

Ejercicios.pdf



Anónimo



Ingeniería de Servidores



3º Grado en Ingeniería Informática



**Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación
Universidad de Granada**

Preguntas ISE

Tema 1

- Si el servidor A es un 50% más rápido que el servidor B en ejecutar un determinado programa de benchmark, entonces podemos decir igualmente que el servidor B es un 50% más lento que el servidor A en ejecutar dicho programa de benchmark.

Falso, el servidor B sería un 33% más lento que el servidor A y no un 50%. Si supones que $t_a = 2s$ y $t_b = 3s$, la diferencia de velocidad $AB = (3/2 - 1) * 100 = 50\%$, A es un 50% más rápido que B pero la diferencia de velocidad $BA = (2/3 - 1) * 100 = -33\%$, B es un 33% más lento que A. Diapositiva 33 Tema 1.

- Un determinado proceso monohebra se ejecuta en un servidor durante un tiempo T_o . Sabemos que se hace uso del disco duro durante una fracción f (con $f > 0$) de T_o y que el resto del tiempo transcurre accediendo a la CPU. Si reemplazamos ahora esa CPU por otra el doble de rápida, el nuevo tiempo de ejecución de la hebra pasa a ser T_m con ($T_m < T_o$). Indique si, en este caso, la siguiente afirmación es verdadera o falsa: "Una vez realizada la mejora, la fracción de T_m en la que se usa el disco duro es menor o igual que f ".

Falso, sabemos que hay un proceso determinado que tarda un tiempo $T_o = TiempoDisco + TiempoCPU$. Donde $frecuenciaTiempoDiscoOriginal = TiempoDisco/T_o$.

Ahora se mejora la CPU haciendo que tarde la mitad de tiempo (es el doble de rápida), tenemos que $T_m = TiempoDisco + TiempoCPU / 2$. Entonces $frecuenciaTiempoDiscoMejoraRealizada = TiempoDisco/T_m$.

¿Es $frecuenciaTiempoDiscoMejoraRealizada < frecuenciaTiempoDiscoOriginal$? No ya que $T_m < T_o$ y por tanto, $frecuenciaTiempoDiscoMejoraRealizada > frecuenciaTiempoDiscoOriginal$.

- Una determinada hebra de código se ejecuta en un servidor durante 100s. De esos 100s, el 30% transcurre accediendo al disco duro y el resto transcurre ejecutando instrucciones en la CPU. Si reemplazamos ahora esa CPU por otra el doble de rápida, la hebra se ejecuta en T_m segundos. Indique si es verdadera o falsa la siguiente afirmación: "Tras la mejora, el tiempo que pasa la hebra accediendo al disco duro supone un 60% de T_m "

Falso, tenemos una hebra que tarda 100 segundos, 30 segundos de disco duro y 70 segundos de CPU. La CPU se sustituye por una que tarda la mitad (el doble de rápido). Ahora la hebra tarda $30 + 70/2 = 65$ segundos.

¿Es el tiempo de disco duro el 60%? $30s/65s * 100 = 46\%$, es falso.

- El uso de Cloud Computing y de la virtualización facilita el diseño de servidores escalables.

Verdadero, una manera de afrontar un aumento significativo de la carga es utilizar cloud computing y la virtualización.

- Si la ganancia en velocidad (speedup o aceleración) entre dos servidores para un determinado programa es mayor que 0, es razonable concluir que uno de los servidores es más rápido que el otro para ese programa.

Falso, el speedup es un cociente entre velocidades o tiempos de ejecución. Suponemos que $T_a = 5$ segundos y $T_b = 5$ segundos, el speedup A respecto a B = $T_b/T_a = 1$. El speedup es mayor que 0 y ambos programas duran lo mismo. No representa nada.

Tema 2

- Las memorias con ECC se usan para aumentar la disponibilidad de un servidor.

Falso, las memorias con Error Collecting Code tienen un byte donde se almacenan los códigos errores, que guarda los códigos de error no quiere decir que los evite, solo informa. Esto ayuda a la fiabilidad no a la disponibilidad.

- La frecuencia de reloj de las CPU sigue todavía incrementándose de forma exponencial con los años.

Falso, lo que aumenta de forma exponencial es el número de transistores que tienen pero la frecuencia de reloj no. Si un año está en 3GHz, en un año no va a estar en 30GHz y al siguiente en 300GHz.

- Las DRAM, a diferencia de las SRAM, necesitan refresco.

Verdadero, pero ¿qué es el refresco? Es un proceso periódico en el que se lee información de un área del ordenador y se almacena, normalmente se refresca cada 64 ms.

Pues bien, la DRAM (la memoria RAM de toda la vida) tiene que preguntar cada 64 ms por los datos para ver si ha habido algún cambio. Cada bit de la memoria DRAM tiene un condensador (un dispositivo electrónico que guarda una carga durante un tiempo, pero no de manera infinita, se va descargando). Se sabe que un bit de la memoria de 0 si el condensador está descargado y 1 si está cargado, pero si no se refrescara, al final acabaría todo descargado y perderíamos la memoria. Para que esto no pase, las memorias DRAM deben refrescarse.

Sin embargo, las SRAM están formadas por flips-flops y un circuito biestable formado por 4 o 6 transistores. Este componente no se descarga con el tiempo por lo que no necesita refresco.

- Existen servidores con fuentes de alimentación reemplazables en caliente.

Verdadero, las llamadas hot-swapping, se utilizan para aumentar la disponibilidad del servidor.

- El módulo regulador de voltaje, entre otras cosas, convierte la corriente alterna en corriente continua.

Falso, convertir la corriente alterna a continua es función de la fuente de alimentación. El módulo regulador de voltaje, como bien indica su nombre, regula el voltaje a tensiones menores que necesitan otros componentes, dándoles estabilidad.

- Las unidades de estado sólido son capaces de alcanzar anchos de banda superiores a los que el protocolo SATA-3 puede proporcionar.

Falso, nos está preguntando si existe un ancho de banda para la SSD superior que el que nos da el puerto SATA-3. Sí que lo existe, lo da el protocolo NVMe y se conecta a PCIe.

Principalmente venden 2 tipos de SSD, las de SATA y las de NVMe



La de la izquierda es para SATA y la de la derecha las de NVMe. La de la izquierda se conecta al puerto SATA y tiene la misma forma que un HDD (disco duro). Por ejemplo, puedes quitar tu HDD del portátil y poner este tipo de SSD, también son más baratas que las NVMe. La de la derecha se conecta al puerto PCIe y su protocolo es más rápido.

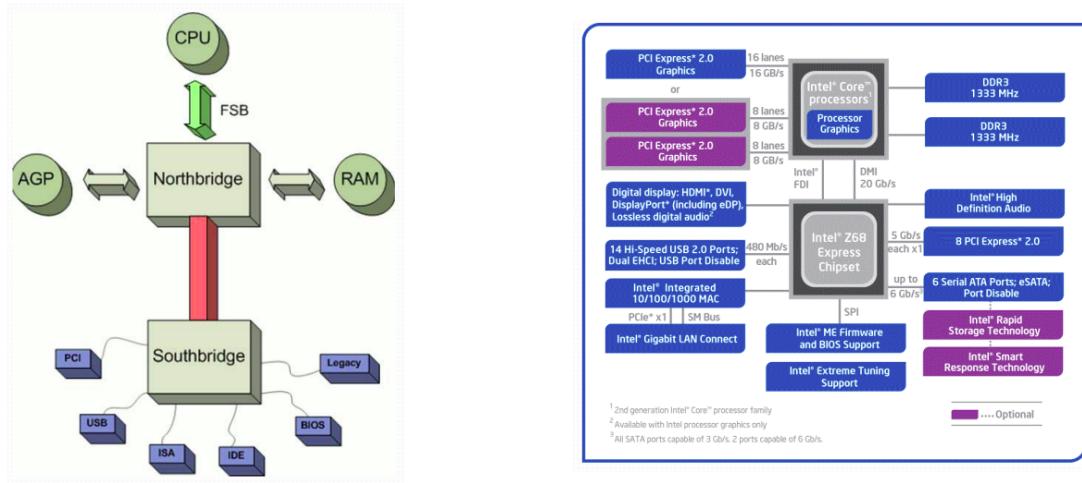
- Time skew es un protocolo de comunicación en paralelo.

Falso, time skew no es un protocolo de comunicación. Time skew es un fenómeno que ocurre cuando enviamos una señal a varios dispositivos conectados en paralelo. Por la distancia entre unos y otros, el material conductor, etc... puede ser que no llegue la señal exactamente en el mismo momento a todos y se provoca este problema.

Sin embargo, si conectamos las cosas en serie pues no ocurre ya que la señal solo le tiene que llegar a un componente. Esto es lo que hace PCIexpress, por lo que logra grandes resultados de ancho de banda, como por ejemplo, que las SSD NVMe sean más rápidas que SATA.

- Es el microprocesador de muchas placas base actuales el que realiza la función de puente norte del chipset.

