inversamente proporcional a la suma de los tiempos de ejecución ponderados por la frecuencia del programa en la carga de trabajo**

$$\frac{1}{R_a} = \sum_{i=1}^{n} f_i * tiempo_i$$

Con $tiempo_i$, tiempo de ejecución del programa i-esimo de un total de $n$ de la carga de trabajo y $f_i$, la frecuencia relativa del i-esimo programa en la carga de trabajo

# Ejemplo…

**Media aritmética ponderada** para las 3 máquinas A, B y C según 2 cargas distintas

| f1 = 50%<br>f2 = 50% | A | B | C |
|---|---|---|---|
| Prog. 1 | 1 | 10 | 20 |
| Prog. 2 | 1000 | 100 | 20 |
| total | 1001 | 110 | 40 |
| media aritmética | 500,5 | 55 | 20 |

ponderación equivalente (media aritmética simple)

$$f_i = \frac{1}{T_i * \sum_{j=1}^{n} 1/T_j}$$

ponderación inversamente proporcional a los tiempos de ejecución en la máquina A

f1 = 1000/1001 = 0,999
f2 = 1/1001 = 0,001

| f1 = 99,9%<br>f2 = 0,1% | A | B | C |
|---|---|---|---|
| Prog. 1 | 1 | 10 | 20 |
| Prog. 2 | 1000 | 100 | 20 |
| total | 1001 | 110 | 40 |
| media aritmética | 2,0 | 10,09 | 20,0 |

# Rendimiento basado en media geométrica (Normalizado)

**Rendimiento inversamente proporcional al producto de los tiempos de ejecución**

$$\frac{1}{R_g} = \sqrt[n]{\prod_{i=1}^{n} f_i * tiempo_i}$$

Con $tiempo_i$, tiempo de ejecución del programa i-esimo de un total de $n$ de la carga de trabajo y $f_i$, la frecuencia relativa del i-esimo programa en la carga de trabajo

# Ejemplo…

**Media geométrica normalizada** para las 3 máquinas A, B y C.

| | Normalizado para A (%) | | | Normalizado para B (%) | | | Normalizado para C (%) | | |
|---|---|---|---|---|---|---|---|---|---|
| | **A** | **B** | **C** | **A** | **B** | **C** | **A** | **B** | **C** |
| Prog. 1 | 100 | 1000 | 2000 | 10 | 100 | 200 | 5 | 50 | 100 |
| Prog. 2 | 100 | 10 | 2 | 1000 | 100 | 20 | 5000 | 500 | 100 |
| M. arit. | 100 | 505 | 1001 | 505 | 100 | 110 | 2503 | 275 | 100 |
| M. Geom. | 100 | 100 | 63 | 100 | 100 | 63 | 158 | 158 | 100 |
| Tiempo total | 100 | 11 | 4 | 910 | 100 | 36 | 2503 | 275 | 100 |

- El rendimiento de la media aritmética varía dependiendo de cual sea la máquina de referencia

$R_A > R_B$ $R_A < R_B$

- El rendimiento de la media geométrica es más consistente independientemente de la normalización

$R_C$= 63% de $R_A$ y $R_C$= 63% de $R_B$ y $R_A = R_B$

# Rendimiento basado en media armónica (ponderado)

$$R_i = \frac{1}{T_i}$$

Rendimiento o velocidad de ejecución para el programa i-esimo de un total de n de la carga de trabajo

$$R_h = \frac{1}{\sum_{i=1}^{n}(f_i / R_i)}$$

Con $f_i$, la frecuencia relativa del i-esimo programa en la carga de trabajo

# Modelos de computadores paralelos

**3 modelos de medición del speed-up:**

- Limitación por carga de trabajo fija:
  - Modelo sugerido por Amdahl (1967)
  - Basado en una carga de trabajo fija o en un problema de tamaño fijo

- Limitación por tiempo fijo:
  - Modelo sugerido por Gustafson (1987)
  - Problemas escalables
  - El tamaño del problema se incrementa al aumentar el tamaño de la máquina
  - Se dispone de un tiempo fijo para realizar una determinada tarea

- Limitación por memoria fija:
  - Modelo sugerido por Sun & Ni (1993)
  - Se aplica a problemas escalables limitados por la capacidad de memoria

# Limitación por carga de trabajo fija (Fixed Workload)

Usada en aplicaciones donde es importante la respuesta más rápida posible.

