

# Tema-4.pdf



**Anónimo**



**Ingeniería de Servidores**



**3º Grado en Ingeniería Informática**



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

# Tema 4

## Análisis comparativo del rendimiento

¿Qué servidor tiene mejor rendimiento?

### 4.1. REFERENCIACIÓN (BENCHMARKING)

#### CARACTERÍSTICAS DE UN BUEN ÍNDICE DE RENDIMIENTO DE UN SISTEMA INFORMÁTICO

¿Qué índice de rendimiento debemos usar? El que tenga mejores características. Estas son:

- Representatividad y fiabilidad: Tiene que representar el índice de rendimiento. (por ejemplo, de nada nos sirve la temperatura para el rendimiento). Si un sistema A siempre presenta un índice de rendimiento mejor que el sistema B, es porque siempre el rendimiento real de A es mejor que el de B. Es decir, tiene que ser verdad.
- Repetibilidad: Siempre que se mida el índice en las mismas condiciones, el valor de éste debe ser el mismo. Mismas condiciones = Mismos resultados (mismo índice).
- Consistencia: El índice se debe poder medir en cualquier sistema informático.

Además, es recomendable:

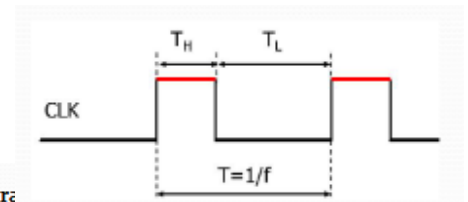
- Facilidad de medición: La medida debe ser fácil de tomar.
- Linealidad: Si el índice de rendimiento aumenta, el rendimiento real del sistema debe aumentar en la misma proporción.

#### TIEMPO DE EJECUCIÓN, FRECUENCIA DE RELOJ Y CPI

¿Pueden ser la frecuencia de reloj ( $f_{RELOJ}$ ) o el número medio de ciclos por instrucción (CPI) buenos índices de rendimiento?

$$T_{EJEC} = NI \times CPI \times T_{RELOJ} = \frac{NI \times CPI}{f_{RELOJ}}$$

- $T_{EJEC}$  = Tiempo de ejecución del progr.
- NI = Número de instrucciones del programa.



No lo son. Es posible encontrar ejemplos de sistemas con  $f_{RELOJ}$  (o CPI) peores que otros pero con mejores prestaciones.

¿Y si usamos directamente el tiempo de ejecución ( $T_{EJEC}$ ) de un determinado programa?

Hay ciertos problemas también:

- ✗ Consistencia: Si está escrito en un lenguaje ensamblador, no podríamos usar ese lenguaje en otras máquinas (pueden tener otro repertorio). Solución: El programa debería estar escrito en un lenguaje de alto nivel.
- ✗ Repetibilidad: El SO puede tener otros procesos al volver a ejecutar el programa. Solución: El programa debería ejecutarse en un entorno muy controlado, asegurándonos de que esté libre.
- ✗ Representatividad y fiabilidad: ¿Me asegura que el A me ejecuta todo lo demás mejor que el B? Dependería del programa a ejecutar

$$MIPS = \frac{NI}{T_{EJEC} \times 10^6} = \frac{f_{RELOJ}}{CPI \times 10^6}$$

### MIPS (million of instructions per second)

En principio, parece una medida prometedora ya que representa cómo de rápido ejecuta las instrucciones un microprocesador.

Inconvenientes:

- ✗ Representabilidad y fiabilidad: Todos los procesadores no tienen el mismo repertorio de instrucciones. Depende del juego de instrucciones (p.e RISC (instrucciones sencillas, más cantidad de instrucciones) vs CISC (instrucciones complejas, menos cantidad de instrucciones)
- ✗ Repetibilidad: Los MIPS medidos varían incluso entre diferentes programas con el mismo computador.

### MFLOPS (million of floating-point operations per second)

En vez de medir instrucciones, medimos operaciones (en este caso operaciones de coma flotante).

$$MFLOPS = \frac{\text{Operaciones de coma flotante realizadas}}{T_{EJEC} \times 10^6}$$

Inconvenientes:

- ✗ Representatividad y fiabilidad: No todas las operaciones de coma flotante tienen la misma complejidad (p.e suma vs arctg).
  - Posible solución: MFLOPS normalizados: Cada operación se multiplica por un peso que es proporcional a su complejidad. *Ejemplo de asignación de pesos:*
    - ADD, SUB, COMPARE, MULT → 1 operación normalizada
    - DIVIDE, SQRT → 4 operaciones normalizadas
    - EXP, SIN, ATAN → 8 operaciones normalizadas
- ✗ Consistencia: El formato de los números en coma flotante puede variar de una arquitectura a otra y, por tanto, los resultados de las operaciones podrían tener diferente exactitud. No todos los procesadores utilizan el mismo número de bits ni el mismo formato para las operaciones en coma flotante. Además, ¿y si no necesito las operaciones en coma flotante en mi servidor?

Conclusión final: Tampoco nos vale y no hay más candidatos. Nos quedaremos con la idea menos mala. Nos contentaremos con el tiempo de ejecución ( $T_{EJEC}$ ) de un determinado programa o conjunto de programas. Utilizo el Tiempo de ejecución sabiendo que esa medida solo servirá para ese programa → El índice de rendimiento va a depender de la carga con la que se haga la comparación.

¿Qué carga utilizará? La carga real

### CARGA REAL

Me servirá para medir las prestaciones en mi máquina → DE FORMA LOCAL

No es la mejor idea, es difícil de utilizar en la evaluación de sistemas

- ✗ Varía a lo largo del tiempo: ¿Uso la carga de hoy, la de ayer...?
- ✗ Resulta complicado reproducirla: Si la uso con un servidor muy lento es posible que llegue una solicitud basada en la respuesta que todavía no ha llegado. Si quiero la carga de 1 semana, tengo que esperar a tenerla durante una semana → no es muy razonable.
- ✗ Interacciona con el sistema informático: Si un servidor contesta muy rápido, es posible que reciba las solicitudes antes o incluso más solicitudes que si fuera muy lento.

Es más conveniente utilizar un modelo de la carga real como carga de prueba (test workload) para hacer comparaciones.



## REPRESENTABILIDAD DEL MODELO DE CARGA

Los modelos de carga son representaciones aproximadas de la carga que recibe un sistema informático. Se trata de tener una serie de peticiones que se puedan reproducir en un tiempo razonable. El modelo de la carga:

- Debe ser lo más representativo posible de la carga real.
- Debe ser lo más simple/compacto que sea posible (tiempos de medición y espacio en memoria razonables)



## PRINCIPALES ESTRATEGIAS PARA OBTENER MODELOS DE CARGA

Hay dos formas:

- Ajustar un modelo paramétrico “personalizado” a partir de la monitorización del sistema ante la carga real (caracterización de la carga). Es decir, hacernos un modelo personalizado de nuestra carga.

Solo en el contexto:

- *Mido en mi servidor → Cambio algo en mi servidor → Mido en mi servidor.*
- *No puedo interactuar con gente de otros servidores para comparar.*

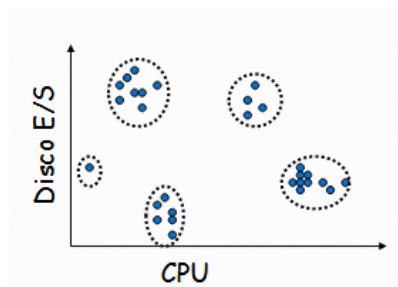
*Vamos a suponer que tengo 1 CPU y un disco duro. Durante 1 semana estoy viendo la carga (las peticiones que me llegan). A lo largo de este tiempo, veo que me llegan muchas peticiones de x, pocas de y... Realizo una simulación. Hago mi propia carga.*

- Usar programas de prueba que usen un modelo genérico de carga lo más similar posible al que se quiere reproducir (referenciación o benchmarking). Es decir, nos bajamos un benchmark que sea parecido a lo que nos interese.

## CARACTERIZACIÓN DE LA CARGA

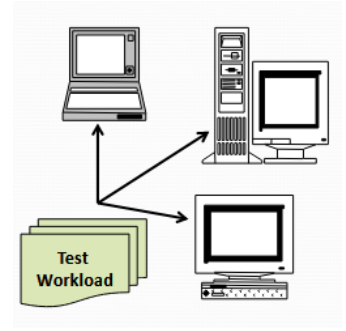
La forma más fácil para obtener un modelo de la carga a la que está sometido un servidor durante un determinado periodo de tiempo consiste en:

- Identificar los recursos que más demande la carga (CPU, memoria, discos, red, etc.)
- Elegir los parámetros característicos de dichos recursos (utilización de CPU, lecturas/escrituras que hay que hacer en cada disco, lecturas/escrituras a memoria, número de accesos a la red, etc.)
- Medir el valor de dichos parámetros usando monitores de actividad (muestreo).
- Analizar los datos: medias, histogramas, agrupamiento o clustering, etc.
- Generar el modelo de carga seleccionando representantes de la carga (=solicitudes al servidor) junto con información estadística sobre su distribución temporal



## REFERENCIACIÓN (BENCHMARKING)

Consiste en utilizar un programa o un conjunto de programas (benchmark programs) con el fin de comparar alguna característica del rendimiento entre equipos informáticos. Se trata de ir a webs donde hay determinados tipos de programas de prueba. Con estos programas al ejecutarse, podemos testear el rendimiento. Hay dos características principales que definen a un buen benchmark:



- ★ La carga de prueba (test workload) específica con la que estresa el sistema evaluado.
- ★ El conjunto de reglas que se deben seguir para la correcta ejecución, obtención y validación de los resultados.

## VENTAJAS DE USAR BENCHMARKING

- ✓ No voy a tener que hacer el proceso tedioso de la caracterización de la carga real.
  - Voy a buscar un benchmark parecido a lo que necesito.
- ✓ Existen muchos benchmarks diferentes para distintos tipos de servidores y cargas. Hay una alta probabilidad de encontrar uno que reproduzca unas condiciones parecidas a las que experimenta nuestro servidor.
- ✓ Las comparaciones entre el rendimiento de varios servidores son justas, ya que todas las ejecuciones se realizan de forma idéntica siguiendo las reglas del benchmark.
- ✓ Muchos benchmarks permiten ajustar la carga de tal forma que podemos medir la escalabilidad del servidor. Muchos permiten cambiar parámetros para escalarlo a nuestro servidor. *En lugar de 100.000 clientes, 10.000, en lugar de una base de datos de 3TB, de 500GB, etc...*
- ✓ Al poder conocer tanto el rendimiento para un determinado benchmark que obtienen diferentes servidores como cómo están diseñados y configurados dichos servidores, obtenemos una información muy valiosa sobre cómo diseñar y/o configurar nuestros propios servidores.
  - Nadie sabe lo que va a necesitar el servidor en cuanto a recursos. Veo un benchmark y miro las configuraciones para él y así puedo basarme en este.
    - La mayoría de los servidores ahora se montan en la nube → Puedo decir: ponme dos cores más, dame mas RAM...



## TIPOS DE PROGRAMAS DE BENCHMARK: SEGÚN LA ESTRATEGIA DE MEDIDA

- Programas que miden el tiempo necesario para ejecutar una cantidad pre-establecida de tareas. (La mayoría de benchmarks).
- Programas que miden la cantidad de tareas ejecutadas para un tiempo de cómputo pre-establecido. Es decir, te dan un Tiempo de ejecución y miran a ver hasta donde puede llegar/ejecutar.
  - SLALOM: Mide la exactitud de la solución de un determinado problema que se puede alcanzar en 1 minuto de ejecución.
- Programas que permiten modificar sus parámetros para adaptarlos a cada sistema. Te permiten adoptarlo a un programa en cuestión → modificar la carga (en vez de 3000 clientes, que haya 500...)
  - TPC-C: Calcula cuántas consultas por segundo se realizan, de media, a un servidor de base de datos permitiendo aumentar tanto el nº de usuarios como el tamaño de la base de datos. Exige un tiempo mínimo de respuesta para un tanto por ciento de usuarios.