Verdadero, el puente norte tiene las transferencias rápidas: microprocesador, memoria y gráfica. El puente norte tiene transferencias más lentas con el resto de periféricos. El esquema normal y el actual son los siguientes:

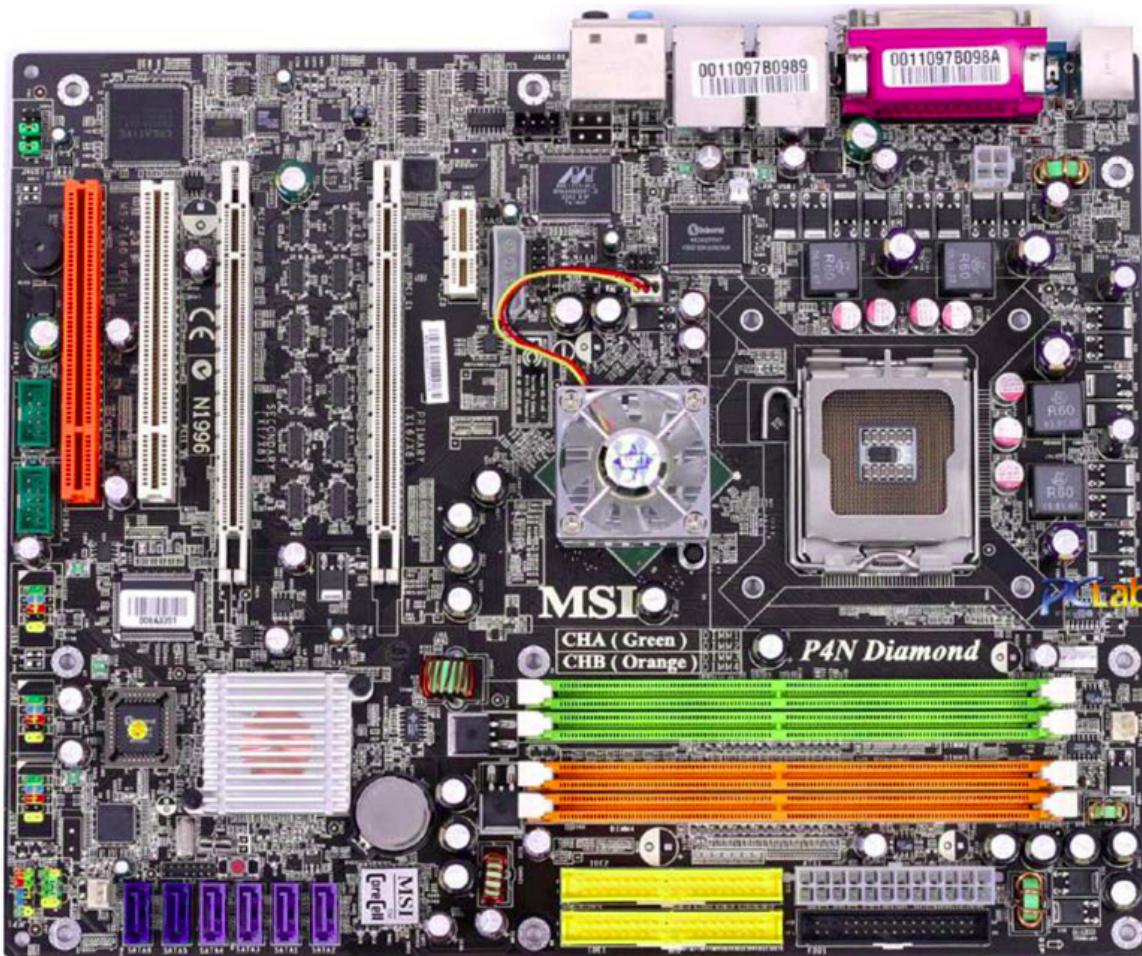


Podemos observar que es el microprocesador el encargado de realizar la función de puente norte.

- Es muy importante en las placas de servidores que los conectores de audio y vídeo del panel trasero sean de altas prestaciones.

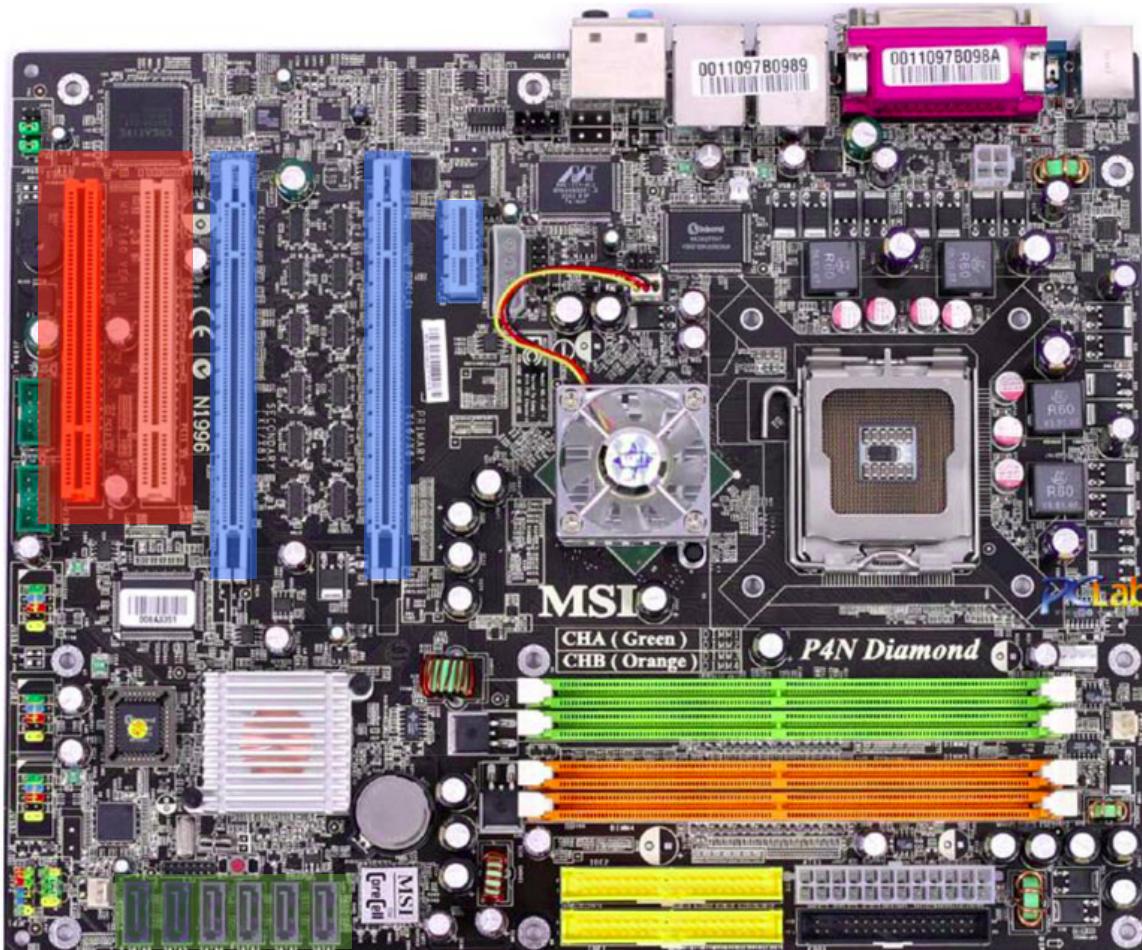
Falso, un servidor está para darle servicio a otros ordenadores, a una tarjeta de sonido excelente y salida a 4k no se le va a dar uso en un servidor.

- La siguiente placa base de la figura tiene dos ranuras PCI, tres PCIe y seis conectores SATA.



Verdadero. Para identificarlos:

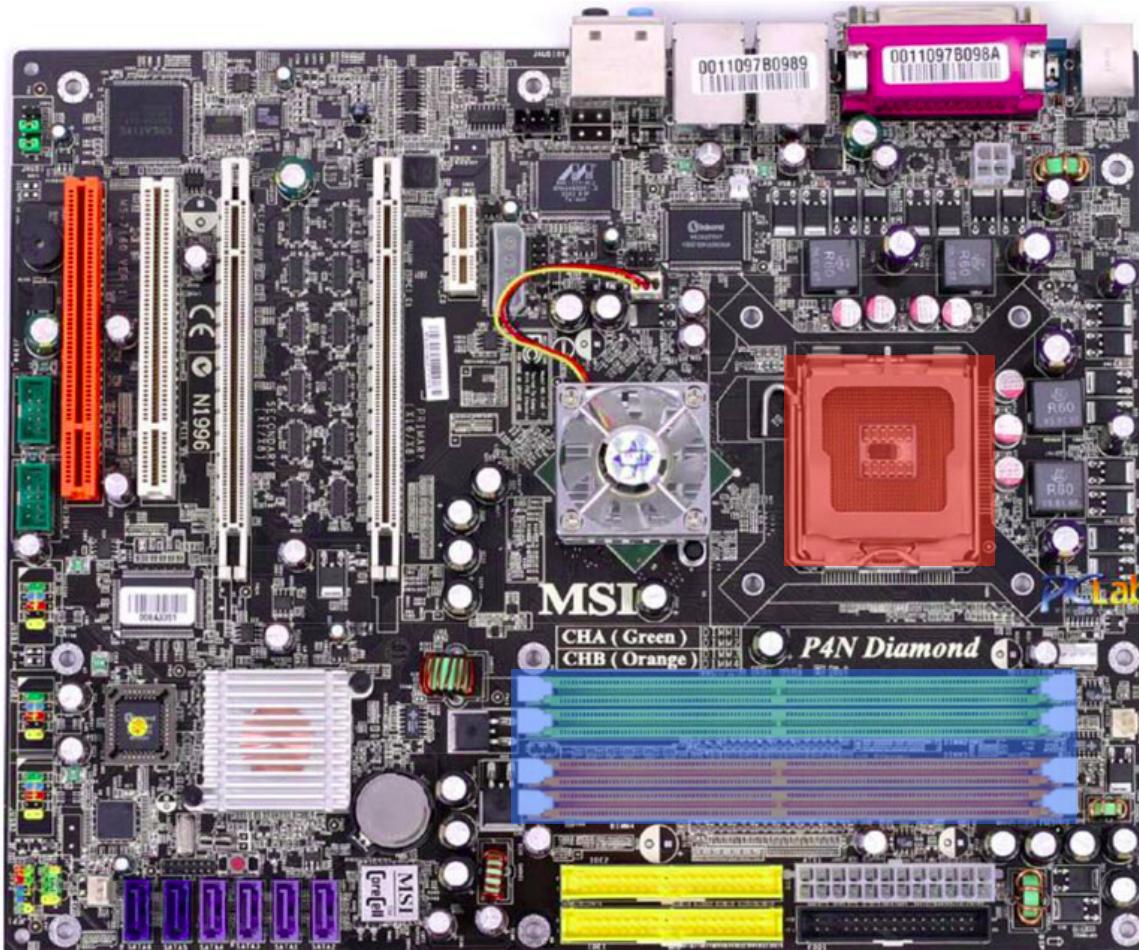
 PCI  PCIe  SATA



- La placa base de la figura anterior solo admite una CPU y cuatro DIMM de memoria RAM dinámica.

