*UNIVERSIDAD DE PANAMÁ*
*CENTRO REGIONAL UNIVERSITARIO DE VERAGUAS*
*FACULTAD DE INFORMÁTICA, ELECTRÓNICA Y COMUNICACIÓN*

*MONOGRAFÍA:*

# *MÉTRICAS APLICABLES A LA MEDICIÓN DEL RENDIMIENTO DEL HARDWARE*

*PRESENTA:*

*RAÚL ENRIQUE DUTARI DUTARI*

*2009*

# *TABLA DE CONTENIDOS*

# 1. OBSERVACIONES PRELIMINARES.

El rendimiento del hardware es clave para la efectividad de un sistema completo hardware/software, pero cuantificarlo es una tarea difícil. Según que aplicación vaya a tener, se debe utilizar una métrica u otra.

Suele suceder que máquinas que se venden como de máximo rendimiento no dan la talla en las aplicaciones para las que se usan. La definición de rendimiento dependerá, por tanto, del sistema y de la aplicación.

Para comparar el rendimiento de dos estaciones de trabajo independientes manejadas por un único usuario serviría el tiempo de respuesta del computador, es decir el tiempo entre el comienzo y la finalización de la tarea (también llamado tiempo de ejecución).

Pero si se quisiera comparar el rendimiento de dos grandes máquinas multiusuarios de tiempo compartido, la medida del rendimiento vendría dada por el número de tareas acabadas en un día, aunque esto vaya en detrimento del tiempo de respuesta (productividad). Incluso dentro de un mismo caso no es lo mismo la visión del rendimiento que tendrá el administrador que la del usuario.

Esta monografía se enfoca en analizar las métricas que se pueden aplicar al momento de cuantificar el rendimiento de un sistema de hardware particular.

# 2. MEDIDAS DE RENDIMIENTO GLOBAL.

Se define rendimiento de un sistema como la capacidad que tiene dicho sistema para realizar un trabajo en un determinado tiempo. Es inversamente proporcional

al tiempo, es decir, cuanto mayor sea el tiempo que necesite, menor será el rendimiento.

Los computadores ejecutan las instrucciones que componen los programas, por lo tanto el rendimiento de un computador está relacionado con el tiempo que tarda en ejecutar los programas. De esto se deduce que el tiempo es la medida del rendimiento de un computador.

Intuitivamente se puede decir que el computador que realiza la misma cantidad de trabajo que otro en menos tiempo es el más rápido. El tiempo puede definirse de diferentes formas según lo que contemos. En consecuencia, se pueden establecer algunos criterios utilizando esta base.

## 2.1. TIEMPO DE RESPUESTA.

También llamado tiempo de reloj o tiempo transcurrido, se define como el tiempo total que necesita el sistema para completar una tarea incluyendo:

- Accesos al disco.
- Accesos a memoria.
- Actividades de entrada /salida.
- Gastos de sistema operativo.

El principal inconveniente de esta métrica es que los computadores actuales actúan en modo multitarea, o son de tiempo compartido en su casi gran mayoría, es decir un procesador trabaja en varios programas. En consecuencia, no se

puede asegurar que el tiempo de respuesta sea igual al tiempo gastado por el procesador en trabajar en un proceso.

Para solucionar este problema se distingue entre tiempo de respuesta y tiempo de ejecución de la CPU siendo este el tiempo que la CPU emplea en realizar una tarea. No incluye los tiempos de entrada/salida de información, ni el tiempo de ejecución de otros programas. Este tiempo se descompone a su vez en Tiempo de CPU de usuario y Tiempo CPU de sistema, aunque esta subdivisión no se suele considerar por la complejidad de las medidas que acarrea. Otras métricas habitualmente utilizadas son la duración de los ciclos y la frecuencia. Estas métricas se pueden relacionar mediante las siguientes expresiones:

$$T_{CPU} = \text{cantidad de ciclos de CPU para completar el programa} \times \text{duración del ciclo}$$

O expresada de otra manera:

$$T_{CPU} = \frac{\text{cantidad de ciclos de CPU para completar el programa}}{\text{frecuencia}}$$

Con base en las fórmulas anteriores, para mejorar el rendimiento de un sistema, se tendrá que reducir la duración del ciclo o reducir el número de ciclos necesitados.