## TIPOS DE PROGRAMAS DE BENCHMARK: SEGÚN LA GENERALIDAD DEL TEST

- **Microbenchmarks** o benchmarks para componentes: estresan componentes o agrupaciones de componentes concretos del sistema: procesador, caché, memoria, discos, red, procesador + caché, procesador + compilador + memoria virtual, etc.
  - Es muy difícil testar el rendimiento de un componente de forma aislada. *Si mido por ejemplo el disco duro, puede que esté leyendo de caché.*  
ES COMPLICADO AISLAR COMPONENTES
- **Macrobenchmarks** o benchmarks de sistema completo o de aplicación real: la carga intenta imitar situaciones reales (normalmente servidores con muchos clientes) típicas de algún área. Realizan una simulación completa. Son muy útiles. *P.ej. e-comercio, servidores web, servidores de ficheros, servidores de bases de datos, sistemas de ayuda a la decisión, paquetes ofimáticos + correo electrónico + navegación, etc*

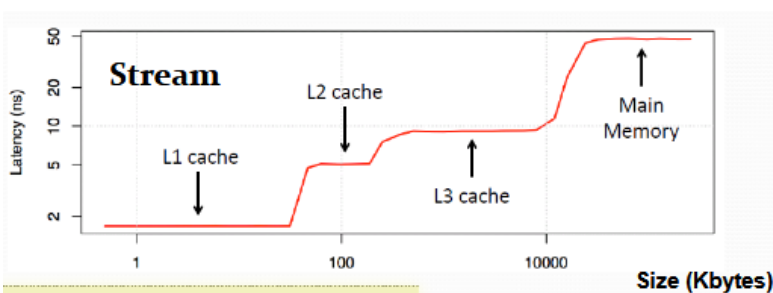
El objetivo del benchmark es hacer la medición del rendimiento. NO es un test de diagnóstico (no comprueba si funciona la RAM, por ejemplo). No es lo mismo un programa que testea si un componente funciona, que un benchmark, el cual mide el rendimiento.

### Ejemplos de microbenchmarks:

Se iniciaron para supercomputadores con operaciones en coma flotante que calculaban ecuaciones diferenciales (con senos, cosenos, etc...)

- Whetstone (1976): Mide el rendimiento de las operaciones en coma flotante por medio de pequeñas aplicaciones científicas que usan sumas, multiplicaciones y funciones trigonométricas.
- Linpack (1983): Mide el rendimiento de las operaciones en coma flotante a través de un algoritmo para resolver un sistema denso de ecuaciones lineales. El benchmark incorpora una rutina para comprobar que la solución a la que se llega es la correcta con un grado de exactitud prefijado. Se utiliza para confeccionar la lista de los 500 mejores supercomputadores del mundo.
- Dhrystone (1984): Mide el rendimiento de operaciones con enteros, esencialmente por medio de operaciones de copia y comparación de cadenas de caracteres.
- Stream: para medir el ancho de banda de la memoria
- IOzone: rendimiento del sistema de ficheros (p.ej. lecturas y escrituras a/desde el disco duro). Igualmente HD Tune, Iometer, fio (flexible I/O tester, Linux) o el comando 'hdparm -tT' (Linux).
- Netperf: rendimiento TCP y UDP (Linux y Windows). Se usa en combinación con otro programa (netserver) que debe estar instalado en el servidor. <http://www.netperf.org/netperf/>. También pchar (=traceroute que calcula el ancho de banda por cada salto).
- También existen aplicaciones que incorporan varios paquetes de microbenchmarks para poder realizar diversos tests de forma cómoda:
  - Phoronix Test Suite (Open Source, <https://www.phoronix-test-suite.com/>).
  - AIDA64 (Windows, <http://www.aida64.com>).
  - Sandra (Windows, <http://www.sisoftware.net>).

64



```
$ fio --name=seqwrite --rw=write --bs=128k --size=122374m
[+]
seqwrite: (groupid=0, jobs=1): err=0: pid=22321
write: io=122374MB, bw=840951KB/s, iops=6569, runt=149011msec
clat (usec): min=41, max=133186, avg=148.26, stdev=1287.17
lat (usec): min=44, max=133188, avg=151.11, stdev=1287.21
bw (KB/s): min=10746, max=1983488, per=100.18%, avg=842503.94,
stdev=262774.35
cpu: usr=2.67%, sys=43.46%, ctx=14284, majf=1, minf=24
IO depths: 1=100.0%, 2=0.0%, 4=0.0%, 8=0.0%, 16=0.0%, 32=0.0%, >=64=0.0%
submit: 0=0.0%, 4=100.0%, 8=0.0%, 16=0.0%, 32=0.0%, 64=0.0%, >=64=0.0%
complete: 0=0.0%, 4=100.0%, 8=0.0%, 16=0.0%, 32=0.0%, 64=0.0%, >=64=0.0%
issued r/w/d: total=0/978992/0, short=0/0/0
lat (usec): 50=0.02%, 100=98.30%, 250=1.06%, 500=0.01%, 750=0.01%
lat (msec): 1000=0.01%
lat (msec): 2=0.01%, 4=0.01%, 10=0.25%, 20=0.29%, 50=0.06%
lat (msec): 100=0.01%, 250=0.01%
```



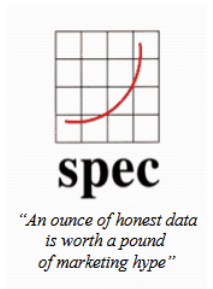
Cuando hablamos de memoria, se habla de latencia en vez de Tiempo de Respuesta. Pero en realidad es lo mismo.

## SPEC

Es una corporación sin ánimo de lucro cuyo propósito es establecer, mantener y respaldar la estandarización de benchmarks y herramientas para evaluar el rendimiento y la eficiencia energética de los equipos informáticos.

Proporciona un caso más específico de benchmark.

Se ha usado de forma intensiva desde hace años para testar el rendimiento de CPU.



## EL PAQUETE DE MICROBENCHMARKS SPEC CPU 2017

Compuesto por cuatro conjuntos de benchmarks distintos:

- SPECSpeed 2017 Integer (rendimiento en aritmética entera).
  - SPECSpeed 2017 Floating Point (rendimiento en coma flotante)
  - SPECrate 2017 Integer (rendimiento en aritmética entera)
  - SPECrate 2017 Floating Point (rendimiento en coma flotante)
- Speed: Cuánto tarda en ejecutarse un programa (Tiempo de respuesta)
  - Rate: Cuántos programas puedo ejecutar por unidad de tiempo (productividad)

Por tanto, cuando hablamos de rendimiento son importantes el Tiempo de Respuesta y la Productividad.

En esta asignatura veremos los dos speed con detenimiento.

Componentes que se evalúan:

- ★ Procesador (enteros o coma flotante según el caso)
- ★ Sistema de memoria
- ★ Compilador: Por el tema de consistencias (C, Fortran y C++).

Existen ciertas reglas estrictas para validar los resultados.

SPEC CPU2017 se distribuye como una imagen ISO que contiene:

- Código fuente de todos los programas de benchmark
- Data sets que necesitan algunos benchmarks para su ejecución (conjunto de datos con los que opera)
- Herramientas varias para compilación, ejecución, obtención de resultados, validación y generación de informes.
- Documentación, incluyendo reglas de ejecución y de generación de informes.

El tiempo de ejecución depende del índice a obtener, la máquina en la que se ejecuta y cuántas copias o subprocesos se eligen.

Metric	Config Tested	Individual benchmarks	Full Run (Reportable)
SPECrate2017_int_base	1 copy	6 to 10 minutes	2.5 hours
SPECrate2017_fp_base	1 copy	5 to 36 minutes	4.8 hours
SPECSpeed2017_int_base	4 threads	6 to 15 minutes	3.1 hours
SPECSpeed2017_fp_base	16 threads	6 to 75 minutes	4.7 hours

Criterios generales para qué programas utilizar:

- Ejemplo: SPECspeed 2017 Integer: 10 programas (la mayoría en C y C++)

- Ejemplo: *SPECspeed 2017 Floating Point: 10 programas (la mayoría en Fortran y C)*

- También llamados, de forma genérica, **índices SPEC**. Para cada SPECSpeed hay dos:

- Significado de “base” y “peak”:

- ### ¿Cómo se calcula cada uno de estos índices?

Ejemplo: Si llamamos  $T_i$  al tiempo que tarda la máquina a evaluar en ejecutar el programa de benchmark  $i$ -ésimo y  $T_i^{\text{REF}}$  lo que tardaría la máquina de referencia para ese programa (recordemos que hay 10 programas en el benchmark):

$t1$ : lo que tarda en ejecutar el programa 1

t2: " " " " " " 2

*t1REF: lo que tarda en ejecutar P1 en la máquina de referencia (siempre se va a coger el base de la máquina de referencia)*


El resultado del índice es  $>1$  normalmente porque la máquina de referencia tardará más que yo normalmente en ejecutar.

El valor PICO nunca va a ser menor que el base  $\rightarrow$  El índice SPEC del pico siempre va a ser mayor o igual. Sería absurdo que fuese al contrario.

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.



## RESULTADOS DE SPECSpeed 2017 Integer



### All SPEC CPU2017 Integer Speed Results Published by SPEC

These results have been submitted to SPEC; see [the disclaimer](#) before studying any results.

[Search published CPU2017 results](#)

Last update: 2017-10-19T11:49

#### CPU2017 Integer Speed (7):

[\(Search in CPU2017 Integer Speed results\)](#)

Test Sponsor	System Name	Parallel	Base Threads	Processor			Results	
				Enabled Cores	Enabled Chips	Threads/Core	Base	Peak
HPE	Integrity Superdome X (384 core, 2.20 GHz, Intel Xeon E7-8890 v4) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	No	384	384	16	2	5.31	5.86
HPE	ProLiant DL580 Gen9 (2.20 GHz, Intel Xeon E7-8890 v4) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	No	96	96	4	1	5.35	5.95
HPE	ProLiant ML350 Gen9 (2.20 GHz, Intel Xeon E5-2699 v4) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	No	44	44	2	1	5.80	6.43
HPE	ProLiant DL380 Gen10 (2.10 GHz, Intel Xeon Platinum 8170) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	Yes	52	52	2	1	8.96	Not Run
HPE	ProLiant DL380 Gen10 (2.10 GHz, Intel Xeon Platinum 8176) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	Yes	56	56	2	1	9.16	Not Run
Huawei	Huawei 2288H V5 (Intel Xeon Platinum 8180) <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	Yes	56	56	2	1	9.46	9.79
Oracle Corporation	Sun Fire V490 <a href="#">HTML</a>   <a href="#">CSV</a>   <a href="#">Text</a>   <a href="#">PDF</a>   <a href="#">PS</a>   <a href="#">Config</a>	Yes	1	8	4	1	1.00	Not Run

Hardware		Software	
CPU Name:	Intel Xeon E7-8890 v4	OS:	SUSE Linux Enterprise Server 12 (x86_64) SP1
Max MHz:	3400		3.12.53-60.30-default
Nominal:	2200	Compiler:	C/C++: Version 17.0.0.098 of Intel C/C++ Compiler for Linux; Fortran: Version 17.0.0.098 of Intel Fortran Compiler for Linux
Enabled:	384 cores, 16 chips, 2 threads/core	Parallel:	No
Orderable:	2 to 16 chips	Firmware:	ITP Tunnel: 008.004.004 SPW: 043.025.000 08/16/2016
Cache L1:	32 KB L1 + 32 KB D on chip per core	File System:	xfs
L2:	256 KB I+D on chip per core	System State:	Run level 5 (multi-user, w/GUI)
L3:	60 MB I+D on chip per chip	Base Pointers:	64-bit
Other:	None	Peak Pointers:	32/64-bit
Memory:	4 TB (128 x 32 GB 2Rx4 PC4-2400T-L, running at 1600 MHz)	Other:	Microquill SmartHeap V10.2
Storage:	8 x C8S59A, 900 GB 10 K RPM SAS		
Other:	None		

Results Table														
Benchmark	Base							Peak						
	Threads	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio	Threads	Seconds	Ratio	Seconds	Ratio	Seconds	Ratio
600.perlbenc h_s	384	365	4.86	358	4.96	357	4.98	384	298	5.95	295	6.02	295	6.01
602.gcc_s	384	553	7.20	546	7.29	546	7.29	384	540	7.37	538	7.45	534	7.45
605.mcf_s	384	866	5.45	866	5.45	898	5.26	384	708	6.67	700	6.75	699	6.75
620.omnetpp_s	384	276	5.90	271	6.03	289	5.65	384	251	6.50	247	6.61	246	6.64
623.xalanbm k_s	384	189	7.50	188	7.52	187	7.57	384	179	7.91	179	7.93	180	7.87
625.x264_s	384	233	6.24	232	6.25	233	6.23	384	271	6.51	272	6.49	270	6.52
631.deepsjeng_s	384	407	3.52	408	3.52	407	3.52	384	343	4.18	343	4.18	343	4.18
641.look_s	384	369	3.61	369	3.61	369	3.61	384	343	4.18	343	3.98	340	3.85

Información Hardware y Software.