Verdadero. Para identificarlos:

 CPU  DIMM



- La transmisión de información entre un módulo de memoria de tipo DDR4 y la CPU es full-duplex.

Falso, en primer lugar, tenemos que saber que es full-duplex y half duplex. Full-duplex es la capacidad de enviar y recibir información a la misma vez. Half-duplex es la capacidad de enviar o recibir información pero no las 2 a la vez.

Pues resulta que las memorias DRAM (la RAM de toda la vida) son half duplex, todas sin excepción, tanto SDRAM como DDR1, DDR2, ...

- PCIe permite la conexión serie punto a punto, una comunicación de tipo full-duplex y la conexión de dispositivos en caliente.

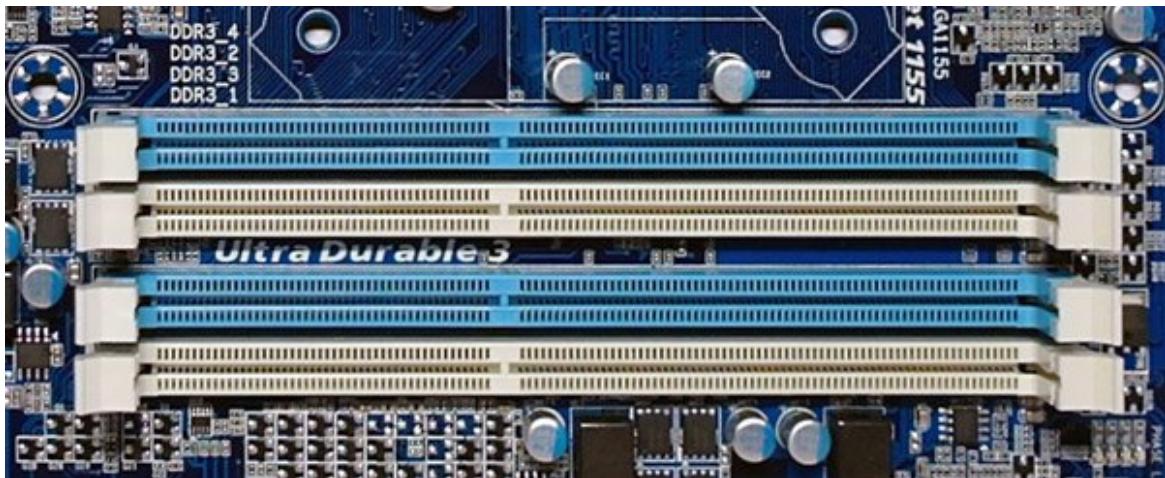
Verdadero, como bien sabemos el puerto PCI es un bus que manda señales en paralelo y esto da problemas de timing skew, para aumentar el rendimiento, surgió PCIe que manda las señales en serie en vez de en paralelo, esto evita timing skew y aumentó su rendimiento. Debido a esto la conexión es serie punto a punto.

PCIe no tiene un bus, está formado por 4 cables llamados LANES, 2 para enviar información y 2 para recibirla. Por esto es full-duplex.

A la interfaz PCIe se le pueden conectar periféricos sin tener que apagar el servidor y tener que encenderlo de nuevo. Permite hot-plugging.

- Las memorias de tipo U-DIMM, al carecer de buffer/registro interno, son las que permiten albergar la mayor cantidad de memoria por módulo.

Vamos a explicar como están distribuidas las DRAM y las definiciones de cada cosa. Para que se entienda bien todo



En primer lugar, tenemos donde se conectan las DRAM a la placa base. Esto se llama **slot** o ranura. Como podemos ver, en la imagen hay 4 slots, 2 azules y 2 blancos. Cabe 1 única memoria DRAM en cada uno de los slots, es decir, en un mismo slot no puedes meter 2 tarjetas DRAM porque todas tienen un mismo tamaño estándar. El slot se refiera al trozo de plástico como tal, para referirnos a la DRAM conectada se le llama **módulo**.

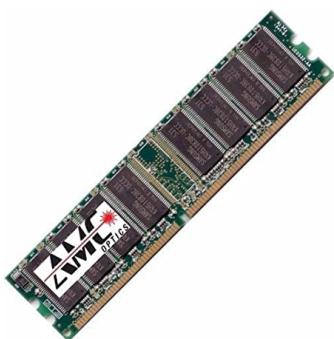
Un **banco de memoria** es una agrupación de slots. Podría darse el caso de que haya 1 banco con 4 slots o 2 bancos con 2 slots cada uno o 4 bancos con un único slot por banco. A simple vista no se puede saber cuántos bancos tiene una placa base, se tendrían que mirar las especificaciones de la placa o de la BIOS. Aparentemente, para que hay 2 bancos con 2 slots cada uno por el hueco que hay en medio, pero no es seguro.

Un **canal de memoria** es el bus (cables) que conectan la CPU con la memoria DRAM. Esto tampoco se puede saber a ciencia cierta hasta que no se miren las especificaciones pero parece que hay 2 canales de memoria, uno para los slots blancos y otro para los slots azules. Suponemos que hay 2 canales.

La CPU puede acceder en paralelo a los canales, es decir, de forma simultánea puede acceder a la DRAM conectada en un slot azul y a la DRAM conectado en un slot blanco.

La CPU no puede acceder en paralelo a dos módulos que están en el mismo canal. Es decir, no puede acceder de forma paralela a los 2 slots azul o a los 2 slots blancos.

Ya sabemos como están distribuidos los cables, los canales y los bancos. Ahora se va a profundizar en el módulo de DRAM como tal (módulo es la tarjeta que se mete en el slot)



Esto es un módulo de memoria.

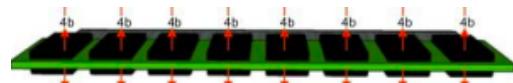
Los módulos de memoria tienen 2 caras. Dependiendo de como estén los pines con los que se conectan a los slots son de distintos tipos:

- ~ SIMM (Single In-line Memory Module): une los pines de ambas caras, dando lugar a una palabra (resultados que devuelve la RAM) de 32 bits.
- ~ DIMM (Dual In-line Memory Module): los pines están separados en cada cara, dando lugar a una palabra de 64 bits.

Los módulos de memoria están formados por **chips**. Los chips son los trozos negros que tiene el módulo de memoria. Puede que estén en una sola cara o en ambas.



Por un lado



Por ambos lados

El primer módulo de memoria tiene 8 chips y el segundo tiene 16 chips.

Los chips se juntan en una agrupación llamada **rango**. Los rangos son independientes de cómo estén distribuidos los chips, es decir, el módulo de la izquierda puede tener 1 rango y el de la derecha también. No porque haya más chips quiere decir que hay más rangos.

La función de un rango es devolver una palabra de 64 bits (72 si tiene ECC). Entonces, suponemos que el módulo de la izquierda es de 1 rango y tiene que devolver la palabra de 64 bits, si tiene 8 chips, cada chip deberá devolver 8 bits.

Si suponemos que el módulo de la derecha tiene 1 rango y tiene que devolver la palabra de 64 bits. Cada chip tendrá $64\text{ bits}/16\text{ chips} = 4\text{ bits por chip}$.

Hasta aquí bien, un módulo tiene chips (por uno o ambos lados, nos la suda) que se agrupan en rangos, cada rango devuelve 64 bits (72 si tiene ECC). Pero, ¿y si un módulo tiene 2 o más rangos? Si tenemos n rangos es como si tuviéramos n módulos distintos.

Si suponemos que el módulo de la derecha tiene 2 rangos y tiene 16 chips, hay 8 chips para cada rango, como cada rango tiene que devolver 64 bits, cada chip dará 8 bits.

La notación para estos módulos son NúmeroRangosRxNúmeroChips. Por ejemplo, un módulo con 8 chips y 4 rangos tendrá notación 4Rx8. Un módulo con 16 chips y 1 rango será 1Rx16.

Si me preguntan cuántos bits debe tener cada chip en 4Rx8, 4 rangos para 8 chips, 2 chips por rango. Cada rango debe dar 64 bits, en total cada chip debe dar 32 bits.

FIN DE LA EXPLICACIÓN para entender estas notaciones asquerosas de las DRAM.
Ahora vamos con el ejercicio.

Falso, el módulo que tiene más memoria es del tipo LR-DIMM, esto se debe a una arquitectura interna que tienen que transforman las señales eléctricas en una sola, dando lugar a más almacenamiento. No obstante, son más lentas y gastan más energía que otras memorias.

- El puente sur del chipset se encarga de las líneas de PCIe x16.

Falso, las líneas de PCIe x16 son las que usan las tarjetas gráficas, y como hemos dicho antes, de los componentes más rápidos se encarga el chipset norte.

- SSD procede de las siglas "Solid State Disk".

Falso, proviene de las siglas Solid State Drive. HDD viene de "Hard Disk Drive", al final siempre es Drive.

- El fenómeno llamado timing skew motivó la aparición de protocolos de comunicación paralelos como P-ATA o PCI.

Falso, precisamente lo que provocaba timing skew eran los protocolos paralelos, esto impulsó que aparecieran protocolos en serie como PCIe o SATA.

- Se presenta la siguiente figura:



- La placa base tiene una ranura PCI y 4PCIe.

Verdadero, para identificarlos:

PCI PCIe



- La placa base tiene al menos dos conectores mini-SAS.

Verdadero, para identificarlos:

 mini-SAS



- La placa base admite hasta dos microprocesadores y un máximo de cuatro DIMM de memoria RAM dinámica en total.