La carga de trabajo se mantiene fija y es el tiempo de ejecución lo que se debe intentar reducir

El factor de speed-up está acotado superiormente por el cuello de botella secuencial

**2 casos posibles:**

**DOP = *i* ≥ P**

Los P procesadores se usan para ejecutar Wi

$$t_i(P) = \frac{W_i}{i\Delta}\left\lceil \frac{i}{P} \right\rceil$$

$$T(P) = \sum_{i=1}^{m} \frac{W_i}{i\Delta}\left\lceil \frac{i}{P} \right\rceil$$

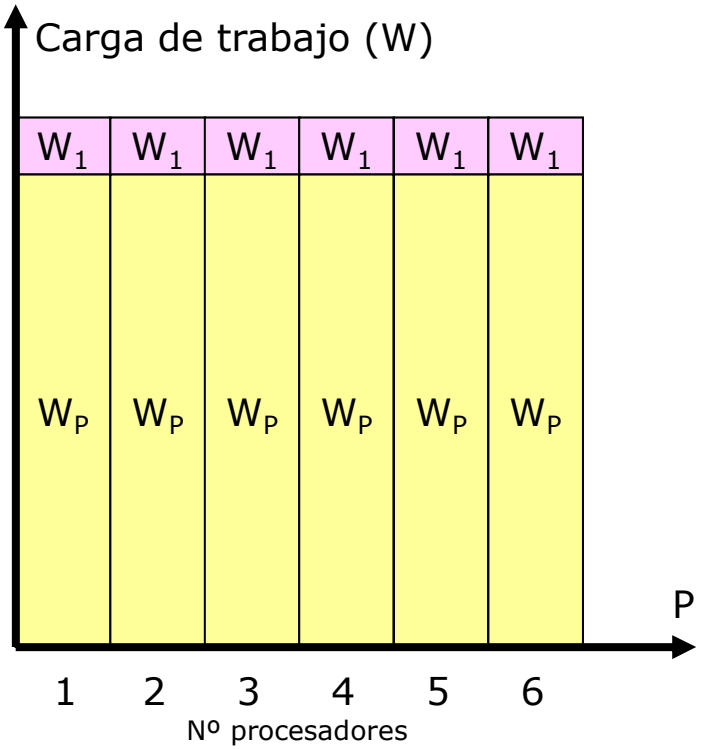
**DOP = *i* < P**

i procesadores se usan para ejecutar Wi

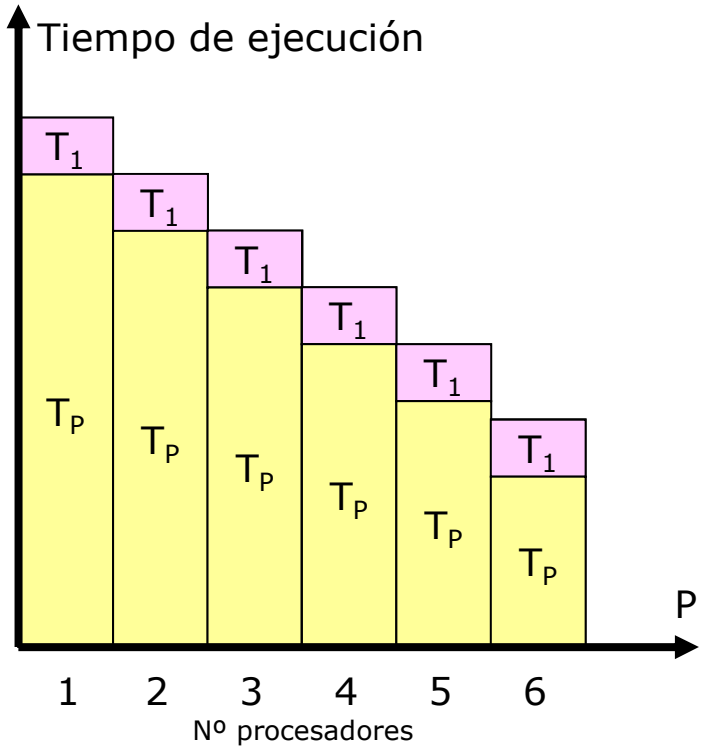
$$t_i(P) = \frac{W_i}{i\Delta}$$

$$T(P) = \sum_{i=1}^{m} \frac{W_i}{i\Delta}$$

# Limitación por carga de trabajo fija (Fixed Workload)




Carga fija




Tiempo de ejecución decreciente

# Limitación por carga de trabajo fija (Fixed Workload)

De T(1) y T(P) se puede derivar el speed-up para P procesadores.

$$S_P = \frac{T(1)}{T(P)} = \frac{\sum_{i=1}^{m} W_i}{\sum_{I=1}^{m} \frac{W_i}{i} \left\lceil \frac{i}{P} \right\rceil} \approx P$$

En el caso ideal

En un caso real se debe considerar factores como latencias de sincronización, demoras de acceso a memoria, accesos a buses, etc