Tabla de resultados para los 10 programas

Con qué parámetros/flags se ha compilado cada fichero:

Base Optimization Flags	
<b>C benchmarks:</b>	<code>-qopt-prefetch -qopt-mem-layout-trans=3 -DSPEC_SUPPRESS_OPENMP</code>
<b>C++ benchmarks:</b>	<code>-Wl,-z,muldefs -qopt-prefetch -qopt-mem-layout-trans=3 -DSPEC_SUPPRESS_OPENMP -L/sh10.2 -lsmarheap64</code>
Peak Optimization Flags	
<b>C benchmarks:</b>	<code>600.perlbench_s: -prof-gen(pass 1) -prof-use(pass 2) -O2 -xCORE-AVX2 -auto-p32 -ipo -qopt-prefetch -O3 -no-prec-div -qopt-mem-layout-trans=3 -DSPEC_SUPPRESS_OPENMP</code>
<b>C++ benchmarks:</b>	<code>602.gcc_s: Same as 600.perlbench_s</code>

EN EL EXAMEN PROBABLMENTE TENDREMOS QUE CALCULAR UN ÍNDICE SPEC

### EJEMPLO DE CÁLCULO DE SPECSPEED 2017 Integer base

Benchmark	$t^{REF}$ (s)	Exp1 (s)	Exp2 (s)	Exp3 (s)	$t^{base}$ (s)	$t^{REF} / t^{base}$
<a href="#">600.perlbench s</a>	1774	365	<a href="#">358</a>	357	358	4,96
<a href="#">602.gcc s</a>	3981	553	<a href="#">546</a>	546	546	7,29
<a href="#">605.mcf s</a>	4721	<a href="#">866</a>	866	898	866	5,45
<a href="#">620.omnetpp s</a>	1630	<a href="#">276</a>	271	289	276	5,91
<a href="#">623.xalancbmk s</a>	1417	189	<a href="#">188</a>	187	188	7,54
<a href="#">625.x264 s</a>	1764	<a href="#">283</a>	282	283	283	6,23
<a href="#">631.deepsjeng s</a>	1432	<a href="#">407</a>	408	407	407	3,52
<a href="#">641.leela s</a>	1706	469	469	<a href="#">469</a>	469	3,64
<a href="#">648.exchange2 s</a>	2939	<a href="#">329</a>	329	329	329	8,93
<a href="#">657.xz s</a>	6182	2165	2161	<a href="#">2164</a>	2164	2,86

$$\text{SPECspeed@2017 Integer}_{base} = \sqrt[10]{\frac{t^{REF}_1}{t^{base}} \times \frac{t^{REF}_2}{t^{base}} \times \dots \times \frac{t^{REF}_{10}}{t^{base}}} = \sqrt[10]{4,96 \times 7,29 \times 5,45 \times \dots} = 5,31$$

SPEC es un caso de microbenchmark → Testar el rendimiento de un componente / conjunto de componentes asociado a un conjunto de reglas para que las medidas y comparaciones sean justas (xq se usa para los vendedores)

### BENCHMARKS DE SISTEMA COMPLETO: TPC

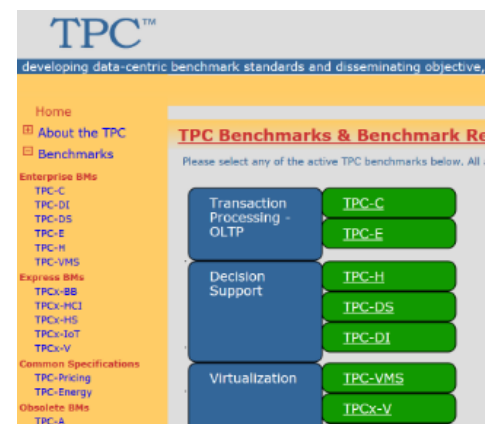
TPC (Transactions Processing Performance Council): Organización sin ánimo de lucro especializada en benchmarks relacionados con comercio electrónico y con bases de datos

Se trata de simular una aplicación real.

Situaciones particulares:

- OLTP: Procesamiento de ventas online
- Decision Support: Ayuda a la decisión

No es lo mismo un benchmark sobre una máquina física que en un entorno virtualizado → TPC ofrece esta opción



Principales benchmarks (todos son escalables → te pide el factor de escala cuando buscas en la web.)

- TPC-C: Tipo OLTP (on-line transaction processing). Simula una gran compañía que vende 100.000 productos y que tiene varios almacenes (configurable), cada uno a cargo de 10 zonas, con 3000 clientes/zona. Las peticiones involucran acceso a las bases de datos tanto locales como distribuidas (a veces el producto no está en el almacén más cercano). Distribuidos. Mucho más complejo. BD y tenemos un montón de tiendas distribuidas. Muchos clientes y peticiones. Se simula esto para ver el rendimiento.
- TPC-E: Tipo OLTP. Simula una correduría de bolsa en donde hay una única base de datos central. Monolítico. Los clientes entran y acceden a la BD centralizada
- TPC-H, TPC-DS: Tipo DS (decision support). Se deben ejecutar consultas altamente complejas a una gran base de datos y analizar enormes volúmenes de datos (minería de

datos, big-data). Gran BD → Consultas muy complejas. Acceso a una gran parte de la BD.  
Gran uso de la CPU para procesar toda esa información

Métricas: cuantas transacciones (compras/ventas) se procesan por unidad de tiempo (tps/tpm/tph) superando unos ciertos requisitos de tiempos de respuesta (ej. el 90% deben tener un tiempo de respuesta inferior a 1s). También: coste por transacción procesada (incluido mantenimiento) y consumo de potencia por transacción procesada. (En SPEC la métrica era la media geométrica).

- Problema: Muchos clientes a los que tengo que contestar → No contesto a muchos
- Solución: Requisitos de tiempo de respuesta

En la web puedo ver la información de la máquina para poder saber cuando quiera montar algo parecido.

Sponsor	System	Scale Factor	Performance (QphH)	Price/QphH	System Availability	Date Submitted	DB Software Name
Hewlett Packard Enterprise	HPE ProLiant DL380 Gen9	1,000	717,101	0.61 USD	10/19/2017	4/17/2017	Microsoft SQL Server 2017 Enterprise Edition
Hewlett Packard Enterprise	HPE ProLiant DL380 Gen9	1,000	543,102	0.69 USD	7/31/2016	3/9/2016	Microsoft SQL Server 2016 Enterprise Edition
Lenovo	Lenovo System x3850 X6	3,000	969,504	0.72 USD	7/31/2016	3/9/2016	Microsoft SQL Server 2016 Enterprise Edition
Hewlett Packard Enterprise	HPE ProLiant DL380 Gen9	1,000	678,492	0.64 USD	7/31/2016	7/24/2016	Microsoft SQL Server 2016 Enterprise Edition

#### TPC-H Result Highlights

As of 25-Oct-2017 at 3:54 PM [GMT]

**CISCO** Cisco UCS C460 M4 Server

Reference URL: <http://www.tpc.org/3322>

##### Benchmark Stats

Result ID:	116051401
Status:	Accepted Result
Report Date:	02/14/16
TPC-H Row:	2.17.1

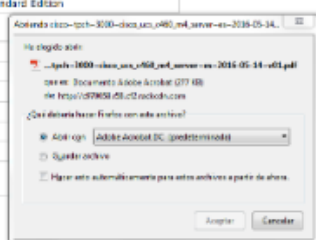
##### System Information



Total System Cost:	\$34,322 USD
Performance:	1,071,618 QphH@3000GB
Price/Performance:	.60 USD per QphH@3000GB
TPC-Energy Metric:	Not reported
Availability Date:	06/01/16
Database Manager:	Microsoft SQL Server 2016 Enterprise Edition
Operating System:	Microsoft Windows Server 2012 R2 Standard Edition

##### Server Specific Information

CPU Types:	Intel Xeon E7-8850 v3 2.5GHz
Total # of Processors:	4
Total # of Cores:	72
Total # of Threads:	144
Clock:	N/A
Load Time (hours):	1.94
Total Storage/Database Size Ratio:	2.99

- Executive Summary
- Full Disclosure Report
- Supporting Files-1



		Cisco UCS C460 M4 Server		TP
Total System Cost		Composite Query per Hour Metric		
\$634,322 USD		1,071,018.2 QphH@3000GB		
Database Size	Database Manager	Operating System		Other Software
3000GB	Microsoft SQL Server 2016 Enterprise Edition	Windows 2012 R2 Standard Edition		
				
Database Load Time = 1h 56m 26s		Storage Redundancy L		
Load Includes Backup: Y		Base Tables and Auxiliary Data Structures		
Total Data Storage / Database Size 2.99		DBMS Temporary Space		
Percentage Memory / Database Size 102.4%		OS and DBMS Software		
System Configuration:		Cisco UCS C460 M4 Server		
Processors/Cores/Threads/Model:		4/72/144 Intel Xeon E7-8890 v3 Processor (2.5 GHz)		
Memory:		3 TB		
Storage:		8 X 400GB 2.5 inch Ent Performance 12G SAS S5		
Table Storage:		4 X UCS Rack PCIe Storage 1600 GB SanDisk S5		
		9.38 TB		

Description	Unit Price	Qty	Extended Price
<b>Server Hardware</b>			
UCS C460 M4 base chassis w/o CPU/DIMM/HDD	16,500.00	1	\$16,500.00
3YR SNTC 24X7X40S UCS C460 M4 Server	3,487.00	1	
2.5GHz E7-8890 v3/165W/18C/45M Cache	21,000.00	4	\$84,000.00
32GB DDR4-2133-MHz RDIMM/PC4-17000/dual rank/x4/1.2v	1,100.00	96	\$105,600.00
UCS C460 M4 DDR4 Memory Riser with 12 DIMM slots	800.00	8	\$6,400.00
Riser card with 5 PCIe slots	500.00	2	\$1,000.00
400GB 2.5 inch Ent Performance 12G SAS SSD (10X endurance)	5,267.00	8	\$42,136.00
1400W V2 AC Power Supply (200 - 240V) 2U & 4U C Series	800.00	4	\$3,200.00

## BENCHMARKS DE SISTEMA COMPLETO: SPEC

SPEC también hace macrobenchmarks.

File Server: SFS2014: Tiempos de respuesta y productividades de servidores de ficheros.

High Performance Computing, OpenMP, MPI, OpenCL:

- SPEC MPI2007: Message Passing Interface (MPI).
- SPEC OMP2012: Open MultiProcessing (OpenMP).
- SPEC ACCEL: OpenCL y OpenACC

JAVA Cliente/Servidor:

- SPECjEnterprise2010: Java Enterprise Edition (JEE).
- SPECjms2007: Java Message Service (JMS).
- SPECjvm2008: Java Runtime Environment (JRE).

Cloud: SPEC Cloud IaaS 2018 (Servicios en la nube)

Virtualization: SPECvirt\_sc2013 (Virtualización en Centros de Procesamiento de Datos).

Consumo de potencia: SPECpower\_ssj2008 (Rendimiento de un servidor ejecutando aplicaciones JAVA frente al consumo de potencia).