Otra métrica utilizada es la de Ciclos de reloj por instrucción (CPI) que es el número medio de ciclos de reloj que necesita cada instrucción para ejecutarse. Proporciona un medio de comparar dos implementaciones diferentes de una misma arquitectura ya que el recuento de instrucciones es el mismo. Se suele obtener por simulación detallada de una implementación y se relaciona con el tiempo de CPU mediante la siguiente expresión.

$$T_{CPU} = \text{cantidad de intrucciones} \times CPI \times \text{duración del ciclo}$$

En resumen, el rendimiento de un procesador para un determinado programa queda en función de tres factores:

- **Frecuencia de la CPU:** la cual depende fundamentalmente de la tecnología de fabricación del procesador. Cuanto mayor sea la frecuencia de la CPU, mejor será el rendimiento.

- **Número de instrucciones del programa:** que depende del programador, del lenguaje de programación y del compilador. Cuanto mayor sea el número de instrucciones del programa peor rendimiento tendrá.

- **CPI:** que depende de diseño interno o arquitectura del computador y del software o instrucciones que se hayan elegido. Es importante optimizar el programa con instrucciones que tengan pocos ciclos. Cuanto mayor sea el CPI, peor será el rendimiento.

## 2.2. MIPS.

MIPS es la abreviatura de “Millones de Instrucciones por Segundo”. Esencialmente, tal como su nombre lo indica, la métrica se establece con base en la cantidad de instrucciones por segundo que es capaz de ejecutar un sistema particular.

Esta métrica se puede relacionar con las anteriores a través de las siguientes expresiones:

$$MIPS = \frac{\text{Millones de instrucciones}}{\text{Tiempo de ejecución}}$$

$$MIPS = \frac{\text{Millones de instrucciones}}{\text{Ciclos de CPU} \times \text{Tiempo de ciclos}}$$

$$MIPS = \frac{\text{Frecuencia}}{CPI \times 10^6}$$

Esta métrica establece una media de la frecuencia de ejecución de instrucciones para una máquina particular. Tiene la ventaja de ser fácil de comprender, ya que intuitivamente se ve que las máquinas más rápidas tienen mayores MIPS.

Sin embargo, su principal problema es que especifica la frecuencia de ejecución pero no depende del repertorio de instrucciones, es decir no se pueden comparar computadores con diferentes repertorios de instrucciones ya que difieren los recuentos de instrucciones, ya que:

- MIPS puede variar entre programas en el mismo computador

- Puede variar inversamente con el rendimiento. Puede ocurrir por ejemplo que el sistema no tenga implementada la multiplicación y tenga que implementarse por software. Entonces aunque se ejecuten muchas instrucciones el rendimiento será menor que el de un sistema que tenga un multiplicador en hardware.

Existen otras variantes para esta métrica, como son:

- **MIPS de pico o MIPS máximo:** que se obtiene cuando se realiza la prueba de rendimiento utilizando código que contiene una mezcla de instrucciones que difícilmente se va a dar en la realidad.

- **MIPS relativos:** que se obtienen cuando se compara la ejecución de una máquina cualquiera, al compararla contra el VAX 11/780.

## 2.3. MFLOP.

MFLOP es la abreviatura de “Millones de Operaciones en Punto Flotante por Segundo”. Esencialmente, tal como su nombre lo indica, la métrica se establece con base en la cantidad de instrucciones de punto flotante por segundo que es capaz de ejecutar un sistema particular.

Esta métrica depende del programa porque diferentes programas necesitan la ejecución de diferente número de operaciones en coma flotante. Esta medida se basa en las operaciones del programa y no en las instrucciones. Es mucho más útil para comparar diferentes máquinas que los MIPS porque el mismo programa ejecutándose en diferentes máquinas puede tener un número diferente de instrucciones, pero siempre tendrá el mismo número de operaciones en punto flotante.