Falso, si que hay hueco para dos microprocesadores pero admite hasta 8 DIMM:

 Microprocesador  DIMM



- El conjunto de instrucciones que ejecutan el auto-test de arranque (Power On self-test) se encuentran almacenadas en las primeras direcciones de la DRAM.

Falso, están en un lugar de la ROM/Flash BIOS. Si la DRAM es volátil como van a estar ahí.

Tema 3

- "sar" es un monitor de actividad por muestreo.

Verdadero, cada T segundos realiza una medida.

~ Despues de instrumentar un programa con la herramienta gprof, se ha obtenido el perfil plano (flat-profile) que aparece en la siguiente tabla.

Self seconds	Calls	Self ms/calls	Total ms/call	Name
X1	8	10	X2	escala
0,4	X3	10	40	ordenada
X4	20	30	X5	inicia

A partir de esta información, indique si las siguientes afirmaciones son verdaderas o falsas:

- $X2 \geq 10 \text{ ms}$

Verdadero, sabemos que Total ms/calls siempre es \geq que Self ms/calls ya que Self ms/calls es el tiempo que tarda en ejecutarse la propia rutina y Total ms/call es el tiempo que tarda en ejecutarse la propia rutina y las subrutinas. Si no hay subrutinas, Total ms/call = Self ms/calls, si tiene subrutinas Total ms/calls $>$ Self ms calls.

- $X_3 = 40$

Verdadero. Sabemos que $\text{Self seconds} = \text{Calls} * \text{Self seconds/calls}$. CUIDADO con las unidades, que estamos en ms/calls. De aquí sacamos que:

$$X_3 = 0.4 \text{ self seconds} * 1000 \text{ ms/seconds} = 400 \text{ self ms} / 10 \text{ self ms/calls} = 40 \text{ calls.}$$

- En Linux, el profiler gprof utiliza monitorización por muestreo para estimar el tiempo de CPU que consume cada función de nuestro programa escrito en C.

Verdadero. Gprof es un profiler por muestreo y estima el tiempo de CPU de cada función/hilo.

~ Se sabe que el monitor de actividad de un determinado servidor consume 6ms de tiempo de CPU y 150KiB de memoria principal cada vez que se activa. Sabiendo de nuestro equipo solo tiene una CPU y 2GiB de memoria principal, indique si las siguientes afirmaciones sobre la carga del monitor sobre diferentes recursos del servidor son verdaderas o falsas:

- Si el monitor de actividad se activa una vez cada minuto, la sobrecarga de la CPU será el 0.1%.

Falso. Pasamos todo a segundos, $6\text{ms} * 1\text{ segundo}/1000\text{ ms} = 0.006\text{s}$.

Se activa 0.006s de cada 60s, entonces la sobrecarga = tiempo monitor/tiempo total * 100.

$$\text{Sobrecarga} = 0.006/60 * 100 = 0.6/60 = 0.01\%$$

- Cada vez que el monitor se activa, la sobrecarga de memoria principal es aproximadamente el 7%.

Falso. CUIDADO con las unidades, para empezar, sabemos que un GiB = 2^{30} bytes, (no confundir con GB que es 10^9 bytes). Pasamos los 2GiB a KiB, $2\text{GiB} * 2^{20} \text{KiB} / 1 \text{GiB} = 2^{21}\text{KiB}$.

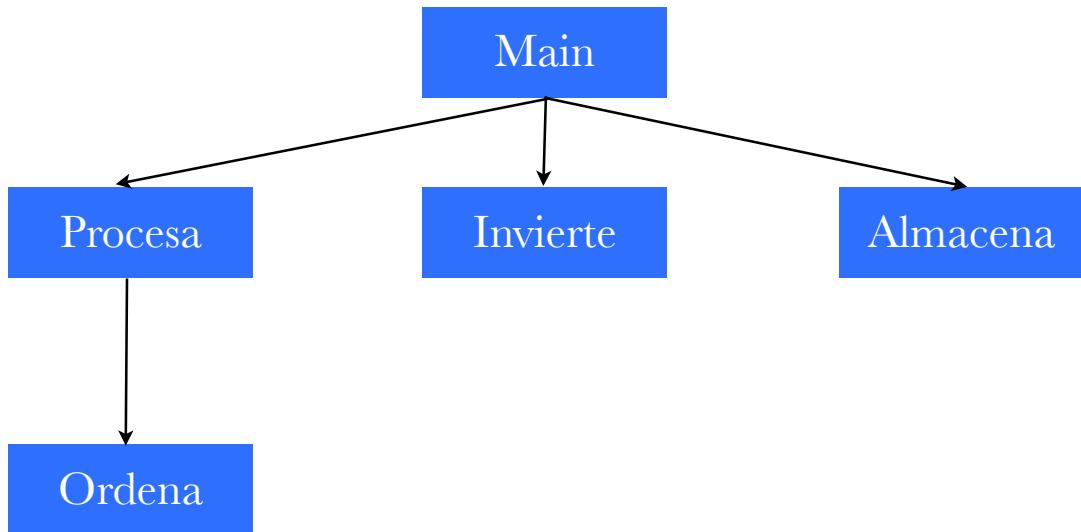
$$\text{Sobrecarga} = 150 \text{ KiB} / 2^{21} \text{KiB} * 100 = 0,007\%$$

~ El resultado de la monitorización de la actividad de una aplicación informática que está siendo ejecutada dentro de un servidor dedicado a streaming de vídeo se muestra a continuación. Como información adicional, el perfil de llamadas indica que todos los procedimientos son llamados únicamente desde el programa principal main (que solo se ejecuta una vez y cuyo tiempo propio de ejecución se puede despreciar), excepto ordena, que solo es llamado desde el procedimiento procesa.

% time	Cumulative seconds	Self seconds	Calls	Self ms/call	Total ms/call	Name
??	??	1,8	??	225	225	ordena
??	??	??	2	900	??	procesa
??	??	1,4	4	350	350	invierte
??	??	??	4	175	??	almacena

- Complete la información no disponible en la tabla ¿Cuánto tiempo de CPU consume la aplicación?

Lo primero que se debe hacer en estos ejercicios a ser posible es dibujar el grafo de llamadas. En el enunciado nos da esta información adicional.



A continuación, vamos a llenar unos campos de la tabla. Sabemos que $\text{Self seconds} = \text{Calls} * \text{Self seconds/calls}$. Entonces, teniendo 2 de estos datos, podemos obtener el otro.

MUCHO CUIDADO con las unidades, que tenemos self ms/call y no self seconds.

% time	Cumulative seconds	Self seconds	Calls	Self ms/call	Total ms/call	Name
??	??	1,8	??	225	225	ordena
??	??	??	2	900	??	procesa
??	??	1,4	4	350	350	invierte
??	??	??	4	175	??	almacena

$$\text{Calls ordena} = 1.8 \text{ self seconds} / (225 \text{ self ms/call} * 10^{-3}\text{ms/s}) = 8 \text{ calls}$$

$$\text{Self seconds procesa} = 2 \text{ calls} * 900 \text{ self ms} * 10^{-3}\text{ms/s} = 1.8 \text{ self seconds}$$

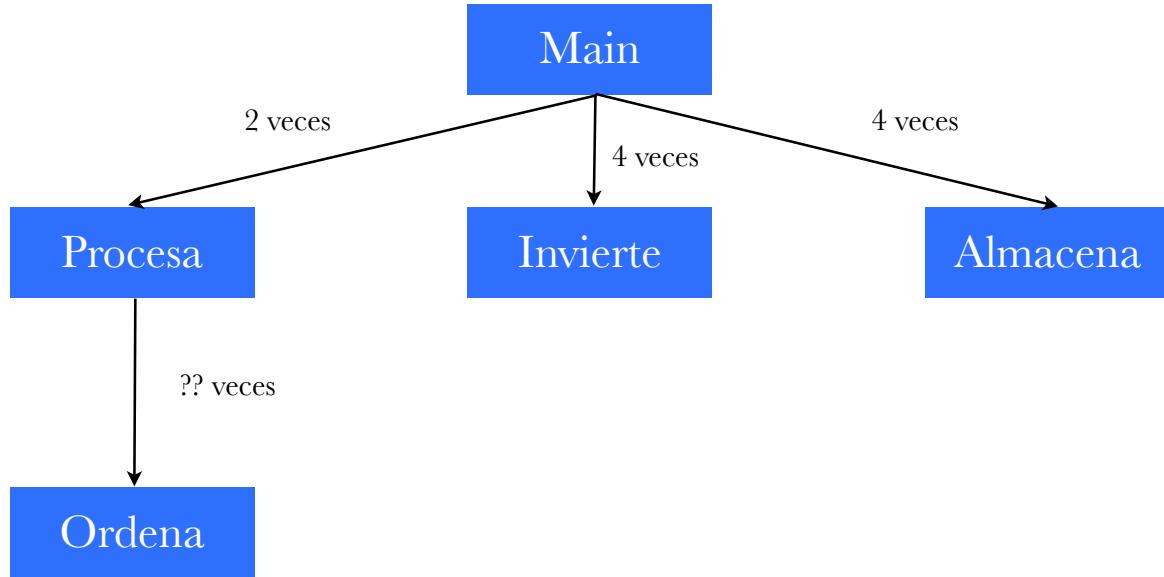
$$\text{Self seconds almacena} = 4 \text{ calls} * 175 \text{ self ms} * 10^{-3}\text{ms/s} = 0.7 \text{ self seconds}$$

% time	Cumulative seconds	Self seconds	Calls	Self ms/call	Total ms/call	Name
??	??	1,8	8	225	225	ordena
??	??	1,8	2	900	??	procesa
??	??	1,4	4	350	350	invierte
??	??	0,7	4	175	??	almacena

Hasta ahora bien, fácil. Ahora viene una parte más complicada que es saber el Total ms/call. Total ms/call indica la cantidad de tiempo que tarda en ejecutarse una rutina y una subrutina, para invierte y almacena todo correcto porque no tienen subrutinas y entonces su self ms/call = total ms/call. Rellenamos almacena.