## SYSMARK25

No cae. Es para comparar PC con SO Windows.

## 4.2. ANÁLISIS DE LOS RESULTADOS DE UN TEST DE RENDIMIENTO

### ¿CÓMO EXPRESAR EL RESULTADO FINAL TRAS LA EJECUCIÓN DE UN TEST DE RENDIMIENTO?

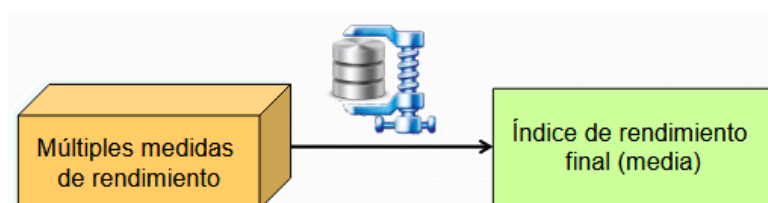
Muchos tests de rendimiento se basan en la ejecución de diferentes programas y, por tanto, producen diferentes medidas de rendimiento.

Sin embargo, estos tests suelen resumir todas estas medidas en un único valor: el índice de rendimiento de dicho test.

SPEC INT (por ejemplo) tenía 10 programas para comparar el rendimiento. Tenemos 10 valores. Preferimos uno solo por simplicidad.

¿Cómo concentrar todos los índices en uno solo?

- Método habitual de síntesis: uso de algún tipo de media.



## LA MEDIA ARITMÉTICA

Dado un conjunto de medidas,  $t_1, \dots, t_n$ , definimos su media aritmética:

$$\bar{t} = \frac{1}{n} \sum_{k=1}^n t_k$$

**Problema:** Puede haber programas cuyo Tiempo de Ejecución ( $T_E$ ) sea mayor que otros.

- Desproporción
- Premiaría/tendría en cuenta de igual forma a todos los programas. No es lo mismo bajar 500 a 2000 que a 6000.

No es la mejor forma de obtener un valor medio. No pesa igual un valor que otro.

Surge una alternativa:

## MEDIA ARITMÉTICA PONDERADA:

Si no todas las medidas tienen la misma importancia, se puede asociar a cada medida  $t_k$  un peso  $w_k$ .

Obteniéndose la media aritmética ponderada:

Los pesos están normalizados de forma que la suma de todos valga 1

$$\bar{t}_W = \sum_{k=1}^n w_k \times t_k \quad \text{con} \quad \sum_{k=1}^n w_k = 1$$

**Problema:** ¿Qué peso pongo? Es una decisión difícil. ¿Qué importancia le doy a cada medida?

**Alternativa:** Peso inversamente proporcional a los  $T_E$

Pero... ¿con qué valores calculará esos pesos? Tengo que tener una máquina de referencia.

$$w_k \equiv \frac{C}{t_k^{REF}}$$



$$C = \frac{1}{\sum_{k=1}^n 1/t_k^{REF}}$$

- **Problema:** Pesos dependen de la máquina de referencia.
- **Alternativa:**

## MEDIA GEOMÉTRICA

Media geométrica de las ganancias de velocidad respecto a una máquina de referencia.

Dado un conjunto de  $n$  medidas,  $S_1, \dots, S_n$ , definimos su media geométrica:

$$\bar{S}_g = \sqrt[n]{\prod_{k=1}^n S_k} = \left( \prod_{k=1}^n S_k \right)^{1/n}$$

Propiedad: cuando las medidas son ganancias en velocidad (speedups) con respecto a una máquina de referencia, este índice mantiene el mismo orden en las comparaciones independientemente de la máquina de referencia elegida. Usado en los benchmarks de SPEC y SYSMARK

Ventajas:

- ✓ Si la máquina de referencia es la misma, el numerador es el mismo.

$$SPEC(M) = \sqrt[n]{\frac{t_1^{REF}}{t_1^M} \times \frac{t_2^{REF}}{t_2^M} \times \dots \times \frac{t_n^{REF}}{t_n^M}} = \frac{\sqrt[n]{t_1^{REF} \times t_2^{REF} \times \dots \times t_n^{REF}}}{\sqrt[n]{t_1^M \times t_2^M \times \dots \times t_n^M}}$$

- ✓ Una máquina será mejor que otra cuando el valor del denominador sea menor que otra. No va a depender de la máquina de referencia porque el numerador es constante.

El hecho de que una máquina sea mejor que otra NO depende de la máquina de referencia

$$SPEC(M1) > SPEC(M2) \Leftrightarrow \sqrt[n]{t_1^{M1} \times t_2^{M1} \times \dots \times t_n^{M1}} < \sqrt[n]{t_1^{M2} \times t_2^{M2} \times \dots \times t_n^{M2}}$$



Programa	$t_{REF}$ (s)	$t^A$ (s)	$t^B$ (s)	$t^C$ (s)	$t^D$ (s)
1	1400	141	170	136	134
2	1400	154	166	215	25
3	1100	96,8	94,2	146	201
4	1800	271	283	428	523
5	1000	83,8	90,1	77,4	81,2
6	1200	179	189	199	245
7	1300	120	131	87,7	75,5
8	300	151	158	138	192
9	1100	93,5	122	88	118
10	1900	133	173	118	142
11	1500	173	170	179	240
12	3000	243	100	100	150
Suma	17000	1839,1	1846,3	1912,1	2126,7

## EJEMPLO DE COMPARACIÓN CON TIEMPOS

La máquina más rápida es "A" ya que es la que tarda menos en ejecutar, uno tras otro, todos los programas del benchmark (1839,1 segundos).

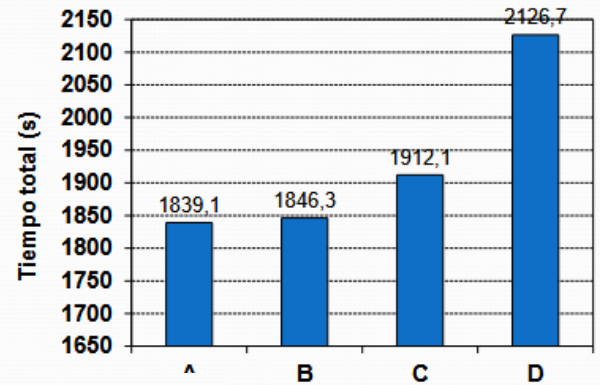
### COMPARACIÓN CON EL TIEMPO TOTAL

Me interesa comparar las 4 máquinas. No la de referencia  
→ esta la utilizamos para los índices solo.

Ordenación con el tiempo total:

De más rápida a más lenta: A, B, C, D

Esto no significa que A sea siempre la más rápida (depende del programa), aunque, en conjunto, sí que lo es.



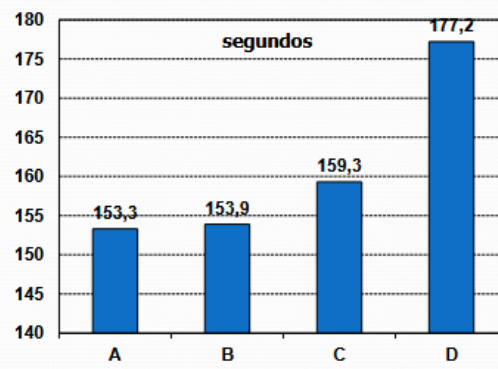
### COMPARACIÓN CON LA MEDIA ARITMÉTICA

$$\bar{t}_A = \frac{1}{12} \sum_{k=1}^{12} t_k^A = 153,3s$$

$$\bar{t}_B = \frac{1}{12} \sum_{k=1}^{12} t_k^B = 153,9s$$

$$\bar{t}_C = \frac{1}{12} \sum_{k=1}^{12} t_k^C = 159,3s$$

$$\bar{t}_D = \frac{1}{12} \sum_{k=1}^{12} t_k^D = 177,2s$$



Nos sola que la D es la peor y la A es la que gana. La máquina que ejecuta los programas del benchmark, uno tras otro, en menor tiempo es la de menos media aritmética de los tiempos de ejecución.

### USANDO LA MEDIA ARITMÉTICA PONDERADA

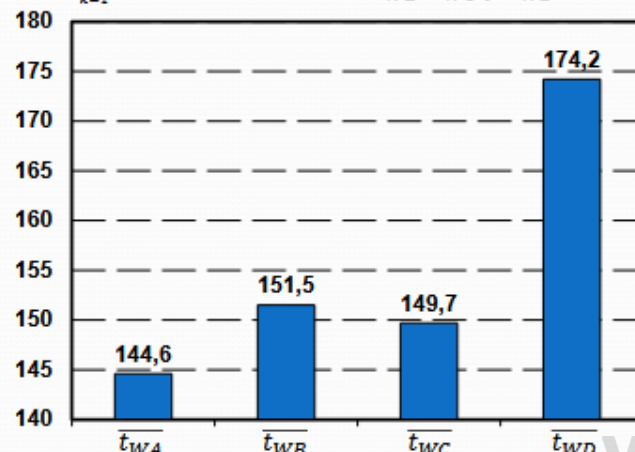
Prog	$t_{REF}$ (s)	$w_k$
1	1400	0,06
2	1400	0,06
3	1100	0,08
4	1800	0,05
5	1000	0,09
6	1200	0,07
7	1300	0,07
8	300	0,30
9	1100	0,08
10	1900	0,05
11	1500	0,06
12	3000	0,03
Suma	17000	1

$$w_k \equiv \frac{C}{t_k^{REF}}$$

$$C = \frac{1}{\sum_{k=1}^n 1/t_k^{REF}} = 88,77s$$

$$\bar{t}_{WA} = \sum_{k=1}^{12} w_k \times t_k^A = 144,6s$$

Igualmente, se calculan:  
 $\bar{t}_{WB}$ ,  $\bar{t}_{WC}$  y  $\bar{t}_{WD}$





El peso es inversamente proporcional a la que tarda la máquina de referencia en cada programa del benchmark.

*Programa 8 es el más importante (el que tiene más peso). Hace que la máquina C sea mejor que la máquina B ahora.*

Según este criterio, la máquina más rápida sería la de menor tiempo medio ponderado de ejecución. Nótese que esta ponderación depende, en este ejemplo, de la máquina de referencia.

### USANDO LA MEDIA GEOMÉTRICA DE SPEEDUPS

Calculamos la ganancia en velocidad de cada máquina con respecto a la máquina de referencia (tal y como lo hacen SPEC Y Sysmark):

Programa	$t^{\text{REF}}$ (s)	$s^A$ speedup	$s^B$ speedup	$s^C$ speedup	$s^D$ speedup
1	1400	9,9	8,2	10,3	10,4
2	1400	9,1	8,4	6,5	56,0
3	1100	11,4	11,7	7,5	5,5
4	1800	6,6	6,4	4,2	3,4
5	1000	11,9	11,1	12,9	12,3
6	1200	6,7	6,3	6,0	4,9
7	1300	10,8	9,9	14,8	17,2
8	300	2,0	1,9	2,2	1,6
9	1100	11,8	9,0	12,5	9,3
10	1900	14,3	11,0	16,1	13,4
11	1500	8,7	8,8	8,4	6,3
12	3000	12,3	30,0	30,0	20,0
M. Geom.		8,78	8,66	8,97	9,00

El speedup (ganancia en velocidad) es un índice a maximizar (cuanto más grande el speedup, mejor). Según este índice, la mejor máquina es... ¡la D!

Calculamos la media geométrica a los speedup → Nos da un índice SPEC → cuánto mas grande mejor → Sale la D mejor

Independientemente de la máquina de referencia (esos datos no van a cambiar), la máquina D es la que habría que comprar.

No hay que estar tan seguros.