Aunque esto último no es cierto porque el número de operaciones no es consistente entre máquinas y por lo tanto aunque el número supuesto de operaciones sea el mismo el número final de ellas no lo es. El ejemplo clásico de esta situación lo ofrece el supercomputador Cray 2 que no tiene operación de división mientras que el MOTOROLA 68882 – un humilde procesador para estaciones de trabajo Macintosh – sí la tiene, por lo tanto en el Cray 2 se necesitan varias operaciones en coma flotante para implementar la división, operación que realizará en una única instrucción el 68882.

Adicionalmente, existen otros problemas en esta métrica, ya que no todas las operaciones en punto flotante se ejecutan a la misma velocidad. Por ejemplo no es lo mismo que las operaciones de punto flotante sean sumas a que sean divisiones. Para solucionar este problema se suele realizar un recuento de instrucciones en coma flotante en un programa escrito en alto nivel y ponderar

las operaciones más complejas asignándolas un peso mayor. A este tipo de información se le llama MEGAFLOPS normalizados. De todos modos los resultados que se obtienen se alejan de la realidad.

# 3. APLICACIONES PARA EVALUAR EL RENDIMIENTO GLOBAL.

El rendimiento de un sistema de cómputo se mide analizando una serie de componentes físicos que determinan el rendimiento completo del sistema. A la hora de determinar el rendimiento global de un sistema, también hay que evaluar el sistema operativo, los equipos lógicos de red, los compiladores y las librerías gráficas, entre otros componente.

De allí que la mejor manera de evaluar el rendimiento de un sistema debería ser que el usuario que trabaja con “SUS” programas diariamente, los ejecutara en el nuevo computador para evaluar su rendimiento frente al computador anterior.

En la práctica, es muy es difícil que ocurra esta situación, por limitaciones de tiempo, disponibilidad de equipo y conocimiento técnico de los usuarios finales.

Por eso se utilizan los Benchmark, que son programas escogidos para medir el rendimiento de una maquina. Los programas de prueba forman una carga de trabajo en teoría capaz de medir el rendimiento de la carga real de trabajo. Hoy en día se sabe que los mejores benchmark son aplicaciones reales, como las que el usuario emplea regularmente o aplicaciones típicas.

La utilización de programas reales como banco de prueba hace difícil que el diseñador pueda utilizar trucos para acelerar el rendimiento. Ha sido costumbre muy generalizada, cuando se han utilizado programas de prueba específicos,

diseñar los compiladores para acelerar los segmentos de código del banco de prueba en que el esfuerzo computacional era mayor de este modo se obtienen unos resultados finales mejores de los reales, engañando literalmente a los usuarios.

Los bancos de pruebas aparecieron después de lo tiempos de CPU, los MIPS y los MFLOPS. No todos los bancos de prueba tenían las mismas características, y según éstas se podían clasificar.

## 3.1. BENCHMARKS SINTÉTICOS.

El objeto de este tipo de programas de prueba es simular el comportamiento de aplicaciones del mundo real. Para elaborar estas pruebas sintéticas se estudian una serie de aplicaciones y se desarrolla un código artificial que mezcla los cálculos matemáticos, bucles, llamadas a funciones, etc. Las series de programas de prueba sintéticos más conocidas son Whetstone y Dhrystone. Los Benchmark Sintéticos están formados por las rutinas más repetitivas de los programas más utilizados.

- **Dhrystone Benchmark (MIPS):** miden la eficiencia del procesador y del compilador en un entorno de desarrollo de sistemas con lenguajes de alto nivel. Su valor es expresado en instrucciones Dhrystone por segundo (Dhrystone MIPS, millones de instrucciones Dhrystone por segundo). No realiza operaciones en coma flotante, por lo que muchos fabricantes no lo consideran como una medida adecuada para definir el rendimiento de hoy en día. Los resultados se relativizan respecto al número de instrucciones Dhrystone por segundo que son alcanzadas en un VAX 11/780.