% time	Cumulative seconds	Self seconds	Calls	Self ms/call	Total ms/call	Name
??	??	1,8	8	225	225	ordena
??	??	1,8	2	900	??	procesa
??	??	1,4	4	350	350	invierte
??	??	0,7	4	175	175	almacena

Nos falta únicamente saber el total ms/call de procesa. Se sabe que es el tiempo que se tarda en ejecutar la rutina y las subrutina. ¿Cuántas veces llama procesa a ordena? Vamos a mirar el grafo.



En total a ordena se le llama 8 veces, si a procesa se la llama 2 veces, entonces cada vez que se llama a procesa, se llama 4 veces a ordena.

Entonces, Total ms/call procesa = self ms/call procesa + 4 * self ms/call ordena.

$$\text{Total ms/call procesa} = 900 \text{ ms} + 4 * 225 \text{ ms} = 1800 \text{ ms}$$

% time	Cumulative seconds	Self seconds	Calls	Self ms/call	Total ms/call	Name
??	??	1,8	8	225	225	ordena
??	??	1,8	2	900	1800	procesa
??	??	1,4	4	350	350	invierte
??	??	0,7	4	175	175	almacena

Ya está hecho el ejercicio, lo que queda es fácil. Para llenar la columna cumulative seconds, se van acumulando los tiempos de self seconds

% time	Cumulative seconds	Self seconds	Calls	Self ms/call	Total ms/call	Name
??	1,8	1,8	8	225	225	ordena
??	3,6	1,8	2	900	1800	procesa
??	5	1,4	4	350	350	invierte
??	5,7	0,7	4	175	175	almacena

El tiempo consumido por la CPU es 5.7. Por último, para completar el %time, debemos hacer un porcentaje de cada self seconds respecto al tiempo total (5,7 segundos).

% time	Cumulative seconds	Self seconds	Calls	Self ms/call	Total ms/call	Name
31	1,8	1,8	8	225	225	ordena
31	3,6	1,8	2	900	1800	procesa
24	5	1,4	4	350	350	invierte
14	5,7	0,7	4	175	175	almacena

- Determine la ganancia en velocidad (speedup) que se obtendría si reemplazamos el procedimiento ordena por otro 3 veces más rápido. Exprese esa ganancia también como tanto por ciento de mejora

La función ordena tarda 3 veces menos, ahora en vez de tardar 225 self ms/call ahora tarda 75 ms. El tiempo que tardará self seconds ordena = 8 calls * 75 self ms * 10^{-3} s/ms = 0.6 self seconds.

El tiempo total que tarda en ejecutarse el programa será $0.6 + 1.8 + 1.4 + 0.7 = 4.5$ segundos, con respecto al tiempo anterior que es 5.7 segundos.

La ganancia en velocidad (speedup) mejorado respecto no mejorado = $\text{VelocidadMejorado}/\text{VelocidadNoMejorado} = \text{tiempoNoMejorado}/\text{tiempoMejorado} =$

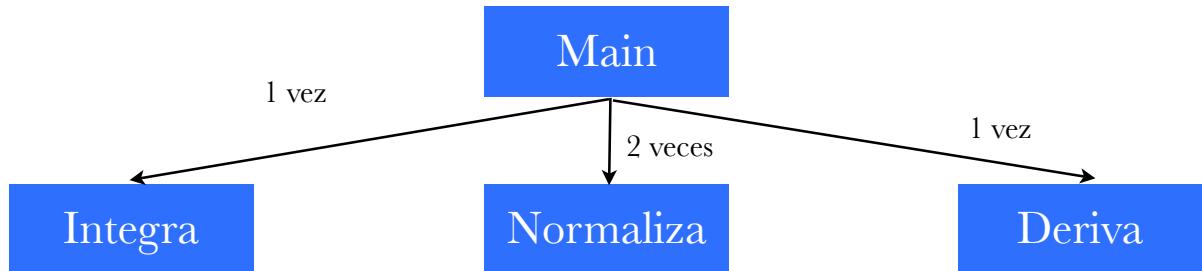
$5.7/4.5 = 1.26$, para representarlo como un porcentaje $(\text{Speedup} - 1) * 100 = 26\%$ más rápido.

- Dado el siguiente perfil de llamadas, obtenga el grafo de llamadas.

index	called	name	
[1]		main	[1]
	1/1	integra	[3]
	2/2	normaliza	[5]
	1/4	deriva	[2]
<hr/>			
	1/4	main	[1]
	3/4	integra	[3]
[2]	4	deriva	[2]
	12/25	redondea	[4]
<hr/>			
	1/1	main	[1]
[3]	1	integra	[3]
	3/4	deriva	[2]
	3/25	redondea	[4]
<hr/>			

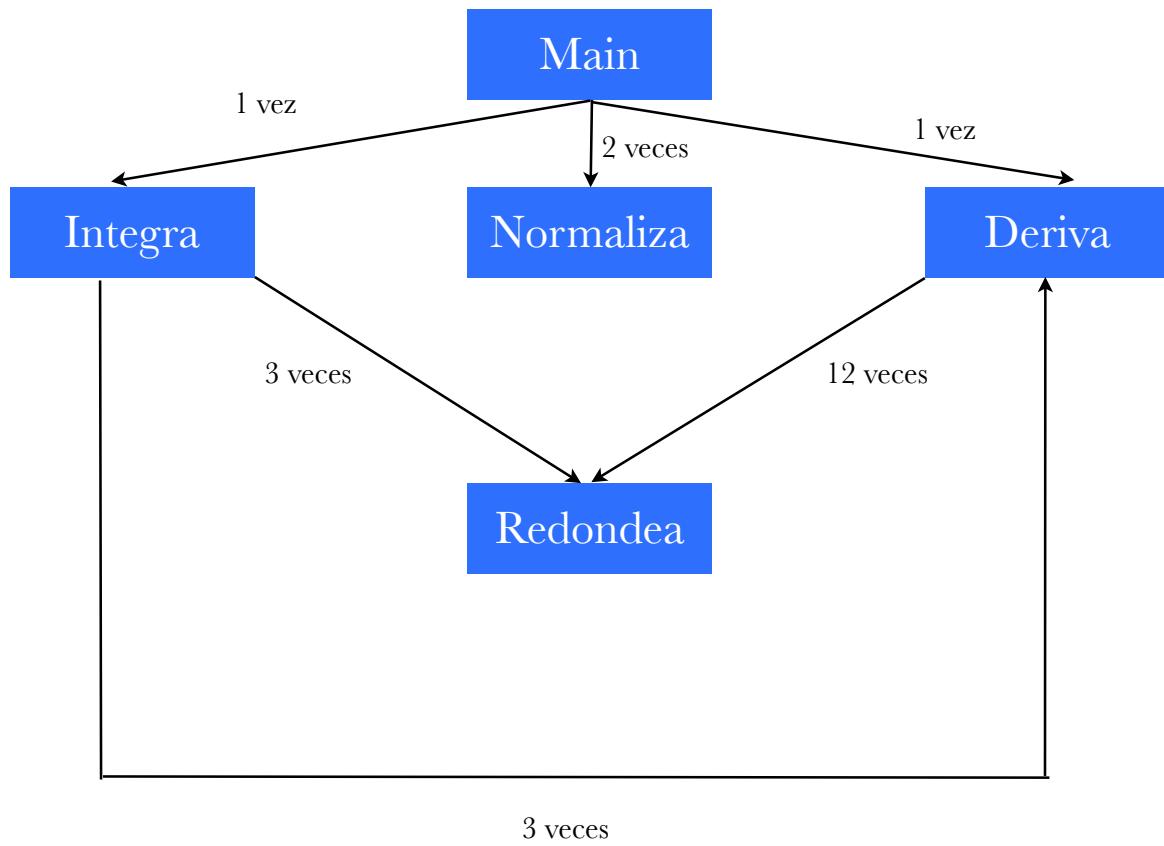
index	called	name	
	3/25	integra	[3]
	10/25	normaliza	[5]
	12/25	deriva	[2]
[4]	25	redondea	[4]
<hr/>			
	2/2	main	[1]
[5]	2	normaliza	[5]
	10/25	redondea	[4]
<hr/>			

Podemos observar que hay 5 funciones, main, integra, normaliza, deriva, redondea. Se sabe que la primera función que se llama es el main que llama 1 vez a integra, 2 a normaliza y 1 a deriva.