### ¿A QUIÉN BENEFICIA LA DECISIÓN DE USAR LA MEDIA GEOMÉTRICA DE SPEEDUPS?

J8	=MEDIA.GEOM(J2:J5)									
	A	B	C	D	E	F	G	H	I	J
	Prog. Bench.	$t^{\text{REF}}$ (s)	$t^A$ (s)	$t^B$ (s)	$t^C$ (s)	$t^D$ (s)	$t^{\text{REF}}/t^A$	$t^{\text{REF}}/t^B$	$t^{\text{REF}}/t^C$	$t^{\text{REF}}/t^D$
1										
2	1	200	100	99	1	1	2,00	2,02	200,0	200,0
3	2	200	100	101	133	1	2,00	1,98	1,50	200,0
4	3	200	100	100	133	1	2,00	2,00	1,50	200,0
5	4	200	100	100	133	397	2,00	2,00	1,50	0,50
6	Suma	800	400	400	400	400				
7										
8				Media Geométrica						
							2,0000	2,0001	5,11	44,81

Tenemos los tiempos de una máquina de referencia (que tarda siempre 200s), máquina A, máquina B, máquina C (y que ejecuta el primer programa en un segundo) y máquina D (ejecuta 3 programas en 1 segundo). El tiempo total de ejecución de las 4 máquinas (sin contar la de referencia) es el mismo → 400 segundos.

Diríamos que las 4 máquinas se ejecutan en 400 segundos. Cada programa de media tardaría 100s en todas las máquinas. Todas tardan lo mismo de media.

Si aplicásemos la media ponderada, si dependiesen de los tiempos de ejecución de la máquina de referencia (recordemos que inversamente proporcional) coincidirían porque los tiempos de ejecución de la máquina de referencia son iguales.

Si hablamos del índice SPEC, intenta premiar las mejoras sustanciales en algún programa aunque sea en decremento en otros. Sin embargo, no se castigan empeoramientos no tan sustanciales. CUIDADO AL HACER COMPARACIONES CON ESTO! Hay que saber que estamos haciendo realmente.

### CONCLUSIONES DE ESTE ANÁLISIS

- ☞ Intentar reducir un conjunto de medidas de un test de rendimiento a un solo “valor medio” final no es una tarea trivial.
- ☞ La media aritmética de los tiempos de ejecución es una medida fácilmente interpretable e independiente de ninguna máquina de referencia. El menor valor nos indica la máquina que ha ejecutado el conjunto de programas del test, uno tras otro, en un tiempo menor.
- ☞ La media aritmética ponderada nos permite asignar más peso a algunos programas que a otros. Esa ponderación debería realizarse, idealmente, según las necesidades del usuario. Si se hace de forma dependiente de los tiempos de ejecución de una máquina de referencia, la elección de ésta puede influir significativamente en los resultados. Se debería usar de forma subjetiva a la hora de asignar pesos (yo soy el interesado en comprar la máquina)
- ☞ La media geométrica de las ganancias en velocidad con respecto a una máquina de referencia es un índice de interpretación compleja cuya comparación no depende de la máquina de referencia. Premia mejoras sustanciales con respecto a algún programa del test y no castiga al mismo nivel los empeoramientos

## 4.3. COMPARACIÓN DE PRESTACIONES EN PRESENCIA DE ALEATORIEDAD

### REPASO DE ESTADÍSTICA: DISTRIBUCIÓN NORMAL

Independientemente de qué índice se escoja, un buen ingeniero debería, en primer lugar, determinar si las diferencias entre las medidas obtenidas por un test de rendimiento en presencia de aleatoriedad son estadísticamente significativas. Nuestro objetivo es realizar una comparación de rendimiento donde pueda haber aleatoriedad.

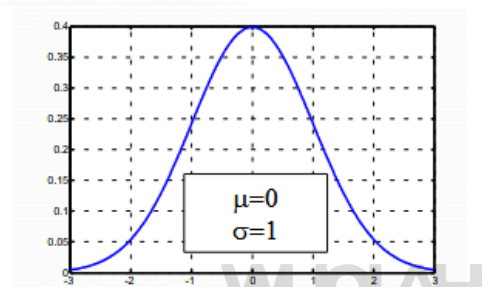
Necesitaremos repasar algunos conceptos de estadística.

Distribución normal: Es una distribución de probabilidad caracterizada por su media  $\mu$  y su varianza  $\sigma^2$  cuya función de probabilidad viene dada por:

$$Prob(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

\*No hay que saberse la fórmula\*

La probabilidad de obtener un elemento en el rango  $[\mu-2\sigma, \mu+2\sigma]$  es del 95%.



Si  $\mu=0$  y  $\sigma=1$ , habría una probabilidad del 95% de obtener un elemento en el rango  $[-2, 2]$ .

**Teorema del límite central:** la media de un conjunto grande de muestras aleatorias de cualquier distribución e independientes entre sí pertenece a una distribución normal. Es decir, me da igual la distribución de donde llegan los valores que se va a parecer a la distribución normal.

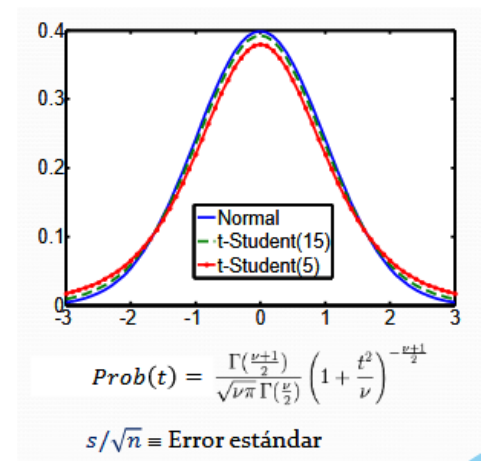
## DISTRIBUCIÓN T DE STUDENT

Su extraemos  $n$  muestras  $\{d_1, d_2, \dots, d_n\}$  pertenecientes a una distribución Normal de media  $\bar{d}_{real}$  y calculo la siguiente medida (=estadístico):

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

siendo  $\bar{d}$  la media muestral (es la media de la distribución normal) y  $s$  la desviación típica muestral (estimación de  $\sigma$ )

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



y repetimos el experimento muchas veces, veremos que estos  $t_{exp}$  pertenecen a una distribución t-Student con  $n-1$  grados de libertad.

Por ejemplo, si tenemos 100 muestras:

$$\bar{d} = \frac{d_1, d_2 \dots d_{100}}{100}$$

Repetimos y calculamos  $t_{exp}$  una y otra vez  $\rightarrow$  100 valores, otros 100 valores, otros 100 valores...  
Dibujo qué distribución me sale  $\rightarrow$  distribución t-student  $\rightarrow$  tiene 1 solo parámetro.

¿Para qué me puede servir esto?

## EJEMPLO 1: TEST DE RENDIMIENTO ENTRE A Y B

Tiempos de ejecución (en segundos) de 6 programas (P1...P6) en dos máquinas diferentes (A y B) en condiciones donde puede haber alta aleatoriedad. Por tanto  $n=6$ .(programas).

Puede haber aleatoriedad al calcular el tiempo de ejecución porque hay presencia de otros procesos. No tengo la certeza de que los tiempos de ejecución sean los mismos

Programa	$t_A$ (s)	$t_B$ (s)	$d = t_A - t_B$ (s)
P1	142	100	42
P2	139	92	47
P3	152	128	24
P4	112	82	30
P5	156	148	8
P6	166	171	-5

$\bar{t}_A = 144,5s$   
 $\bar{t}_B = 120,2s$   
**¿Son significativas estas diferencias?**  
 $\bar{d} = 24,3s \quad s = 19,9s \quad s/\sqrt{n} = 8,12s$

¿Compro la B porque me sale mejor el tiempo de ejecución? No me la quiero jugar. Tengo que asegurarme. Debo comprobar si las diferencias son significativas o no.

Calculo las diferencias de los dos máquinas para cada programa. (cuarta columna de la tabla)

Si partimos de la hipótesis (“hipótesis nula”,  $H_0$ ) de que las máquinas tienen rendimientos equivalentes, entonces las diferencias se deben a una suma (= una media) de factores aleatorios independientes. En este caso di serán muestras de una distribución normal de media cero ( $\underline{d}_{real}=0$ ). Es decir, partimos de la hipótesis de que  $Rend(A) \equiv Rend(B)$ . Si es verdad y la diferencia es mayor que 0, es por aleatoriedad (hay elementos aleatorios). Sin aleatoriedad, los tiempos deberían ser iguales/equivalentes y la diferencia debería ser 0. Supongo que la diferencia mayor que 0 es por elementos/factores aleatorios. Si esto es verdad,  $t_{exp}$  debería pertenecer a una distribución. Por tanto:

$$n = 6 \quad | \quad \underline{d} = 24.3s \quad | \quad s = 19.9s \quad | \quad s/\sqrt{n} = 8.12$$

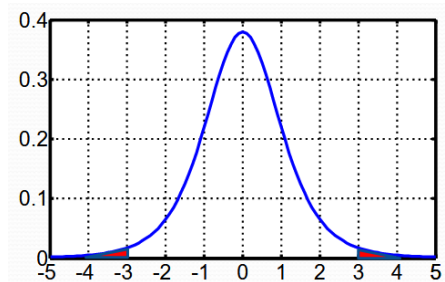
$$t_{exp} = \frac{\bar{d}}{s/\sqrt{n}} = \frac{24,3s}{8,12s} = 2,99$$

**CUIDADO CON LAS UNIDADES!** → Poner bien en el examen

Si es verdad la hipótesis,  $t_{exp}$  debería pertenecer a una distribución  $t_{Student}$  de 5 grados de libertad.

¿ $t_{exp} \sim T_5$ ?

¿Cuál es la probabilidad de encontrar un valor 2,99 o peor en T-Student? = Calcular el área de:



El  $p\_value$  con 5 grados de libertad es de 0.03.

La probabilidad de obtener un valor de  $|t|$  igual o superior a 2,99 de una distribución  $t$  de Student con 5 grados de libertad es de 0,03 (P-value (Valor-P) = 0,03). ¿Es eso mucho o poco? Debemos definir un umbral para establecer lo que es mucho o poco: el nivel o grado de significatividad  $\alpha$ . Normalmente,  $\alpha=0,05$  (5%) por defecto.

Conclusión:

1. Si  $P\text{-value} < \alpha$  diremos que, para un grado de significatividad  $\alpha$  o para un nivel de confianza  $(1-\alpha) * 100$  (normalmente 95%), las máquinas tienen rendimientos estadísticamente diferentes. Se rechazaría la Hipótesis Nula.  
 $Nivel\ de\ confianza = (1 - \alpha) * 100(\%) = (1 - 0.05) * 100(\%) = 95\%$
2. Si  $P\text{-value} \geq \alpha$ , se aceptaría la hipótesis nula.

En nuestro ejemplo rechazo la hipótesis. Acepto los valores y me quedo con la máquina B.

En ese caso, B sería, de media, 1,2 veces más rápida que A en ejecutar cada programa ( $144,5/120,2 = 1,2$ ). En caso contrario, no podríamos descartar la hipótesis de que las máquinas tengan rendimientos equivalentes.