- **Whetstone Benchmark:** predecesora del Dhrystone, es una medida desarrollada para evaluar sistemas que se vayan a destinar a la ejecución de pequeños programas científicos y de ingeniería. Sus programas se han implementado en FORTRAN e incluyen cálculos con enteros y en coma flotante, manipulación de arrays y saltos condicionales. Esta prueba predice cómo serán ejecutadas aplicaciones que hacen un uso intensivo de la unidad central de proceso. Los resultados son expresados en Kwips (miles de instrucciones Whetstone por segundo.

Estos dos tipos de Benchmarks, a partir de los 80, cayeron en desuso y aparecieron los Benchmark reducidos.

## 3.2. BENCHMARKS REDUCIDOS O DE JUGUETE.

Los programas reducidos tienen entre 10 y 100 líneas de código y producen un resultado que el usuario conoce antes de ejecutarlo. Algunos ejemplos de este tipo de Benchmarks serían:

- **El Towers:** que resuelve el problema de las torres de Hanoi con muchas llamadas recursivas.

- **El Perm:** que calcula permutaciones de 7 tornadas de 5 en 5.

- **Criba de Eratóstenes.**

- **Puzzle.**

- **Quicksort**, entre otros.

Todos son pequeños, fáciles de introducir y de ejecutar en cualquier computador. Su principal ventaja es su facilidad de compilación y de ejecución por simuladores, permitiendo el cálculo del rendimiento aproximado en las fases de diseño del sistema.

Sin embargo, al ser tan pequeños y sencillos, eran muy vulnerables, ya que era muy fácil mejorar el rendimiento para un programa concreto, por lo que se pasó a los Benchmark Kernel o de núcleo.

## 3.3. KERNEL BENCHMARKS.

Son programas de pruebas formados por pequeñas piezas clave de programas reales que evalúan el rendimiento y lo aíslan de las características individuales de una máquina, permitiendo explicar las razones de las diferencias en los rendimientos de programas reales.

Los ejemplos más conocidos son el “Livermore Loops”, una serie de 21 fragmentos de bucles pequeños, y el “Linpack”, formado por un paquete de subrutinas de álgebra lineal. Sólo tratan algunos aspectos y son antiguos. No existen núcleos para evaluar prestaciones gráficas.

A continuación, se analizará parte de la historia de uno de los programas de prueba más famosos dentro de la industria informática, que ilustra las situaciones antes descritas.

## 3.4. SPEC (SYSTEM PERFORMANCE EVALUATION COOPERATIVE).

En 1988 surge la compañía SPEC formada por un consorcio de fabricantes de hardware, entre los que destacan: IBM, DEC, INTEL Hewlett, y otros más. Su objetivo era desarrollar programas de prueba normalizados para evaluar la potencia de los computadores. Para ello lo que hacía era escoger un conjunto real de programas y entradas. Esta tarea se vio facilitada por la portabilidad de los SO y la popularidad de los LAN. Inicialmente fueron 6 bancos de prueba en coma flotante y 4 de enteros.

El uso de bancos de pruebas dio lugar a la aparición de acciones no éticas por parte de los mismos fabricantes de hardware. Generalmente el rendimiento final dependía de pequeños segmentos de código que se repetían intensivamente. Para obtener mejores rendimientos los fabricantes introducían pequeñas modificaciones, o bien en la arquitectura del sistema, o en los compiladores, de manera que falseaba lo resultados finales, presentando al sistema con un rendimiento superior al real.

Este era el caso del programa matrix300, que formaba parte del banco de pruebas SPEC88. Este programa que multiplicaba matrices, estaba pensado para medir el acceso a la memoria de un sistema. Existían compiladores que reducían al mínimo los accesos a memoria consiguiendo rendimientos hasta 5 veces mayores. Este programa fue eliminado del banco de pruebas SPEC90 que estaba compuesto por SPECint, y SPECfp

La prueba SPECint estaba compuesta por una batería de pruebas sobre valores enteros, que involucraba, esencialmente:

- Cálculos con enteros.
- Minimizar funciones lógicas.
- Traducir ecuaciones booleanas a tablas de verdad.
- Cálculo de valores en una hoja de cálculo.