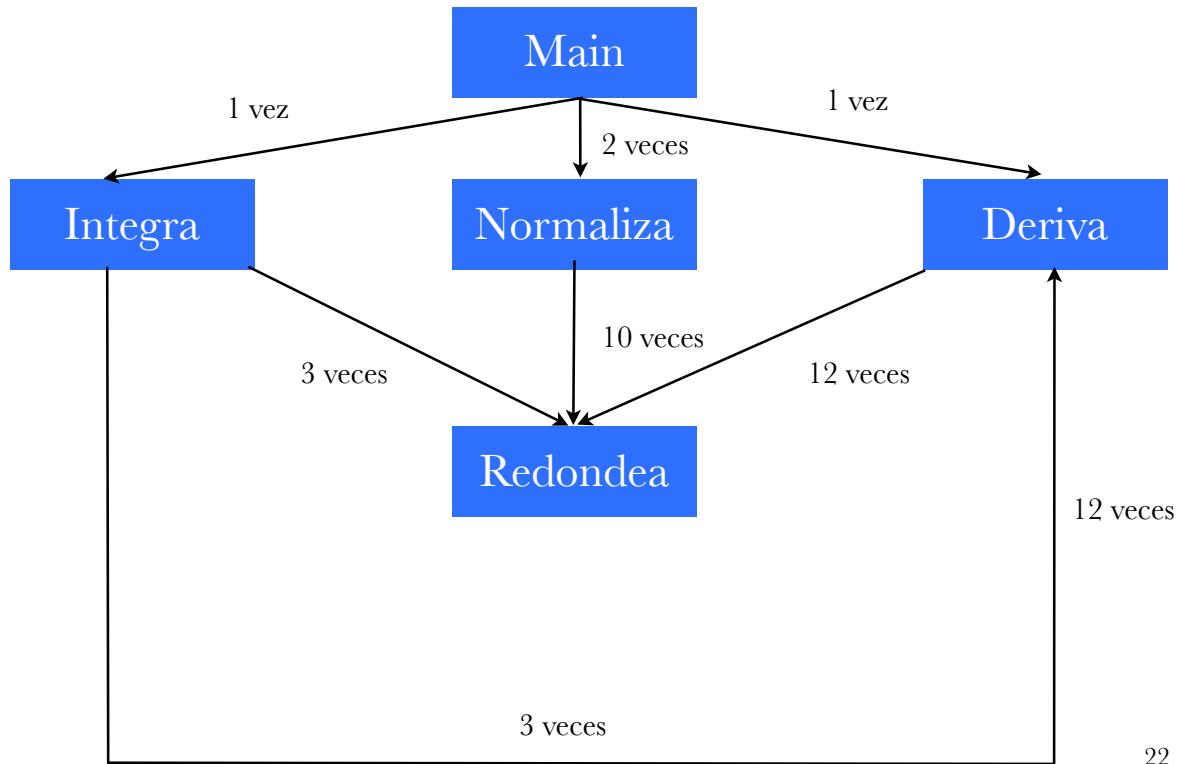


A continuación, vemos que la función deriva es llamada 1 vez por el main, 3 veces por integra y llama 12 veces a la función redondea.

Además, la función integra llama 3 veces a redondea.



Para finalizar seguimos con la función redondea, que como bien ya sabemos, es llamada 3 veces por integra, 12 por deriva y además, 10 veces por normaliza.



Tema 4

- Si el servidor A es el doble de rápido que el servidor B para todos los programas de un benchmark cuyo rendimiento se calcula según el criterio SPEC, entonces ese índice SPEC del servidor A será mayor que el del servidor B, independientemente de la máquina de referencia elegida.

Verdadero, en primer lugar tenemos que saber que el índice SPEC, se calcula de la siguiente forma:

$$SPEC\ A = \sqrt[n]{\frac{Tref1}{TMaquinaA1} * \frac{Tref2}{TMaquinaA2} * \dots * \frac{Trefn}{TMaquinaAn}}$$

Cuanto menos tarde la máquina A, mayor valor tendrá el índice SPEC. El índice SPEC cuánto más alto, más rápida ha sido la máquina. Vamos a hacer la demostración de que $SPEC\ A > SPEC\ B$ independientemente de la máquina de referencia.

El índice SPEC se puede representar de esta forma.

$$SPEC\ A = \frac{\sqrt[n]{Tref1 * Tref2 * \dots * Trefn}}{\sqrt[n]{TMaquinaA1 * TMaquinaA2 * \dots * TMaquinaAn}}$$

Llamamos al numerador TReferenciaTotal y al denominador TMaquinaATotal. Pues bien, como nos indica el enunciado $TMaquinaATotal < TMaquinaBTotal$ y estos valores no cambian. Independientemente del TReferenciaTotal que haya, TMaquinaATotal siempre será menor que TMaquinaBTotal y su índice SPEC será mayor.

~ En Google están intentando mejorar la técnica de distribución de carga de sus servidores de YouTube. Para ello, han realizado 100 medidas de la productividad media de los servidores durante un número determinado pero fijo, de horas para las 2 configuraciones principales de distribución de carga: Conf1 y Conf2. Como los experimentos se han realizado en presencia de aleatoriedad, ha realizado un test-t cuyos resultados son:

Measure 1	Measure 2	t	df	p	Mean difference	90% CI for Mean Difference	
						Lower	Upper
Conf1 -	Conf2	113	99	0.91	0.88	-19.5	21.3

A partir de esta información, indique si las siguientes afirmaciones son verdaderas o falsas:

- Para un 80% de nivel de confianza podemos afirmar que hay diferencias significativas y que la mejor configuración, según el criterio de la media aritmética es Conf1.

Falso, para un 80% de confianza, alfa = $(100 - 80) / 100 = 0.2$. El p valor es 0.91 que es mayor que 0.2, entonces no podemos rechazar la hipótesis de que los rendimientos sean iguales.

- Para un 99% de nivel de confianza no hay diferencias significativas entre las productividades medias obtenidas por ambas configuraciones.

Verdadero, para un 99% de confianza, alfa = $(100 - 99) / 100 = 0.01$. El p valor es 0.91 que es mayor que 0.01, entonces no podemos rechazar la hipótesis de que los rendimientos sean iguales que es lo que dice el enunciado.

Si en el ejercicio anterior no se podían demostrar diferencias con una confianza para el 80% como se van a poder mostrar con un 99%.

- Con un benchmark especializado en DRAM puedo diagnosticar el correcto funcionamiento de un módulo de DRAM.

Falso, la finalidad de un benchmark es comparar características de rendimiento, no de un diagnóstico para comprobar que un componente funciona correctamente. Por ejemplo, un benchmark puede ver la velocidad con la que se leen los datos de la DRAM pero no comprueba si los datos leídos son los correctos.

- La hipótesis de partida de un test ANOVA es que el factor que se está estudiando influye en el rendimiento.

Falso, los factores que se estudian en el test ANOVA son aquellos que CREEMOS que pueden afectar pero pueden no afectar y realizarse el test ANOVA.

- Cuando comparamos tiempos de ejecución, expresados en segundos, de programas ejecutados en servidores utilizando el test t, el estadístico texp también se puede expresar en segundos.

Falso, texp no tiene unidades.

- La tabla que se muestra a continuación refleja los tiempos de ejecución, en segundos de los 14 programas de prueba que integran un determinado benchmark empleado para el cálculo del rendimiento en aritmética de coma floatante. En particular, los tiempos corresponden a la máquina de referencia y a una máquina que denominaremos A

Programa	Referencia	A-Base	A-Peak
168.wupwise	1600	419	300
171.swim	3100	562	562
172.mgrid	1800	607	607
173.applu	2100	658	605
177.mesa	1400	273	242
178.galgel	2900	571	571
179.art	2600	1040	1038
183.equake	1300	501	387
187.facerec	1900	434	434
188.ammp	2200	705	697
189.lucas	2000	784	758
191.fma3d	2100	534	534
200.sixtrack	1100	395	336
301.apsi	2600	866	866

- a) Calcula los índices SPECfp_base y SPECfp de la máquina A según el criterio de SPEC.

La fórmula para calcular el SPEC es la siguiente. La diferencia entre SPECfp_base y SPECfp es que el primero se calcula con los tiempos base y el segundo con los peak

$$SPEC\ A = \sqrt[n]{\frac{Tref1}{TMaquinaA1} * \frac{Tref2}{TMaquinaA2} * \dots * \frac{Trefn}{TMaquinaAn}}$$

SPECfp_base = 3.48

SPECfp_base = 3.72

- b) Para la columna A-Base, si se considera el tiempo total de ejecución, ¿cuántas veces es más rápida la máquina A que la máquina de referencia?

La máquina A-Base tarda en total en ejecutarse 8349 segundos y la máquina de referencia tarda 28700. La máquina A es $28700/8349 = 3.43$ veces más rápida que la máquina de referencia.

- c) ¿Qué mejora del rendimiento se obtiene utilizando las opciones de optimización que ofrece el compilador?

La máquina A-Base tarda en total en ejecutarse 8349 segundos y la máquina de A en peal tarda en ejecutarse 7937 segundos. La mejora que ofrece es de $8349/7937 = 1.05$ o un 5% de mejora

- La empresa Facebook está estudiando dos grandes propuestas con el objetivo de actualizar los computadores personales de su oficina principal en Menlo Park, California. El precio de cada computador es de 1850€ para el Modelo A y 2200€ para el Modelo B. Los responsables informáticos de la empresa han ejecutado los ocho programas que utilizan habitualmente en un computador de cada propuesta, y han obtenido los tiempos de ejecución, expresados en segundos, que se muestran a continuación:

Programa	Modelo A	Modelo B
1	23,6	24,0
2	33,7	41,6
3	10,1	8,7
4	12,9	13,5
5	67,8	66,4
6	9,3	15,2
7	47,4	50,5
8	54,9	52,3

Determine, para un nivel de confianza del 95%, si existen diferencias significativas en el rendimiento de los computadores personales de las dos propuestas y qué opción sería mejor. DATO: $|t_{0,025}, 7| = 2,365$.

En primer lugar, tenemos que ver si existen diferencias significativas al nivel de confianza del 95%. Si no existen diferencias significativas, la mejor opción será el modelo A ya que es el más barato.

Vamos a comparar la máquina A con respecto a la B, comenzaremos calculando la media muestral.

Programa	Modelo A	Modelo B	$d = ta - tb$
1	23,6	24	$23,6 - 24 = -0,4$
2	33,7	41,6	$33,7 - 41,6 = -7,9$
3	10,1	8,7	$10,1 - 8,7 = 1,4$
4	12,9	13,5	$12,9 - 13,5 = -0,6$
5	67,8	66,4	$67,8 - 66,4 = 1,4$
6	9,3	15,2	$9,3 - 15,2 = -5,9$
7	47,4	50,5	$47,4 - 50,5 = -3,1$
8	54,9	52,3	$54,9 - 52,3 = 2,6$
			$dMediaMuestral = -1,5625$

A continuación, se calcula la desviación típica que tiene la siguiente fórmula. Mucho CUIDADO que aquí nuestra media muestral (d) es negativa y al restar un negativo se suman.

$$s = \sqrt{\frac{\sum_{i=1}^n (d_i - \bar{d})^2}{n - 1}}$$

$$s = \sqrt{\frac{(-0.4 + 1.56)^2 + (-7.9 + 1.56)^2 + \dots + (2.6 + 1.56)^2}{8 - 1}}$$

$$s = 3,75$$

Para terminar, tenemos con la media muestral, la desviación típica y el número de muestras, calculamos la texp.

$$t_{exp} = \frac{\bar{d}}{s/\sqrt{n}}$$

$$t_{exp} = -1,5625 / (3,75 / \text{sqrt}(8)) = -0.23$$

Una vez sabemos la t experimental solo nos queda saber si rechazamos o no la hipótesis nula, nos han dado como dato del ejercicio que $|t_{0,025}, 7| = 2,365$, que es el intervalo superior de confianza para un 95% con 8 muestras. El intervalo de confianza es [-2.365, 2.365].