Si el nivel de confianza es 95% y  $1 - \alpha$ , entonces  $\alpha$  es 0.05

### INTERVALOS DE CONFIANZA PARA $t_{exp}$

En lugar de hallar  $p$ , quiero que me de un valor absoluto (antes 2 en distribución normal). Para un nivel de significatividad  $\alpha$  (tip 0.05 = 5%), buscamos el valor  $t_{\alpha/2, n-1}$  que cumpla  $Prob(|t| > t_{\alpha/2, n-1}) = \alpha$  o equivalentemente:

$$Prob\left(-t_{\frac{\alpha}{2}, n-1} \leq t \leq t_{\frac{\alpha}{2}, n-1}\right) = 1 - \alpha$$

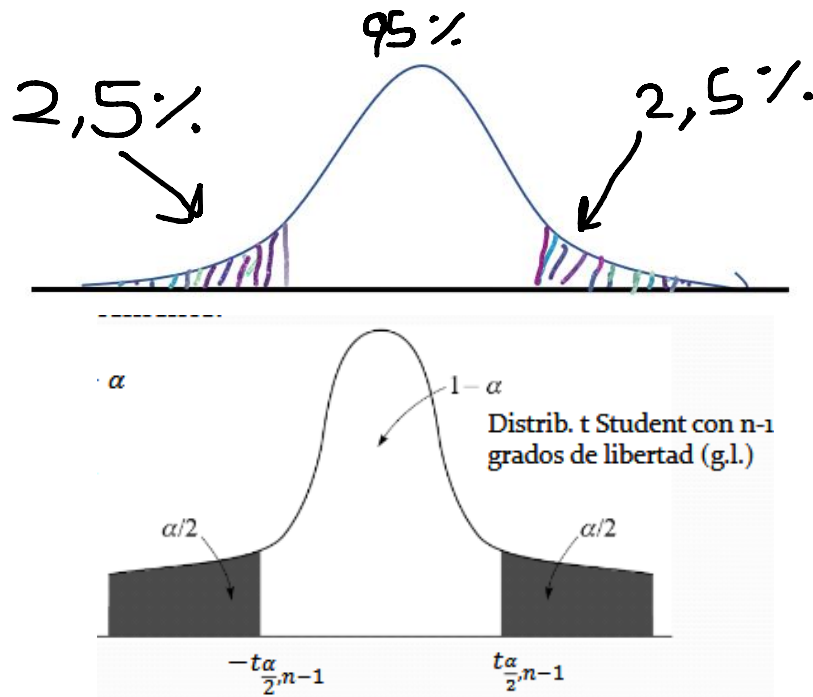
Diremos que para un nivel de confianza  $1 - \alpha$  (tip.  $0.05 = 95\%$ ), para aceptar  $H_0$  el valor de  $t_{exp}$  debería situarse en el

$$\left[ -t_{\frac{\alpha}{2}, n-1}, t_{\frac{\alpha}{2}, n-1} \right]$$

intervalo:

A dicho intervalo se le denomina intervalo de confianza de la medida para un nivel de significatividad  $\alpha$ .

$\alpha/2$  porque se trata de hallar el área del trozo de la izquierda y de la derecha



df = degrees of freedom (n-1)  $\rightarrow$  grados de libertad

En el caso del *Ejemplo 1*, para un nivel de significatividad de  $\alpha=0.05$ , buscamos  $t_{\alpha/2, n-1}$  tal que:

$$Prob(t \leq -t_{\frac{\alpha}{2}, n-1}) = \alpha/2 = 0,025$$

para una distribución t de Student con 5 grados de libertad ( $df = 6-1 = 5$ ). Es decir, no hace falta hallar  $p\_value$ . Miro si  $t_{exp}$  está en el rango hallado. Eso se puede obtener consultado las tablas estadísticas.

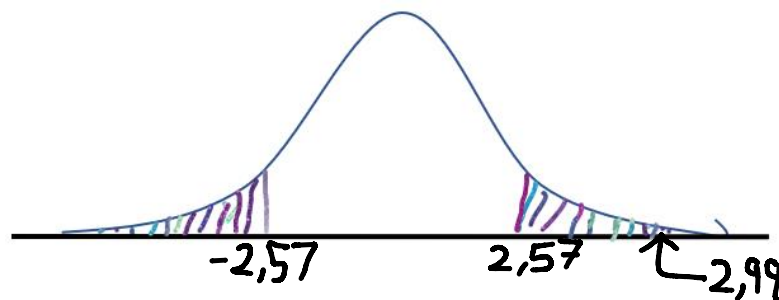
$$t_{\alpha/2, n-1} = 2.57$$

Dicho de otra forma, si las diferencias entre los tiempos de ejecución de ambas máquinas se debieran a factores aleatorios, existiría un 95% probabilidad de que

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

Se encuentre en el rango  $[-t_{\alpha/2, n-1}, t_{\alpha/2, n-1}] = [-t_{0.025, 5}, t_{0.025, 5}] = [-2.57, 2.57]$

Como  $t_{exp} = 2.99$  no está en ese rango, concluiremos nuevamente que rechazamos la hipótesis de que ambas máquinas tienen rendimientos equivalentes con el 95% de confianza.



Cualquier valor de  $t_{exp}$  que esté fuera del rango, lo voy a rechazar. Como 2.99 está fuera, lo rechazo. Rechazo la  $H_0 \rightarrow$  mismo resultado que antes

### INTERVALOS DE CONFIANZA PARA $\underline{d}_{real}$

A veces en vez de darme  $p$  o el valor de confianza me dan  $t_{exp}$ . Despejamos  $\underline{d}_{real}$  considerando que  $H_0$  es cierta.

Acabamos de ver que si las diferencias entre los tiempos de ejecución en ambas máquinas se deben a factores aleatorios, existen un 95% de probabilidad de que  $t_{exp}$  se encuentre en el rango  $[-t_{\alpha/2, n-1}, t_{\alpha/2, n-1}] = [-2.57, 2.57]$ .

Como

$$t_{exp} = \frac{\bar{d} - \underline{d}_{real}}{s/\sqrt{n}} \in [-|t_{\alpha/2, n-1}|, |t_{\alpha/2, n-1}|] = [-2.57, 2.57]$$

sin más que identificar  $t_{exp}$  con los valores límite  $\pm t_{\alpha/2, n-1}$  sabemos que, de ser  $H_0$  cierta, habrá un 95% de probabilidad de que el valor medio real  $\underline{d}_{real}$  las diferencias entre los tiempos de ejecución se encuentre en el intervalo:

$$\underline{d}_{real} \in \left[ \bar{d} - \frac{s}{\sqrt{n}} \times |t_{\alpha/2, n-1}|, \bar{d} + \frac{s}{\sqrt{n}} \times |t_{\alpha/2, n-1}| \right] = 24,3 \mp 20,9 = [3,4, 45,2] \text{ s}$$

Y el problema se transforma simplemente en comprobar si ese valor medio real  $\underline{d}_{real}$  puede o no ser cero.

Si  $H_0$  es cierta,  $\underline{d}_{real}$  debe estar en medio del intervalo. Sin embargo, dijimos que tenía que ser 0. Otra forma, por tanto, es calcular el intervalo de confianza y ver si el 0 está incluido. Si no está incluido, rechazamos  $H_0$ .

En nuestro ejemplo, como el intervalo no incluye el cero, concluiremos una vez más que la hipótesis de que ambas máquinas pueden tener rendimientos equivalentes no es cierta al 95% de confianza.

### RESUMEN: TEST T PARA MUESTRAS PAREADAS

- Partimos de:

Exp.	tA	tB	$d_i = tA_i - tB_i$
$P_1$	$tA_1$	$tB_1$	$d_1$
$P_2$	$tA_2$	$tB_2$	$d_2$
...	...	...	...
$P_n$	$tA_n$	$tB_n$	$d_n$

- $H_0$ : Rendimiento A  $\equiv$  Rendimiento B, es decir,  $d_i \sim N(\underline{d}_{real}, \sigma^2)$  con  $\underline{d}_{real} = 0$

- Calculo  $t_{exp} = \frac{\bar{d} - \underline{d}_{real}}{s/\sqrt{n}} \sim T_{n-1}$  siendo  $\bar{d} = \frac{\sum_{i=1}^n d_i}{n}$   $s = \sqrt{\frac{\sum_{i=1}^n (d_i - \bar{d})^2}{n-1}}$



- Definimos el nivel o grado de significatividad  $\alpha$
- Rechazamos  $H_0$  para un nivel de confianza  $(1 - \alpha) \cdot 100(\%)$  si:
  1. Método 1:  $p\text{-value} < \alpha$ . Siendo  $p\text{-value} = P(|t| > |t_{\text{exp}}|)$  en  $T_{n-1} \approx \text{Prob}(H_0 \text{ podría ser cierta})$
  2. Método 2:  $t_{\text{exp}} \notin [-t_{\alpha/2, n-1}, t_{\alpha/2, n-1}]$ . Siendo  $t_{\alpha/2, n-1}$  el valor que hace que  $\text{Prob}(t \leq -t_{\alpha/2, n-1}) = \alpha/2$  para una distribución t de Student con n-1 grados de libertad
  3. Método 3: Intervalo de confianza para  $\underline{d}_{\text{real}}$

$$0 \notin \left[ \bar{d} - \frac{s}{\sqrt{n}} \times t_{\frac{\alpha}{2}, n-1}, \bar{d} + \frac{s}{\sqrt{n}} \times t_{\frac{\alpha}{2}, n-1} \right].$$

## EJEMPLO 2: ¿INFLUYE EL PARÁMETRO SWAPPINESS DEL SO EN ESTE SERVIDOR?

Productividades (en páginas webs/s) obtenidas por el servidor en 5 experimentos diferentes para dos valores diferentes (A y B) del parámetro proxy\_cache\_min\_uses de Nginx en condiciones donde puede haber alta aleatoriedad.

Experimento	$X_A$ (pág/s)	$X_B$ (pág/s)	$d = X_A - X_B$ (pág/s)
Exp1	23	15	8
Exp2	28	22	6
Exp3	19	20	-1
Exp4	29	27	2
Exp5	36	39	-3

Podría decir ciegamente que A es mejor que B. Usando como criterio la media aritmética ( $X_A = 27$  pág/s,  $X_B = 25$  pág/s) parece que el parámetro A obtiene mejor productividad que el B pero, ¿son significativas las diferencias para un nivel de confianza del 95%?  $(1 - \alpha) \cdot 100 = 95\% \rightarrow$  Grado de significatividad  $\alpha = 0.05$ . Deberíamos asegurarnos, antes de sacar una conclusión, que las diferencias son significativas ya que estamos en un entorno con gran aleatoriedad. Lo que resulte será con un nivel de confianza del 95%.

Calculo las diferencias de los valores obtenidos (columna “d”)

Hago la siguiente hipótesis ( $H_0$ ):

Rendimiento A  $\equiv$  Rendimiento B, es decir,  $d_i \sim N(d_{\text{real}}, \sigma^2)$  con  $\underline{d}_{\text{real}} = 0$ . Si es verdad, la diferencia debe estar dentro de la distribución normal.

$$\begin{aligned} \bar{d} &= \frac{\sum_{i=1}^n d_i}{n} = \frac{8 + 6 - 1 + 2 - 3}{5} = 2,4 \text{ pág/s} \\ s &= \sqrt{\frac{\sum_{i=1}^n (d_i - \bar{d})^2}{n - 1}} = \sqrt{\frac{\sum_{i=1}^n d_i^2 - n \cdot \bar{d}^2}{n - 1}} = 4,6 \text{ pág/s} \\ \frac{s}{\sqrt{n}} &= \frac{4,6}{\sqrt{5}} = 2,06 \text{ pág/s} \quad t_{\text{exp}} = \frac{\bar{d}}{s/\sqrt{n}} = \frac{2,4}{2,06} = 1,16 \end{aligned}$$

Tengo que preguntarme qué probabilidad hay de que 1,16 pertenezca a una distribución TStudent de  $5 - 1 = 4$  grados de libertad. ( $T_4$ )  $\rightarrow$  ( $1,16 \approx T_4$ )

Método 1:  $p\text{-value} = 0.31$ . Hay 31% de probabilidad de encontrarme  $t_{\text{exp}}$  (1,16) en esa distribución. Bajo es por debajo del 5%. Como  $p\text{-value}$  es mayor que  $\alpha$ , no podemos rechazar la hipótesis nulas. Puedo decir que  $\text{Rend}(A) \equiv \text{Rend}(B)$ . No puedo decir que una máquina es mejor que otra con un 95% de confianza. Los parámetros A y B sí podrían tener rendimientos equivalentes).

Pero ¿y si me dan otro valor en lugar de p\_value?

Método 2 (intervalos de confianza para  $t_{exp}$ ):  $\alpha = 0.05$  y  $df = n-1 = 4$ . Miro la tabla y  $t_{\alpha/2, n-1} = t_{0.025, 4} = 2.78$

Como  $t_{exp} = 1.16 \in [-2.78, 2.78] \rightarrow$  no puedo rechazar al 95% la hipótesis nula  $H_0 \rightarrow$  Coincide con lo que nos salía en el primer método: los parámetros A y B si podrían tener rendimientos equivalentes).