En tanto, SPECfp estaba compuesta por otra batería de pruebas que se refería, fundamentalmente, a:

- Cálculos en coma flotante.
- Simulaciones de circuitos analógicos.
- Cálculo de integrales derivativas.
- Resolución de redes neuronales.

## 3.5. OTROS BENCHMARKS.

Entre ellos, destacan:

- **AIM Suite III:** es un Benchmark que mide la eficiencia de sistemas multiusuario, en entornos servidores, UNIX, para atender varios usuarios ejecutando cada uno de ellos un proceso diferente.
- **Xmark:** es un Benchmark de dominio público basado en UNIX y disponible bajo el sistema Xwindow, para evaluar el rendimiento de aplicaciones propias de entornos gráficos y de autoedición.

- **Graphstone:** es una prueba gráfica que evalúa el rendimiento del subsistema de vídeo. El resultado se expresa en número de operaciones ejecutadas por segundo.

- **Khornerstone:** es una medida que prueba tanto el subsistema gráfico, como el rendimiento de la UCP y del disco duro, así como la capacidad para la ejecución de operaciones en coma flotante.

- **SYSmark:** se trata de un benchmark para evaluar CPU's bajo el sistema Windows. Hay diferentes versiones como la SYSmark98 y la SYSmark2002.

- **CPUmark99:** mide el rendimiento de los Pentium bajo Windows.

- **Wintune98:** dedicado a procesamientos de texto y hojas de cálculo.

- **Winstone99, Multimediummark99, J.Mark 2.0**, y otros más.

## 4. RESUMEN DE LAS MEDIDAS DEL RENDIMIENTO DE LOS SISTEMAS, PARA EL USUARIO FINAL.

Una vez que se ha decidido cuales son los programas de prueba a utilizar y se han ejecutado para determinar los respectivos rendimientos, hay que decidir como proporcionar la información al usuario final. Si bien es cierto que un cuadro de rendimientos da mayor información, el usuario suele preferir una sola cifra que le ayude a comparar dos máquinas.

## 4.1. RAZONES Y PROPORCIONES.

Una de las formas más simples de realizar esta comparación de una cifra es utilizando las razones y proporciones de la aritmética básica.

Si se tiene un computador $A$ que tiene un rendimiento $R_A$, bajo ciertas condiciones; y se desea comparar contra un computador $B$, que bajo las mismas condiciones posee un rendimiento $R_B$, entonces, el rendimiento del computador $A$ frente al computador $B$, denotado como $R_{A\mapsto B}$ estará dado por la expresión:

$$R_{A\mapsto B} = \frac{R_B}{R_A}$$

Por ejemplo, se supone que se tienen dos máquinas $A$ y $B$ y dos programas el programa 1 y el 2, cuyos respectivos tiempos de ejecución aparecen en la tabla que se muestra a continuación:

| | ***Computador A (s)*** | ***Computador B (s)*** |
|---|---|---|
| ***Programa 1*** | 1 | 10 |
| ***Programa 2*** | 1000 | 100 |
| ***Tiempo total*** | 1001 | 110 |

**Tabla 1: Comparación de Rendimiento vs. Programa por Razones y Proporciones.**

En consecuencia, el computador $B$ tendrá un rendimiento relativo de:

$$R_{B\mapsto A} = \frac{R_A}{R_B} = \frac{1001s}{110s} = 9.1$$

Es decir, el computador $B$ será 9.1 veces más rápido que el computador $A$. Esta es una medida que cualquier usuario medianamente educado debería estar en la capacidad intelectual de comprender su significado correcto.