Como nuestro t_{exp} (-0.23) pertenece a ese intervalo, no podemos descartar la hipótesis nula y quiere decir que no hay diferencias significativas al 95% de confianza y por tanto, la opción mejor es la A ya que es la más barata.

- En la empresa KANDOR GRAPHICS están intentando mejorar la técnica de distribución de carga de su servidor principal de streaming de vídeo. Para ellos, han realizado cinco medidas de la productividad media del servidor durante un número determinado, pero fijo, de horas para las 2 configuraciones principales de distribución de carga: Least Connected y Round Robin. Los resultados, expresados como número media de MB transmitidos por segundo son los que aparecen a continuación:

Nº Experimento	Least_Connected (MB/s)	Round_Robin (MB/s)
1	157	165
2	125	123
3	172	185
4	152	158
5	165	172

Para poder estar seguros de la decisión, los ingenieros informáticos de la empresa realizaron un test t sobre estos datos, obteniéndose los siguientes resultados:

Paired Samples Test

	Paired Differences					t	df	p			
	Mean	Std. Deviation	Std. Error Mean	90% Confidence Interval of the Difference							
				Lower	Upper						
Least_Connected - Round_Robin	-6,40	5,413	2,421	-11,56	-1,24	-2,64	4	0,057			

a) A la vista de los resultados y para un 90% de confianza ¿Qué método utilizaría y por qué?

Sabemos que para un 90% de confianza $\alpha = (100 - 90)/100 = 0.1$, ahora lo comparamos con el p valor = 0.057, como el p valor < α entonces sí que hay diferencias significativas.

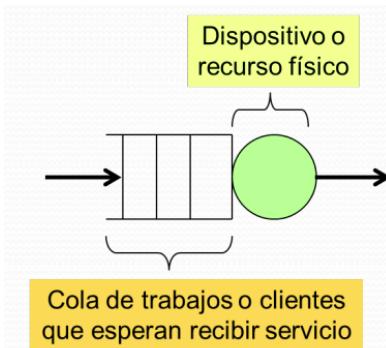
CUIDADO, tenemos que elegir aquel que nos dé MÁS MB/s (normalmente se mide en segundos y se coge el de menor valor, en este caso es al revés). El que más nos da es Round Robin.

Tema 5. CONSEJO PARA APROBAR: aprenderte la hoja de fórmulas de cabeza y escribirla en un folio al principio del examen. Si no te sabes el tema 5 no apruebas.

Antes de comenzar los ejercicios, se va a dar una explicación de las variables y como se calculan.

En los servidores existen los dispositivos: una CPU, disco, la RAM, ... A estos dispositivos le llegan una serie de peticiones de trabajos que tienen que realizar. El problema es que estos dispositivos solo pueden ejecutar un trabajo a la vez, por lo que hay una cola de espera de los trabajos para cuando el dispositivo se quede libre, acceder a él.

El dispositivo y su cola se conocen como estación de servicio, es así:



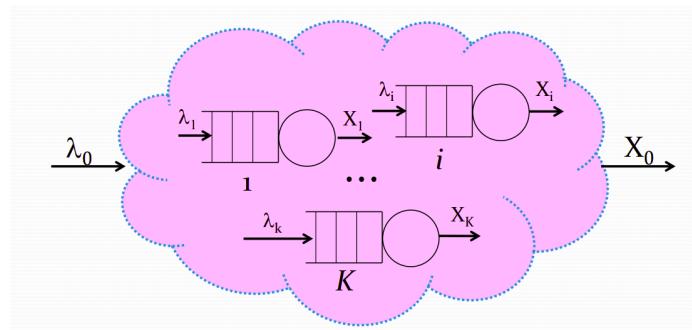
La estación de servicio va a monitorizar, de dónde se sacan una serie de variables:

- ~ T: Periodo de tiempo durante el que se monitoriza la estación de servicio.
- ~ W: Waiting time, tiempo que un trabajo está en la cola de espera.
- ~ S: Service time, tiempo medio que un trabajo está en el dispositivo.
- ~ R: Response time, tiempo desde que un trabajo entra hasta que sale.
- ~ A: Arrivals, número de trabajos que llegan a la estación.
- ~ B: Busy time, tiempo que el dispositivo está ocupado.
- ~ C: Completions, número de trabajos que salen de la estación.
- ~ λ : Número de trabajos medios que llegan en un periodo de tiempo.
- ~ X: Productividad, número de trabajos medio que se completan en un periodo de tiempo.

- ~ U: Utilización media del recurso, de todo el tiempo que se monitoriza, cuánto se está usando el dispositivo.
- ~ N: número medio de trabajos que hay en una estación de servicio (cola + dispositivo).
- ~ Q: número medio de trabajos que hay en una cola de una estación de servicio.
- ~ V: razón media de visitas, de todos los trabajos que se completan en el servidor, cuántos completa una estación de servicio en concreto.
- ~ D: demanda de servicio, cantidad de tiempo que una estación de servicio le dedica a cada trabajo.

Un servidor tiene K estaciones de servicio. Tiene una o varias RAM, CPU, Disco, etc.

Una representación sería la siguiente:



Un servidor también se comporta como una estación de servicio a gran escala, y las mismas variables que tiene una estación de servicio las tiene un servidor. La notación para los servidores será un $_0$ al lado de las variables mientras que para las estaciones de servicio se denotarán con un i . Por ejemplo, λ_0 será el número de trabajos medio que llegan al servidor y λ_i será el número de trabajos medio que llegan a una estación de servicio.

El equilibrio de flujo quiere decir que si el periodo de muestreo es muy largo, las entradas y salidas se tienden a igualar. Por ejemplo, $A = C$ o $\lambda = X$

Ahora viene donde se hacen 500 combinaciones entre las variables y se les da un nombre super chulo.

- ~ $R = W + S$, evidentemente el tiempo total será el que está en la cola más el tiempo que se usa el dispositivo.

- ~ $\lambda = A/T$, número de procesos que llegan entre el tiempo que se monitoriza.
- ~ $X = C/T$, número de procesos que salen entre el tiempo que se monitoriza.
- ~ $S = B/C$, cuánto tiempo está ocupado el dispositivo entre cuántos procesos se completan.
- ~ $U = B/T$, cuánto tiempo está ocupado el dispositivo entre el tiempo total que se monitoriza.
- ~ $N = Q + U$, los trabajos totales de una estación son los que hay de media en la cola más la media que se sirven.
- ~ $V_i = C_i/C_o$ de todos los trabajos completados por el servidor, cuántos ha completado esa estación.
- ~ $D_i = B_i/C_o$ de todos los trabajos completados por el servidor, cuánto tiempo le ha dedicado esa estación. Sustituyendo variables también tenemos que $D_i = V_i * S_i$

Ley de Little

- ~ $N_o = \lambda_o * R_o = (\text{si equilibrio de flujo}) X_o * R_o$
- ~ $N_i = \lambda_i * R_i = (\text{si equilibrio de flujo}) X_i * R_i$
- ~ $Q_i = \lambda_i * W_i = (\text{si equilibrio de flujo}) X_i * W_i$

Ley de la Utilización

- ~ $U_i = X_i * S_i = (\text{si equilibrio de flujo}) \lambda_i * S_i$
- ~ $U_i = X_o * D_i = (\text{si equilibrio de flujo}) \lambda_o * D_i$

Ley general del tiempo de respuesta

- ~ $R_o = V_1 * R_1 + V_2 * R_2 + \dots + V_k * R_k$

Ley general del tiempo de respuesta interactivo

- ~ $N_t = N_o + N_z$
- ~ $N_z = X_o * Z$
- ~ $R_o = N_t/X_o - Z$

Vamos a comenzar con los ejercicios. Mucha calma.

- Una red de colas abiertas se puede considerar un caso particular de red de colas cerrada si hacemos que $Z = 0$.

Falso, una red de colas cerradas es una red por donde recirculan un número constante de trabajos mientras que una red abierta los procesos llegan de una fuente externa que no controlamos. Haya o no haya tiempo de reflexión (Z), sigue siendo una red de colas cerrada. Lo único es que es de tipo batch y no interactivo.

- Si aplicamos la ley de Little a los usuarios en reflexión de una red de colas cerrada interactiva, podemos relacionar el número medio de usuarios en reflexión con la productividad media del servidor y el tiempo medio de reflexión de dichos usuarios.

Verdadero, una red de colas cerradas interactiva es aquella en la que los usuarios tienen un tiempo de reflexión (Z , tienen que poner algo en Entrada o Salida, darle a aceptar, etc).

Podemos ver en nuestro resumen de fórmulas que $N_z = X_o * Z$, pregunta dura, saberte una fórmula perdida en una diapositiva.

- Si un servidor web ha recibido una media de 10 visitas por segundo, entonces la razón media de visitas del servidor es 10 tr/s.

Verdadero, nos están preguntando la razón media de visitas del servidor, V , como bien sabemos es $V_i = C_i / C_o$, pero la del servidor sería $V_o = C_o / C_o = 1$, la razón de visita de un servidor NO EXISTE, existe la razón de visita de una estación de servicio.