Método 3 (intervalos de confianza para  $\underline{d}_{real}$ ):

$$\left[ \bar{d} - \frac{s}{\sqrt{n}} \times t_{\frac{\alpha}{2}, n-1}, \bar{d} + \frac{s}{\sqrt{n}} \times t_{\frac{\alpha}{2}, n-1} \right] = [2.4 - 2.06 \times 1.16, 2.4 + 2.06 \times 1.16] = [-3.3, 8.1] \text{ pág/s.}$$

Despejo  $\underline{d}_{real} \rightarrow$  el intervalo de confianza si  $H_0$  es cierta incluye el valor 0. No podemos rechazar la hipótesis nula  $H_0$  (que los rendimientos son equivalentes)

### \*TEST T CON JASP\*

Posible ejercicio: Posibilidad 4: Me pone un ejercicio sin datos, me da la tabla final de JASP y tengo que decir qué está pasando

### OTRA UTILIDAD DEL TEST T: ESTIMACIÓN DE INTERVALOS DE CONFIANZA DE MEDIAS DE MEDIDAS EXPERIMENTALES

Hipótesis: Realizamos  $n$  medidas  $\{d_1, d_2, \dots, d_n\}$  de un mismo fenómeno (p.ej. tiempos de ejecución de un programa, tiempos acceso de un disco duro, productividades de red, etc). Si éstas pueden diferir debido a una suma de efectos aleatorios, podemos suponer que se distribuyen según una normal de media  $\underline{d}_{real}$ , que es el valor que buscamos. En ese caso, sabemos que:

$$t_{exp} = \frac{\bar{d} - \underline{d}_{real}}{s/\sqrt{n}}$$

Pertenece a la distribución t-Student con  $n-1$  grados de libertad, siendo  $\bar{d}$  y  $s$  la media y la desviación típica muestrales, respectivamente.

Por tanto, hay un  $(1 - \alpha) * 100\%$  de probabilidad de que el valor medio real  $\underline{d}_{real}$  se encuentre en el intervalo:

$$\bar{d} \pm \frac{s}{\sqrt{n}} t_{\alpha/2, n-1}$$

Utilidad: Podemos usar esta información para determinar un intervalo de confianza para  $\underline{d}_{real}$  y no quedarnos simplemente con el valor medio muestral

#### Ejemplo:

Queremos determinar un intervalo de confianza del 95% para el tiempo medio de escritura de un determinado fichero en un disco duro. Para ello, se han realizado  $n=8$  medidas experimentales:

#exp	d (ms)	$\bar{d} = \frac{\sum_{i=1}^n d_i}{n} = 820ms$ $s = \sqrt{\frac{\sum_{i=1}^n (d_i - \bar{d})^2}{n-1}} = 14ms$ $ t_{\frac{\alpha}{2}, n-1}  =  t_{0.025, 7}  = 2.36$		<ul style="list-style-type: none"><li>En Excel, haciendo: ABS(INV.T(alfa/2;n-1)).</li><li>En Calc, DISTR.T.INV(alfa;n-1).</li><li>En Matlab, haciendo: abs(tinv(alfa/2,n-1)).</li></ul>
1	835			
2	798			
3	823			
4	803			
5	834			
6	825			
7	813			
8	829			

df	0.20	0.10	0.05	0.02	0.01	0.001
1	3.0777	6.3138	12.7062	31.8205	63.6567	636.6192
2	1.8856	2.9200	4.3027	6.9646	9.9248	31.5991
3	1.6377	2.3534	3.1824	4.5407	5.8409	12.9240
4	1.5332	2.1318	2.7764	3.7469	4.6041	8.6103
5	1.4759	2.0150	2.5706	3.3649	4.0321	6.8688
6	1.4296	1.9432	2.4469	3.1427	3.7074	5.9588
7	1.4149	1.8946	2.3646	2.9980	3.4995	5.4079

Por tanto, hay un 95% ( $\alpha=0.05$ ) de probabilidad de que el tiempo medio de escritura real de ese fichero se encuentre en el intervalo:

$$\bar{d} \pm \frac{s}{\sqrt{n}} t_{\alpha/2, n-1} = 820 \pm \frac{14}{\sqrt{8}} t_{0.05, 8-1} = [808, 832]ms$$

## 4.4. DISEÑO DE EXPERIMENTOS DE COMPARACIÓN DE RENDIMIENTO

### PLANTEAMIENTO DEL PROBLEMA

Supongamos que queremos determinar cuáles de los siguientes factores afectan significativamente al rendimiento de un determinado servidor:

1. Sistema Operativo: Windows Server, CentOS, Debian, Ubuntu
2. Memoria RAM: 32GB, 64GB, 128GB
3. Discos duros: SAS, SATA, IDE (=P-ATA)

Y, en el caso de que afecten, cuál de los niveles del factor es significativamente mejor que el resto.

¿Qué experimentos debemos diseñar para ello y cómo debemos analizar los resultados?

Con test t puedo analizar dos alternativas, pero ¿y si tengo 3, 5 o más? ¿Y si no solo se trata del valor de un parámetro sino también de otros como el SO, RAM...? Hay más factores que influyen.

Se trata de hacer un test de rendimiento con varios factores y varias alternativas.

### TERMINOLOGÍA

Variable respuesta o dependiente (métrica): El índice de rendimiento que usamos para las comparaciones. *P.ej. tiempos de respuesta (R), productividades (X). Se trata de qué medida vamos a tomar para decidir qué es mejor o no.*

Factor: Cada una de las variables que pueden afectar a la variable respuesta. *P.ej. sistema operativo, tamaño de memoria, tipo de disco duro, tipo de procesador, número de microprocesadores, número de cores, tamaño de cada caché, compilador, algún parámetro configurable del S.O. o del servidor, etc. Qué puede influir en el rendimiento de un servidor.*

Nivel: Cada uno de los valores que puede asumir un factor. *P.ej. para un S.O.: Windows, CentOS, Debian, Ubuntu; para un tipo de disco duro: SATA, IDE, SAS; para un parámetro del sistema operativo: ON, OFF, etc.*

Interacción: Una interacción ocurre cuando el efecto de un factor cambia para diferentes niveles de otro factor. Complica las cosas. *P.ej. el hecho de usar un tipo determinado de S.O. puede afectar a cómo de importante sea usar una mayor cantidad de memoria DRAM*

Usar un SO u otro, puede hacer que un nivel de otro factor influya de una forma u otra. Hay dependencias entre los niveles de los factores. Por tanto, todo se complica cuando hay interacción.

### TIPOS DE DISEÑOS EXPERIMENTALES

- ☉ Diseños con un solo factor: Se utiliza una configuración determinada como base y se estudia un factor cada vez, midiendo los resultados para cada uno de sus niveles. No hay interacción. Cojo uno como base. *Por ejemplo digo si influye el SO. Modifico los niveles. Uno para Windows, CentOS... solo tengo que hacerlo para ellos.*  
Igual con el resto de los factores. El número de experimentos que hay que hacer es pequeño (en nuestro caso, 8)

Problema: solo válida si descartamos que haya interacción entre factores. Es decir, cuando hay interacción no es tan sencillo. Número total de experimentos =  $1 + \sum_{i=1}^k (n_i - 1)$  donde k es el número de factores y  $n_i$  el número de niveles del factor i. En nuestro ejemplo, habría que hacer 8 experimentos.

- ⊙ Diseños multi-factoriales completos: Se prueba cada posible combinación de niveles para todos los factores. Se realizan un montón de experimentos. Ventaja: se analizan las interacciones entre todos los factores. Número total de experimentos =  $\prod_{i=1}^k n_i$ . En nuestro ejemplo: 36 experimentos.
- ⊙ Diseños multi-factoriales fraccionados: Término medio entre los anteriores. No todas las interacciones se verán reflejadas en los resultados, solo las de las interacciones que se consideren más probables

Todos ellos se pueden realizar con diferentes niveles de repetición:

- sin repeticiones,
- con todos los experimentos repetidos el mismo número de veces,
- con un número de repeticiones diferentes para cada nivel o cada factor.

## DISEÑOS CON UN SOLO FACTOR

No hay interacción.

Ejemplo: Para el servidor principal de nuestra empresa, queremos saber si la elección del tipo de disco duro afecta al rendimiento. Para ello, se ha escogido tres discos duros con tres interfaces distintas: SAS, SATA e IDE y se ha realizado un experimento que consiste en ejecutar, en condiciones reales y en presencia de aleatoriedad, un conjunto de programas usados habitualmente por el servidor y medir el tiempo de ejecución.

El factor por tanto es la interfaz del disco duro. Es lo mismo pero con 3 niveles en vez de dos. Hemos añadido una columna.

Este experimento se ha repetido 5 veces:

#Exp.	SAS (s)	SATA (s)	IDE (s)
1	103	115	143
2	97	102	134
3	123	120	139
4	106	115	135
5	116	122	129
<b>Medias</b>	<b>109.0</b>	<b>114.8</b>	<b>136.0</b>
<b>Efectos (<math>\epsilon_j</math>)</b>	<b>-10.9</b>	<b>-5.1</b>	<b>16.1</b>

$$m_{\text{global}} = 119.9s$$

Según el valor medio para cada nivel del factor, parece que SAS es mejor que SATA pero, ¿son las diferencias entre estos significativas?

Nos hacemos dos preguntas: ¿Tiene influencia el factor “interfaz del disco duro” sobre el rendimiento? ¿Son las diferencias entre los discos duros significativas? → Test ANOVA de un factor.

## ANÁLISIS DE LA VARIANZA (ANOVA) DE UN FACTOR

$$\text{Modelo: } y_{ij} = m_{\text{global}} + \varepsilon_j + r_{ij} \quad i=1, \dots, n_{\text{rep}}; \quad j=1, \dots, n_{\text{niv}}$$

$y_{ij}$ : Las observaciones. En nuestro caso los tiempos de ejecución obtenidos en cada prueba. El índice  $j$  recorre los distintos niveles del factor cuya influencia se quiere medir (en nuestro caso hay  $n_{\text{niv}}=3$  niveles: SAS, SATA e IDE). El índice  $i$  recorre las distintas repeticiones para cada uno de esos niveles (en nuestro caso,  $n_{\text{rep}}=5$  repeticiones).

$m_{\text{global}}$ : Media global de todas las observaciones:

$$m_{\text{global}} = \frac{1}{n_{\text{rep}} \times n_{\text{niv}}} \sum_{i=1}^{n_{\text{rep}}} \sum_{j=1}^{n_{\text{niv}}} y_{ij}$$

$\varepsilon_j$ : Efecto debido al nivel  $j$ -ésimo: Efecto: resto la media local-global

$$\varepsilon_j = \frac{1}{n_{\text{rep}}} \sum_{i=1}^{n_{\text{rep}}} y_{ij} - m_{\text{global}}. \text{ Se cumple que } \sum_{j=1}^{n_{\text{niv}}} \varepsilon_j = 0.$$

Por ejemplo, el efecto de usar SAS es bajarle 10,9s a la media global.

$r_{ij}$ : Perturbaciones o error experimental (ruido). Deben cumplir:

- Que tengan varianza constante, independiente del nivel (si n° de exp./nivel no es homogéneo).
- Que su distribución sea normal.

Si no se cumplen, hay otros test alternativos: test de Kruskal-Wallis, test de Friedman.

La principal pregunta que intenta contestar el test ANOVA es: ¿Tiene influencia el factor sobre la variable respuesta o, dicho de otro modo, algún  $\varepsilon_j$  es distinto de cero? ¿Los efectos pueden ser 0 desde el punto de vista estadístico? Por ejemplo: ¿Influye o no el factor disco duro? Para esto se hace el test ANOVA.