## 4.2. MEDIA ARITMÉTICA PONDERADA.

Otra herramienta matemática que puede facilitar las comparaciones entre equipos es el uso de la media aritmética ponderada (MAP) universalmente conocida. Obviamente, a menor media aritmética mejor rendimiento. Esta medida tomará en cuenta que algunos programas se usan más que otros. Está dada por la expresión:

$$MAP = \frac{\sum_{i=1}^{n} (PC)_i \times (TE)_i}{\sum_{i=1}^{n} (PC)_i} = \frac{(PC)_1 \times (TE)_1 + (PC)_2 \times (TE)_2 + ... + (PC)_n \times (TE)_n}{(PC)_1 + (PC)_2 + ... + (PC)_n}$$

Donde:

- $i, n$ representan, respectivamente, a un programa dado y la cantidad de programas que se consideran.
- $(TE)_i$ representa el tiempo de ejecución de un programa dado.
- $(PC)_i$ representa el porcentaje de carga sobre el total del tiempo de ejecución, para un programa dado.

Por ejemplo, se supone que nuevamente se tienen dos máquinas $A$ y $B$ y dos programas el programa 1 y el 2, con tiempos de ejecución dados, además de las cargas de trabajo dadas, entonces su tiempo medio ponderado se puede

calcular con base en la expresión antes enunciada, tal como se presenta en la tabla que se muestra a continuación:

| | *Computador A* | | *Computador B* | |
|---|---|---|---|---|
| | ***T. de Ejecución (s)*** | ***% Carga*** | ***T. de Ejecución (s)*** | ***% Carga*** |
| ***Programa 1*** | 1 | 95% | 10 | 25% |
| ***Programa 2*** | 1000 | 5% | 100 | 75% |
| ***Tiempo Medio Ponderado (s)*** | 50.95 | | 77.50 | |

**Tabla 2: Comparación de Rendimiento vs. Programa por Media Aritmética Ponderada.**

De la tabla se desprende que la importancia que tiene el que las instrucciones que se ejecutan frecuentemente tengan un tiempo de ejecución mínimo, y que la incidencia de algunas instrucciones muy lentas, no es importante, si ellas se ejecutan esporádicamente.

## 4.3. MEDIA GEOMÉTRICA PONDERADA.

Las medias aritméticas no son recomendadas cuando se requieren comparaciones que sean verdaderamente certeras en todos los casos. Para estas situaciones, se recomienda el empleo de medias geométricas (MG), ya que una de sus propiedades establece que:

$$\frac{MG(Y_i)}{MG(X_i)} = MG\left(\frac{Y_i}{X_i}\right)$$

Lo que para los efectos de las medidas de rendimiento, ofrece una representación normalizada, independiente del computador donde se realicen las mediciones.

Al igual que en el caso de la MAP, la media geométrica ponderada (MGP) se evalúa en términos de que; a menor media geométrica mejor rendimiento. Nuevamente, se tomará en cuenta que algunos programas se usan más que otros. La expresión que representa a la MGP, está dada por:

$$MGP = \left( \prod_{i=1}^{n} (TE)_i^{(PC)_i} \right)^{\frac{1}{\sum_{i=1}^{n}(PC)_i}} = \left( (TE)_1^{(PC)_1} \times (TE)_2^{(PC)_2} \times \ldots \times (TE)_n^{(PC)_n} \right)^{\frac{1}{(PC)_1+(PC)_2+\ldots+(PC)_n}}$$

Donde nuevamente:

- $i, n$ representan, respectivamente, a un programa dado y la cantidad de programas que se consideran.

- $(TE)_i$ representa el tiempo de ejecución de un programa dado.

- $(PC)_i$ representa el porcentaje de carga sobre el total del tiempo de ejecución, para un programa dado.