Si recordamos, la razón de visita es de todos los trabajos que se completan en el servidor, cuántos completa una estación de servicio. Es evidente que los trabajos que completa el servidor son todos los trabajos que se completan en el servidor.

El dato que nos están dando de 10 tr/s es la tasa media de llegada al servidor λ_o .

- Si añadimos una segunda CPU a nuestro servidor, idéntica a la ya existente, es razonable suponer que la razón media de visita de la primera CPU se va a dividir por dos.

Verdadero, nos preguntan otra vez por la V (razón media de visitas), que es del total de trabajos completados en el servidor, cuántos se resuelven en una estación de servicio.

Si una CPU antes completaba 40 trabajos del total y ahora se pone una exactamente igual, pues cada CPU completará 20 trabajos del total.

- $W_i = N_i * S_i$ es una ley operacional válida para servidores modelados mediante una red de colas abierta en equilibrio de flujo.

Falso, esa relación es cierto que se cumple pero no es una ley operacional. Esta es otra pregunta criminal, tienes que saber de todas las combinaciones variables cuales son leyes operacionales.

~ Se ha monitorizado durante 1000s un servidor de base de datos no saturado con el fin de obtener un modelo del mismo basado en redes de colas. En dicho modelo sólo aparecen dos componentes: CPU y disco duro. Como resultado de dicha monitorización, se han obtenido las siguientes medidas:

1. El servidor ha completado un total de 10 000 consultas.
2. El tiempo medio de respuesta de la CPU es 0.25s.
3. La utilización media del disco duro es el 38%.
4. En total, el disco duro ha atendido durante ese intervalo de tiempo 38 000 peticiones de lectura/escritura.
5. El tiempo medio de espera en la cola del disco duro es de 0.75s.

A partir de esta información, indique si las siguiente afirmaciones son verdaderas o falsas:

- La razón media de visita del disco duro es 4, ya que debe ser un número entero y 4 es el número más cercano a 3.8.

Falso, la razón media de visita del disco duro es la $V_{\text{discoDuro}}$, de todos los trabajos completados, cuántos ha completado el disco duro. $V_{\text{discoDuro}} = C_{\text{discoDuro}} / C_o$. Esto de que tiene que ser un número entero es mentira, sale lo que tenga que salir. No hemos tenido que echar ni una cuenta.

- El tiempo medio de respuesta del disco duro es de 0,76s.

Verdadero, sabemos que el tiempo medio en la cola de espera del disco duro ($W_{\text{discoDuro}}$) = 0.75s. También sabemos que el tiempo medio de respuesta del disco duro es $R_{\text{discoDuro}} = W_{\text{discoDuro}} + S_{\text{discoDuro}}$

Nos falta saber $S_{\text{discoDuro}}$ por ejemplo podemos hacer lo siguiente. $S_{\text{discoDuro}} = B_{\text{discoDuro}} / C_{\text{discoDuro}}$. Por el enunciado sabemos que el número de trabajos completados por el disco duro es ($C_{\text{discoDuro}}$) 38 000. Necesitamos saber $B_{\text{discoDuro}}$.

Para hallar $B_{\text{discoDuro}}$ podemos utilizar la siguiente relación. $U_{\text{discoDuro}} = B_{\text{discoDuro}} / T$. Por el enunciado sabemos que $U_{\text{discoDuro}}$ es 0.38 (38%) y que el período de monitorización es 1000s.

$$B_{\text{discoDuro}} = 0.38 * 1000s = 380s$$

$$S_{\text{discoDuro}} = 380s / 38\,000 = 0.01s$$

$$R_{\text{discoDuro}} = 0.75 + 0.01 = 0.76s$$

~ Los parámetros del modelo de un servidor de comercio electrónico (red abierta) son los siguientes:

Dispositivo	tiempo medio de servicio (ms)	razón media de visita
CPU (1)	1	9
SSD (2)	0,5	10

Teniendo en cuenta que el servidor recibe una media de 0.15 peticiones por milisegundos, indique si las siguientes afirmaciones son verdaderas o falsas:

- La demanda de servicio media de la unidad de estado sólido es 5 tr/ms.

Falso, la demanda de servicio es un tiempo por lo que no se puede medir en tr/s. No ha habido que echar cuentas, hay que pararse a pensar.

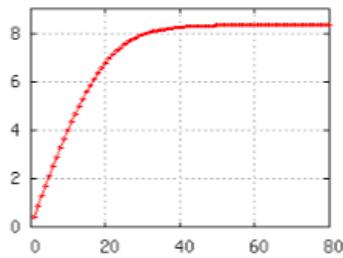
- La utilización media de la unidad de estado sólido es 0.75

Falso, para poder usar la expresión $U_{ssd} = \lambda_o * D_{ssd}$ hace falta demostrar primero que estamos en equilibrio de flujo. Es fácil ver que la CPU es el cuello de botella y que $X_o \text{ max} = 1/D_{CPU} = 0,11 \text{ tr/ms}$. Por tanto, $\lambda_o > X_o \text{ max}$ y NO SE CUMPLE EL EQUILIBRIO DE FLUJO. En este caso, U_{ssd} será la máxima que puede alcanzar la unidad en el seno de este servidor: $U_{ssd} \text{ max} = X_o \text{ max} * D_{ssd} = 0,56$. Pregunta criminal la vdd.

- En una red de colas cerrada interactiva se cumple que: $N_t = X_o \text{ max} * (R_o \text{ min} + Z)$

Falso, la parte de la derecha sirve para calcular el punto teórico de saturación y no el número total de usuarios en una red completa. Si te sabes las 500 fórmulas de cabeza imagino que fácil.

- La siguiente figura puede corresponder a la evolución de la productividad media de un servidor modelado mediante una red de colas cerrada frente al número total de usuarios en dicha red



Verdadero, hay un punto en el que el servidor se satura y su productividad no puede aumentar, por lo que se mantiene al máximo siempre.

~ Durante las últimas 24 horas, se ha monitorizado un servidor de base de datos no saturado con el fin de obtener un modelo del mismo basado en redes de colas. Como resultado de dicha monitorización, se han obtenido las siguientes medidas:

- 1.- La productividad media del disco A ha sido de 10 accesos de lectura/escritura atendidos por segundo y la del disco B el doble.
- 2.- Por cada consulta al servidor, se ha accedido, de media, 5 veces al disco A.
- 3.- La utilización de la CPU ha sido del 50%
- 4.- El servidor tarda 3s en atender, de media, cada consulta que se le hace.

Indique si las siguientes afirmaciones son verdaderas o falsas:

- La razón de visita del disco B es 10.

Verdadero, si la productividad del disco A es 10 y su razón de visita es 5, y la productividad del disco B es 20, entonces su razón de visita debe ser 10.

- Suponiendo que cada cliente envía una única consulta, hay una media de 6 clientes conectados al servidor durante el tiempo de monitorización.

Verdadero, realmente nos están preguntando si $N_o >= 6$. Como se está en equilibrio de flujo, se puede aplicar la Ley de Little $N_o = X_o * R_o$.

Primero hallamos $X_o = (10 \text{ tr/s}) / V_{DA} = 2\text{tr/s}$ y luego calculamos $N_o = X_o * R_o = 2\text{tr/s} * 3\text{s} = 6\text{tr} = 6 \text{ clientes}$.

- La demanda de servicio de la CPU es 250ms.

Verdadero, la demanda de servicio es $D_{CPU} = B_{CPU} / C_o = U_{CPU} / X_o = 0.5\text{s}/2 = 0.25\text{s} = 250 \text{ ms}$.

- En una red abierta en equilibrio de flujo se cumple que $R_o = R_1 + R_2 + \dots + R_k$

Falso, te tienes que saber que la ley general del tiempo de respuesta que es $R_o = V_1 * R_1 + V_2 * R_2 + \dots + V_k * R_k$

- En saturación, el cuello de botella está al máximo de su productividad

Verdadero, el cuello de botella no puede trabajar a más ritmo que ese, por tanto, está al máximo de su productividad.

- La tasa media de llegada de peticiones a un servidor no puede ser superior a $1/D_b$, siendo D_b la demanda media de servicio del cuello de botella.

Falso, lo que no puede ser superior a $1/D_b$ es la productividad.

~ Considere un servidor web que recibe una media de 0.3 peticiones por segundo y es modelado con los siguientes parámetros.

Dispositivo	S_i	V_i
CPU	0,20	15
DiscoA	0,04	6
DiscoB	0,06	8

Indique si las siguientes afirmaciones son verdaderas o falsas:

- Si se substituye el procesador por otro dos veces y media más rápido, es razonable suponer que su razón de visita sea menor.

Falso, los procesos que tienen que pasar por el procesador tardarán más o tardarán menos. Lo que cambia es el tiempo medio de servicio. Si en vez de sustituir el procesador, se colocara otro, sí que cambiaría la razón de visita.

- Si conseguimos repartir el contenido de cada uno de los dos discos para equilibrar la demanda de servicio entre ellos, las nuevas razones de visita de los discos serían: $V_{\text{discoA}} = 8.4$ y $V_{\text{discoB}} = 5.6$.

Verdadero, se tienen que conservar el número medio de visitas que recibe el subsistema de almacenamiento, $V_{\text{discoA}} + V_{\text{discoB}} = 8,4 + 5,6 = 14 = 6 + 8$.

Tema 6.

- En el pliego de condiciones se divide en pliego de cláusulas administrativas particulares y pliego de prescripciones técnicas.

Verdadero, mirar las diapositivas del tema 6.

- La memoria técnica que presenta cada licitador no podrá hacer referencia a una fabricación o una procedencia determinada con la finalidad de favorecer o descartar ciertas empresas o ciertos productos. Si no es posible, se acompañará la mención o equivalente.

Falso, es en el pliego de prescripciones técnicas en donde se aplica esa normativa y no en la memoria técnica.