Entonces, sea Q(P) la suma de todas las sobrecargas.

$$S_P = \frac{T(1)}{T(P) + Q(P)} = \frac{\sum_{i=1}^{m} W_i}{\sum_{I=1}^{m} \frac{W_i}{i} \left\lceil \frac{i}{P} \right\rceil + Q(P)}$$

# Limitación por carga de trabajo fija (Fixed Workload)

Supongamos un caso particular en el que la computadora opera en un modo totalmente secuencial (DOP = 1) o un modo totalmente paralelo (DOP = P)
O sea, $W_i = 0$ para $i \neq 1$ ó $i \neq P$.

Entonces:

$$S_P = \frac{W_1 + W_P}{W_1 + W_P / P}$$

El tiempo de la parte secuencial $W_1$ no cambia, pero la parte paralela $W_P$ se ejecuta en P procesadores y por lo tanto se ejecuta P veces más rápido

Si $W_P = 1-\alpha$ y $W_1 = \alpha$, entonces:

$$S_P = \frac{\alpha + 1 - \alpha}{\alpha + (1-\alpha)/P} = \frac{1}{\alpha + (1-\alpha)/P}$$

que es la ley de Amdahl que expresa la aceleración de rendimiento en una máquina con P procesadores y una parte secuencial α, donde ***$S_p max = 1/ \alpha$*** es la máxima aceleración que se puede esperar.

# Limitación por tiempo fijo (Fixed Execution Time)

**Modelo de Amdahl:**

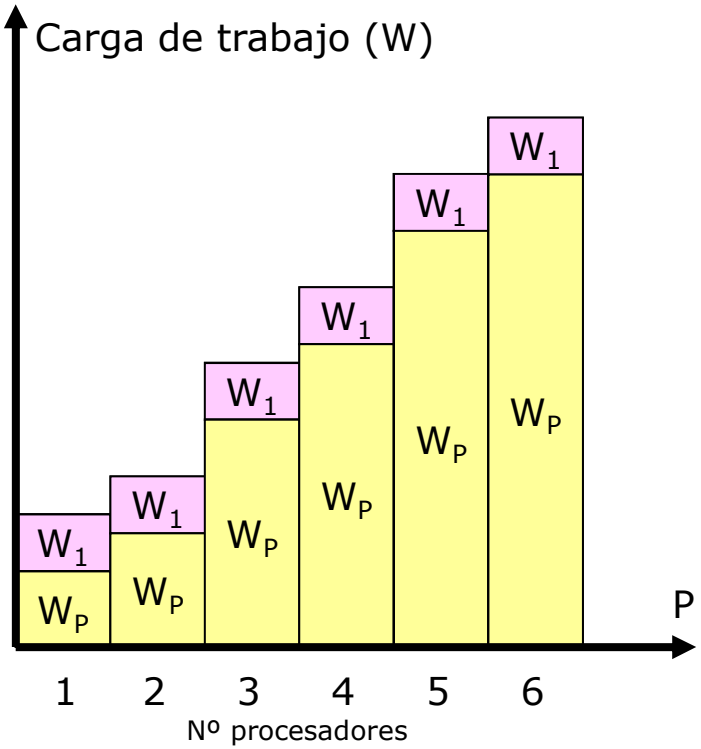
La carga de trabajo permanece fija aún cuando aumento la potencia de cálculo