Para efectos de comparación, se considera nuevamente, la tabla del ejemplo anterior, ahora calculado utilizando la media geométrica ponderada, los resultados obtenidos se presentan a continuación:

| | *Computador A* | | *Computador B* | |
|---|---|---|---|---|
| | ***T. de Ejecución (s)*** | ***% Carga*** | ***T. de Ejecución (s)*** | ***% Carga*** |
| ***Programa 1*** | 1 | 95% | 10 | 25% |
| ***Programa 2*** | 1000 | 5% | 100 | 75% |
| ***Tiempo Medio Ponderado Geométricamente (s)*** | 1.41 | | 56.23 | |
| ***Tiempo Medio Ponderado Aritméticamente (s)*** | 50.95 | | 77.50 | |

**Tabla 3: Comparación de Rendimiento vs. Programa por Media Geométrica Ponderada.**

En la última tabla, se puede observar, claramente, que la media geométrica ponderada ofrece una medida del rendimiento del computador más cónsona con la realidad que la media aritmética. Además, se recalca nuevamente la importancia que tiene el que las instrucciones que se ejecutan frecuentemente tengan un tiempo de ejecución mínimo, y que la incidencia de algunas instrucciones muy lentas, no es importante, si ellas se ejecutan esporádicamente.

## 4.4. LEYES DE AMDAHL.

El posible aumento de rendimiento de un sistema, para una mejora particular, está determinado y limitado por el uso que se de a la mejora en particular.

Por ejemplo, de nada sirve mejorar en un 100% el rendimiento del multiplicador en coma flotante si luego solo se le utiliza el 0.001% de las instrucciones. Es más rentable y eficiente, introducir pequeñas mejoras en elementos que se usan mucho, que introducir grandes mejoras en elementos que se usan poco. Se puede resumir en el enunciado:

***CONVERTIR EN RÁPIDO AL CASO COMÚN.***

Además, conviene recordar que generalmente el caso común suele ser más simple y por lo tanto es más fácil de mejorar.

Precisamente, esta es la filosofía detrás de las leyes de Amdahl, que evalúan las modificaciones en el rendimiento de un computador cuando se introducen mejoras o más recursos. El criterio fundamental que se aplica, consiste, precisamente, en que lo que hay que mejorar o modificar siempre es lo que se usa más frecuentemente, ya que es lo que más afecta al rendimiento. En el estudio de las leyes de Amdahl se utilizan dos expresiones:

| | |
|---|---|
| $A_R = \frac{R_{Sn}}{R_{Sa}}$ | $G_V = \frac{T_{Sn}}{T_{Sa}}$ |

La primera de las expresiones establece una proporción entre el rendimiento de un sistema nuevo $R_{Sn}$ y el rendimiento de un sistema anterior $R_{Sa}$, que se denominará como aumento en el rendimiento $A_R$.

La segunda expresión establece otra proporción, ahora sobre el tiempo que requieren dos sistemas en completar una tarea determinada. La ganancia de velocidad $G_V$ entre dos sistemas, es la razón del tiempo que emplea un sistema nuevo $T_{Sn}$ entre el tiempo que emplea un sistema anterior $T_{Sa}$.

### 4.4.1. PRIMERA LEY DE AMDAHL.

La primera ley de Amdahl establece que el aumento del rendimiento debido a la inclusión de una mejora con un nuevo recurso en el sistema está limitado por el tiempo que se utiliza dicha mejora en la ejecución de la tarea. Su expresión está dada por:

$$T_{nuevo} = T_{antiguo} \times \left( \frac{\text{Fracción de tiempo mejorada}}{\text{G}_\text{V}} + \text{Fracción de tiempo sin mejora} \right)$$

### 4.4.2. SEGUNDA LEY DE AMDAHL.

La segunda ley de Amdahl establece que al introducir una mejora a un computador previamente mejorado, el incremento del rendimiento es menor que si se introduce la mejora sobre el sistema sin mejorar. Dicho de otra forma, la mejora incremental en la aceleración conseguida con la mejora de una parte se va reduciendo a medida que se van introduciendo nuevas mejoras. Su expresión está dada por:

$$A_{R_i} < A_{R_{i+1}} < \ldots$$

Donde $i$ denota al número de mejora que se realiza sobre el sistema.

Es importante aclarar, que la comparación de sistemas a la que hace alusión la segunda ley de Amdahl, es válida en general, cuando se considera la evolución de sistema en particular. Cuando se realizan cambios importantes en la arquitectura del sistema, dicha ley no se cumple.