El método de ANOVA se basa en descomponer la varianza de las muestras en:

$$\sum_{i=1}^{n_{\text{rep}}} \sum_{j=1}^{n_{\text{niv}}} (y_{ij} - m_{\text{global}})^2 = n_{\text{rep}} \sum_{j=1}^{n_{\text{niv}}} (\varepsilon_j)^2 + \sum_{i=1}^{n_{\text{rep}}} \sum_{j=1}^{n_{\text{niv}}} (r_{ij})^2$$

Utilizando notación abreviada:  $SST = SSA + SSE$

- ☞ SST= Varianza total de las muestras. (Sum-of-Squares Total)
- ☞ SSA= Varianza explicada por los efectos o alternativas (intergrupos). (Sum-of-Squares Alternatives)
- ☞ SSE= Varianza residual o del error (intragrupos) (Sum-of-Squares Error)

El objetivo es contrastar la hipótesis ( $H_0$ ) de que el factor no influye sobre los resultados  $\varepsilon_j \approx 0 \forall j = 1 \dots n_{\text{niv}}$ . Si esto es cierto, resulta que el resultado de hacer:

$$F_{\text{exp}} \equiv \frac{SSA / (n_{\text{niv}} - 1)}{SSE / (n_{\text{niv}} \times (n_{\text{rep}} - 1))} \sim F_{n_{\text{niv}} - 1, n_{\text{niv}} \times (n_{\text{rep}} - 1)}$$

debería ser una muestra de una distribución F de Snedecor con  $n_{\text{niv}}-1$  grados de libertad en el numerador y  $n_{\text{niv}} \times (n_{\text{rep}}-1)$  en el denominador.

Idea básica: Tenemos un test y nos preguntamos: ¿Influye en el rendimiento del servidor mi disco duro? (p.e). Utilizo el test ANOVA → Descompongo cada valor del test en  $m_{\text{global}}$  + efecto + ruido.  
→ Planteo una hipótesis de la que parto: “El factor no influye en el rendimiento” ( $SAS \equiv SATA \equiv IDE$ )

¿Es posible que el efecto pueda ser 0? El estadístico hace sus cuentas. Calculará un número (no hace falta sabérselo porque es complejo). Si es verdad que la hipótesis es cierta, el numerito debería pertenecer a la distribución F de Snedecor....

En nuestro ejemplo:

SST = 2809

SSA = 2020

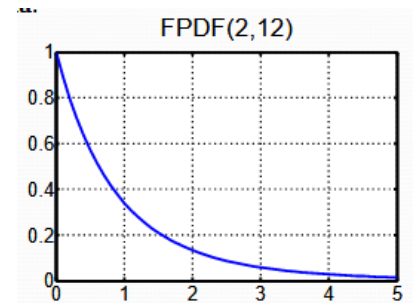
SSE = 789

$$F_{exp} \equiv \frac{\frac{SSA}{n_{niv} - 1}}{\frac{SSE}{(n_{niv} \times (n_{rep} - 1))}} = \frac{\frac{2020}{3 - 1}}{\frac{789}{(3 \times (5 - 1))}} = 15,37$$

¿Qué probabilidad hay de que la muestra 15,37 o superior se haya extraído de una distribución  $F_{2,12}$ ?  
p-value =  $P(F \geq 15,37; 2, 12) = 0.00049$ . Identifico ese valor como la probabilidad de que la hipótesis ( $H_0$ ) de que el factor no influye pueda ser cierta.

Si la probabilidad es menor que  $\alpha = 0.05$ , diremos que descartamos la hipótesis de que el factor no influye a un  $(1 - \alpha) \times 100\% = 95\%$  de confianza.

Si el factor influye, a continuación (análisis post-hoc) comparamos las medias de cada nivel unas con otras usando un test t: prueba de múltiples rangos o de comparaciones múltiples.



Partimos de:

#Experimento	Rendimiento nivel 1	Rendimiento nivel 2	...	Rendimiento nivel $n_{niv}$
1	$y_{11}$	$y_{12}$		$y_{1n_{niv}}$
2	$y_{21}$	$y_{22}$		$y_{2n_{niv}}$
...	...	...		...
$n_{rep}$	$y_{n_{rep}1}$	$y_{n_{rep}2}$		$y_{n_{rep}n_{niv}}$

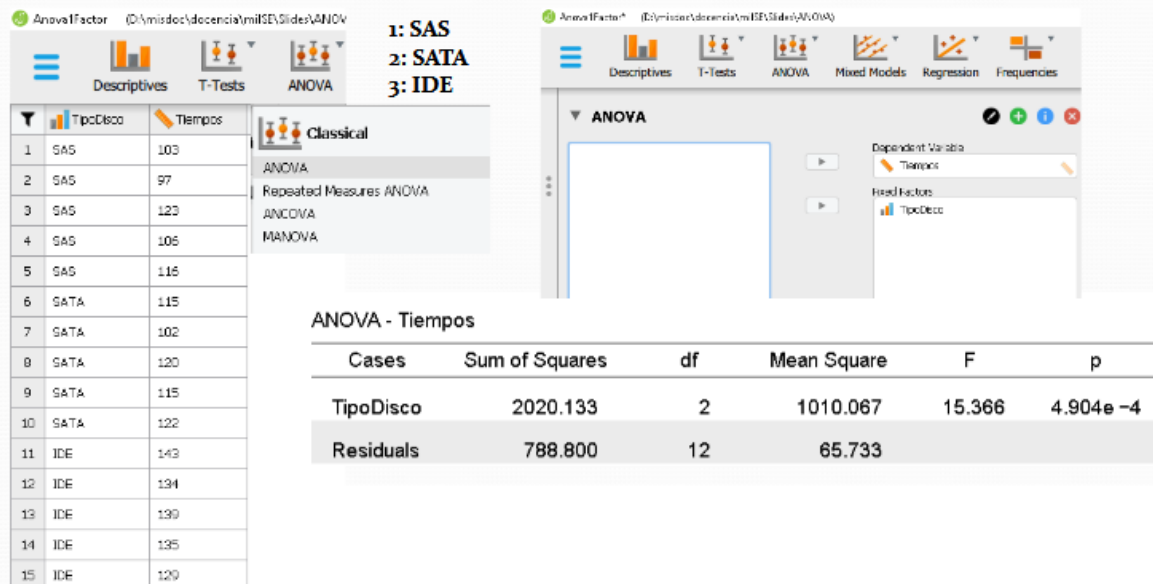
- **Modelo:**  $y_{ij} = m_{\text{global}} + \varepsilon_j + r_{ij}$  ;  $i=1, \dots, n_{rep}$  ;  $j=1, \dots, n_{niv}$
- $H_0$ : Rendimiento de todos los niveles del factor es equivalente ( $\varepsilon_j = 0$ ,  $j=1, \dots, n_{niv}$ )  $\equiv$  El factor no influye en el rendimiento
- Se calcula 
$$F_{exp} \equiv \frac{SSA / (n_{niv} - 1)}{SSE / (n_{niv} \times (n_{rep} - 1))} \sim F_{n_{niv} - 1, n_{niv} \times (n_{rep} - 1)}$$
- $p\_value \approx \text{Prob}(H_0 \text{ podría ser cierta})$
- Rechazamos  $H_0$  para un nivel de confianza  $(1 - \alpha) \times 100(\%)$  si valor  $p < \alpha$
- En este caso comparamos las medias de cada nivel unas con otras usando un test t: prueba de múltiples rangos o de comparaciones múltiples



En resumen: Parto de  $H_0 \rightarrow \text{Rend}(\text{nivel } 1) = \text{Rend}(\text{nivel } 2) = \dots = \text{Rend}(\text{nivel } n)$ . Esto es igual a decir que el factor no influye en el rendimiento.  $\rightarrow$  Se hace un test ANOVA. Si eso es verdad, el valor debe pertenecer a la distribución.

Alguien tendría que darnos  $p\_value$ . Si  $p < \alpha \rightarrow$  rechazo

### ANOVA DE UN FACTOR CON JASP (jasp-stats.org)

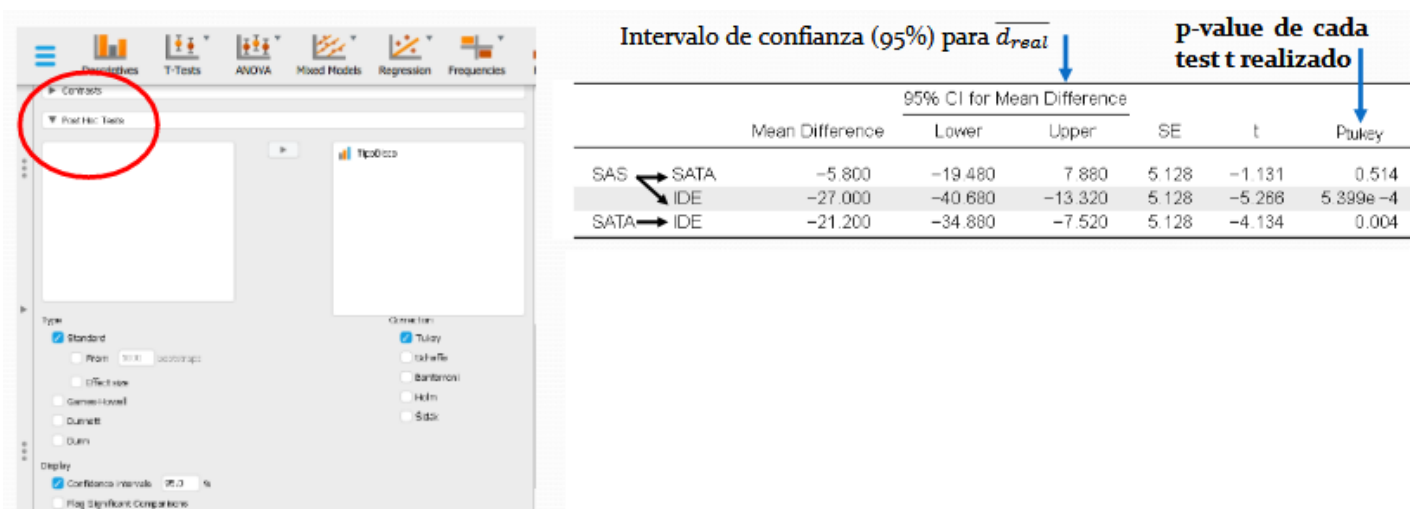


Probabilidad baja  $\rightarrow$  Rechazo  $H_0 \rightarrow$  el factor influye ( $p < \alpha$ ).

Rechazaremos la hipótesis  $H_0$  para  $\alpha=0.05$  (al menos). Concluimos que para un nivel de confianza del 95%, la interfaz de disco duro afecta significativamente al rendimiento del equipo.

### ANOVA DE UN FACTOR CON JASP (post-hoc test)

Como el factor influye, hacemos ahora un test t entre cada combinación de niveles para comparar el efecto en el rendimiento de cada tipo de disco duro: prueba de múltiples rangos o de comparaciones múltiples. Es decir, una vez veo si el factor influye, hago un test todos con todos.



La primera fila de la tabla es un test t entre SAS y SATA, la segunda entre SAS e IDE y la tercera entre SATA e IDE.

La última columna es el p-value de cada test t realizado.

¿Cuál es mejor? SAS-SATA  $\rightarrow p=0.514 (>\alpha) \rightarrow 51\% \rightarrow$  con SAS y SATA no puedo rechazar la hipótesis de que tengan rendimientos equivalentes

SAS-IDE  $\rightarrow p=0.00005 (<\alpha) \rightarrow$  Rechazo la hipótesis de que son equivalentes. Puedo irme a la media o diferencia de las medias y cogerlo. Sale negativo  $d_i \rightarrow$  SAS es mejor

SATA – IDE  $\rightarrow p=0.004 < \alpha \rightarrow$  Rechazo y por tanto no tienen el mismo rendimiento. Sale negativo  $d_i$  ( $T_{ide} > T_{sata}$ )  $\rightarrow$  SATA es mejor

Basta ver si está el 0 en el intervalo de confianza para  $\underline{d}_{real}$

- SAS - SATA  $\rightarrow$  Incluye el 0  $\rightarrow$  no puedo rechazar
- SAS – IDE  $\rightarrow$  no incluye el 0  $\rightarrow$  puedo rechazar
- SATA – IDE  $\rightarrow$  no incluye el 0  $\rightarrow$  puedo rechazar

\*anova con dos factores no entra\*

Concluimos que, al 95% de confianza, el disco IDE es claramente peor que los otros dos, pero que las diferencias entre SAS y SATA, para este problema, no son estadísticamente significativas, por lo que podríamos decidimos por el más barato (o hacer más pruebas para estar más seguros)