**Modelo de Gustafson:**

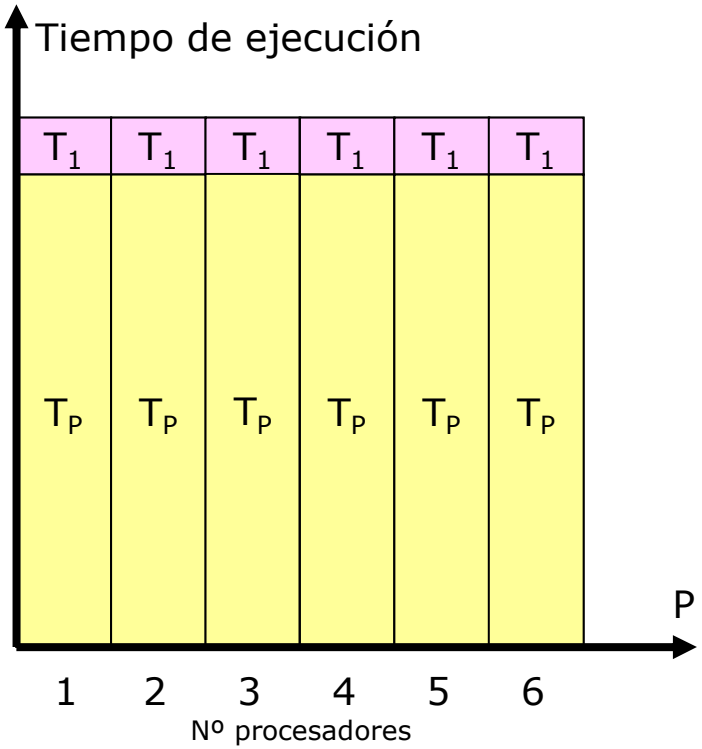
Lo que debe quedar fijo es el tiempo de cálculo…
Entonces, si aumenta la potencia de cálculo, se puede aumentar la ***precisión!!!***

En aplicaciones de precisión crítica, se desea resolver el **problema de mayor tamaño** en una **máquina mayor** en (aproximadamente) el mismo tiempo de ejecución que costaría resolver un **problema menor** en una **máquina menor**

# Limitación por tiempo fijo (Fixed Execution Time)




Carga de trabajo escalada




Tiempo de ejecución fijo

# Limitación por tiempo fijo (Fixed Execution Time)

Sea

m el máximo DOP del problema original
m' el máximo DOP del problema escalado
$W'_i$ la carga de trabajo para DOP=*i*

con

$W'_i > W_i$ para $2 \le i \le m'$ (la carga de trabajo escalada es mayor que la original)
$W'_1 = W_1$

y el supuesto de que T(1) = T'(P)
(Tiempo del problema original sin escalar es igual al tiempo de ejecución del problema escalado)

Entonces:

$$\sum_{i=1}^{m} W_i = \sum_{i=1}^{m} \frac{W_i}{i} \left\lceil \frac{i}{P} \right\rceil + Q(P)$$

Y el speed-up será:

$$S'_P = \frac{T'(1)}{T'(P)} = \frac{T'(1)}{T(1)} = \frac{\sum_{i=1}^{m'} W'_i}{\sum_{i=1}^{m} \frac{W_i}{i} \left\lceil \frac{i}{P} \right\rceil + Q(P)} = \frac{\sum_{i=1}^{m'} W'_i}{\sum_{i=1}^{m} W_i}$$

# Limitación por tiempo fijo (Fixed Execution Time)

En el particular en el que la computadora opera en un modo totalmente secuencial (DOP=1) o un modo totalmente paralelo (DOP=P); o sea $W_i = 0$ para $i \neq 1$ ó $i \neq P$.

el speed-up será:

$$S_P = \frac{W'_1 + W'_P}{W_1 + W_P} = \frac{W_1 + PW_P}{W_1 + W_P}$$

Al aumentar el número de procesadores se aumenta la carga de trabajo en forma proporcional asegurando que el tiempo no varíe.