Así, el ejemplo más emblemático, aunque no preciso, se puede observar en los procesadores Intel 80X86, incluyendo a dos modelos pre 80X86, para efectos de ampliar la comparación. La tabla que se presenta a continuación muestra a los procesadores, con su frecuencia base – considerando que el rendimiento del procesador es proporcional a la frecuencia de funcionamiento del mismo -, así como la mejora que representa el avance de un modelo a otro, bajo la ley de Amdahl.

| *Procesador* | *Frecuencia Máxima (MHz)* | $A_R$ |
|---|---|---|
| 8080 | 2 | |
| 8085 | 5 | 250.00% |
| 8088 | 10 | 200.00% |
| 80286 | 25 | 250.00% |
| 80386DX | 40 | 160.00% |
| 80486DX4 | 100 | 250.00% |
| Pentium | 200 | 200.00% |
| Pentium Pro | 200 | 100.00% |
| Pentium II | 450 | 225.00% |
| Pentium III | 1400 | 311.11% |
| Pentium IV | 3800 | 271.43% |

**Tabla 4: Comparación del rendimiento de los procesadores x86, con base a su frecuencia de funcionamiento, bajo la segunda ley de Amdahl.**

La tabla debe ser interpretada de la siguiente forma. Cada vez que se presenta un cambio de color en los renglones, significa que ocurrió un cambio drástico en el diseño del procesador. Dentro de un mismo color, sólo se presentan mejoras incrementales al diseño del procesador, y se observa el correspondiente descenso en el nivel de rendimiento del procesador. De acuerdo a lo planteado, se observa que se cumple la ley de Amdahl.

# 5. CONCLUSIONES.

- Cuantificar el rendimiento del hardware que compone a un computador, no es una tarea trivial.

- Hay parámetros bien claros que condicionan el rendimiento de un sistema, tales como: el tiempo de respuesta, la frecuencia de funcionamiento de la CPU, el número de instrucciones de los programas, , así como los ciclos de reloj por instrucción.

- Las métricas MIPLS y MFLOP son, en principio, una aproximación hacia la normalización de la medición del rendimiento de un sistema, basada en hardware. Sin embargo, son muy vulnerables a los cambios en la arquitectura de los procesadores, por lo que no se recomiendan cuando se requieran comparaciones importantes.

- Los bancos de pruebas ofrecen una solución al problema, basada en software, agrupadas como: programas de prueba sintéticos, núcleos y programas juguetes.

- A nivel del usuario final, el cálculo de razones o proporciones, combinado con la aplicación de medias aritméticas y geométricas ponderadas, se revelan como un mecanismo razonable para comparar el desempeño de los sistemas de cómputo.

- Las leyes de Amdahl se presentan como la sistematización del cálculo del rendimiento de los computadores.

## 6. REFERENCIAS BIBLIOGRÁFICAS.

**[BILBAO]** **BILBAO EGUIA, Josu.** *Arquitectura De Computadores: Las Soluciones Intel: 80086 – 80186 – 80286 – 80386 – 80486; Pentium I – Pro – II – III – IV; Itanium.* Sin Editorial. Sin Año De Publicación.

**[HEPA93]** **HENNESSY, John; PATTERSON, David.** *Arquitectura de Computadoras, Un enfoque Cuantitativo.* Primera Edición, McGraw-Hill, España, 1993.

**[LANC00]** **LANCHARES DÁVILA, Juan.** *Apuntes De Estructura De Computadores.* Departamento De Arquitectura De Computadores Y Automática. Universidad Complutense De Madrid. 2000.

**[MUEL01]** **MUELLER, Scott.** *Manual de actualización y reparación de PC's.* Doceava edición. Pearson, México, 2001.

**[STAL06]** **STALLINGS, William.** *Organización y Arquitectura de Computadoras. Principios de Estructura y de Funcionamiento.* Séptima Edición, Pearson Prentice-Hall, España, 2006.

**[TANE92]** **TANENBAUM, Andrew S.** *Organización De Computadoras: Un Enfoque Estructurado.* Tercera Edición, Prentice-Hall, México, 1992.