Si $W_P = 1-\alpha$ y $W_1 = \alpha$, entonces:

$$S_P = \frac{\alpha + P(1-\alpha)}{\alpha + (1-\alpha)} = P - \alpha(P-1)$$

que es la ley de Gustafson permite soportar el rendimiento escalable al aumentar el tamaño de la máquina.

# Limitación por memoria fija (Fixed Memory)

- Modelo de speed-up limitado por memoria
- Generaliza la ley de Amdahl y de Gustafson para minimizar el uso de CPU y memoria
- **Idea básica:** resolver el mayor problema posible limitado por el espacio de memoria
- Aplicable a problemas que involucren cálculos científicos o aplicaciones de ingeniería con modelos de datos que requieren grandes cantidades de memoria
- En la mayoría de las computadoras con capacidad de multiproceso, la memoria crece proporcionalmente al número de nodos (procesadores) de cálculo

# Limitación por memoria fija (Fixed Memory)

Sea

- M la demanda de memoria para un problema dado
- W la carga computacional para dicho problema
- Sean g la relación entre M y W dependiendo de las restricciones de la arquitectura de modo que W = g(M) y M = $g^{-1}(W)$
- $W = \sum_{i=1}^{m} W_i$ la carga para una ejecución secuencial del programa en un único nodo
- $W^* = \sum_{i=1}^{m^*} W_i^*$ La carga para el problema aplicado a P nodos
- m' el máximo DOP del problema escalado

Entonces el speed-up será:

$$S^*_P = \frac{\sum_{i=1}^{m^*} W_i^*}{\sum_{i=1}^{m^*} \frac{W_i^*}{i} \left\lceil \frac{i}{P} \right\rceil + Q(P)}$$

# Limitación por memoria fija (Fixed Memory)

La carga de trabajo para la ejecución secuencial en un solo procesador es independiente del tamaño del problema o del tamaño del sistema.

$$W_1 = W'_1 = W^*_1$$

En ejecución paralela, la mejora de memoria para un sistema de *P* nodos se puede expresar con la ecuación:

$$W^*_P = g^*(PM) = G(P)g(M) = G(P)W_P$$

Siendo G(*n*) el incremento en la carga al aumentar la memoria *n* veces

En el particular en el que la computadora opera en un modo totalmente secuencial (DOP=1) o un modo totalmente paralelo (DOP=P); o sea $W_i = 0$ para i≠1 ó i ≠P.

el speed-up será:

$$S^*_P = \frac{W^*_1 + W^*_P}{W^*_1 + W^*_P / P} = \frac{W_1 + G(P)W_P}{W_1 + G(P)W_P / P}$$

# Limitación por memoria fija (Fixed Memory)

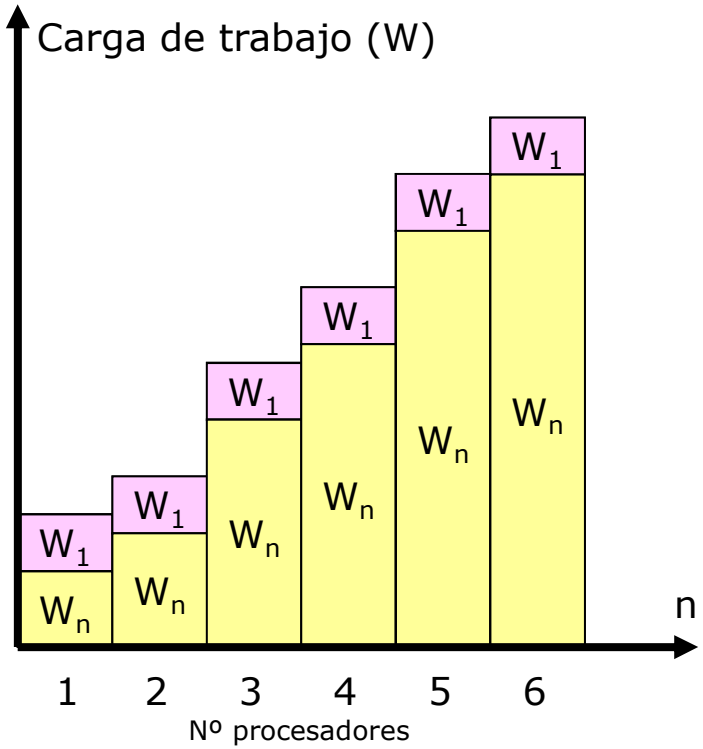
3 casos particulares de aplicación:

- G(P) = 1, entonces el tamaño del problema es fijo y el modelo se corresponde a la ley de amdahl.

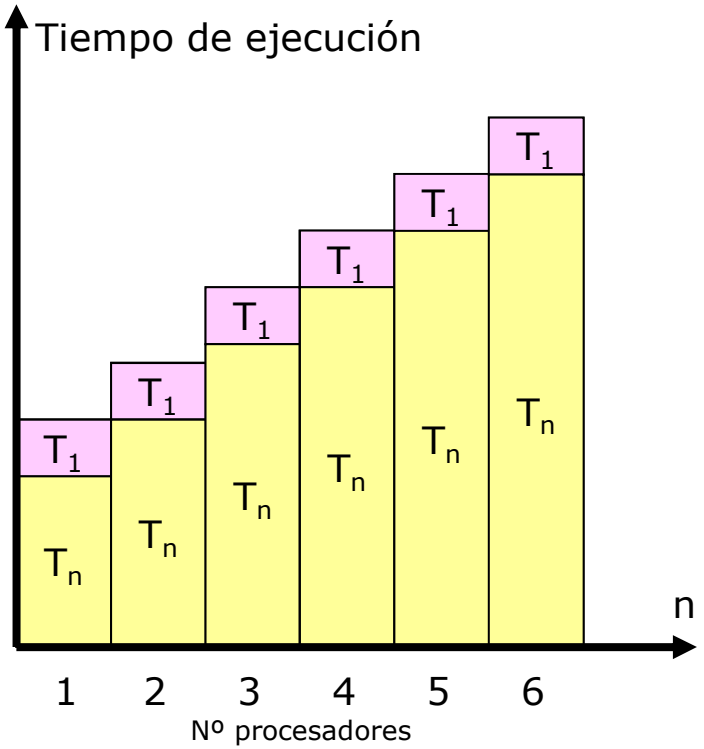
- G(P) = P, se aplica al caso en que la carga se incrementa P veces cuando la memoria se incrementa P veces. En estas condiciones el sistema se comporta según la ley de Gustafson con un tiempo de ejecución fijo.

- G(P) > P, es el caso en que la carga computacional se incrementa más rápido que los requisitos de memoria.

# Limitación por memoria fija (Fixed Memory)




Carga de trabajo escalada




Tiempo de ejecución incrementado

# Resolución de un problema…

¿Cuál es la fracción de código paralelo de un programa secuencial que, ejecutado en paralelo en 8 procesadores tarda un tiempo de 100 ns, durante 50 ns utiliza un único procesador y durante otros 50 ns utiliza 8 procesadores distribuyéndose la carga de trabajo por igual entre los procesadores y despreciando la sobrecarga?.

# Resolución de un problema…

Suponga que un simulador representa una superficie de un objeto por una rejilla n x n. El proceso de simulación consta de dos fases: en la primera se recorre toda la rejilla y en cada nodo ejecuta una subrutina independiente que calcula el valor asignado a ese punto. En la segunda fase de la simulación recorre secuencialmente toda la retícula y suma todos los valores anteriores para calcular la suma global.

a) Estime los tiempos para la primera y segunda fase y calcule el speed-up, si dispone de p procesadores pero la segunda fase es totalmente secuencial.

b) Calcule el Speed-up si se mejora la segunda fase de tal manera que cuando se ejecuta con p procesadores hay dos partes: se calcula la suma intermedia correspondiente a p procesadores y en otra etapa posterior sólo se debe sumar estos valores intermedios.