UNIVERSIDAD DE GUADALAJARA

CUCEI

CARRERA: **INGENERIA EN COMPUTACION**

INVESTIGACION 3: **RENDIMIENTO DE UN PROCESADOR**

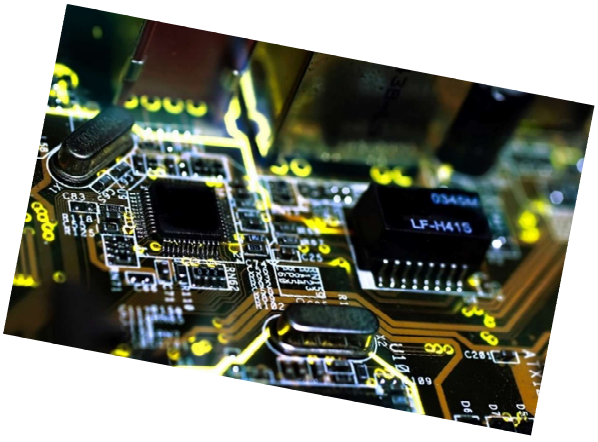
ALUMNO: **EFRAIN ROBLES PULIDO**

CODIGO: **221350095**

NOMBRE DE LA MATERIA: **ARQUITECTURA DE COMPUTADORAS**

SECCIÓN: **D10** *CALENDARIO:* **2021B**

NOMBRE DE LA PROFESORA: **THELMA ISABEL MORALES RAMIREZ**



Rendimiento de un procesador

❖ Medidas del rendimiento de un computador

Para comparar diferentes procesadores es necesario conocer la *unidad de medida* y el *patrón de medida a utilizar*. El primero se refiere a la métrica utilizada para cuantificar la comparación, y el otro al objetivo mismo de la comparación.

El **tiempo** es la *unidad de medida* por excelencia, aunque no siempre coincidan los puntos de vista de los diferentes observadores. Así, que estaremos interesados en el tiempo de respuesta (response time, tiempo que ejecuta un programa) del procesador mientras que otro estará en la productividad (throughput, cantidad de trabajos que hace). Pero en ambos la clave es el tiempo, el que realiza la misma cantidad de trabajo en el menor tiempo posible será el más rápido, la diferencia será en que si medimos una tarea (tiempo de respuesta) o muchas (productividad).

El *patrón de medida mas significativo* es **el conjunto de programas reales que se ejecutan** en los procesadores. Surge de nuevo y con más intensidad la diversidad de puntos de vista. Fijar de la forma más objetiva posible los patrones o programas respecto a los cuales se mida el rendimiento de un procesador será pues una tarea polémica y siempre cuestionada por la comunidad de interesados en los resultados de la medida.

A continuación, se muestran algunas medidas de rendimiento, clasificadas según el nivel de abstracción y la característica que queramos mejorar:

- **Software de aplicación**

Niveles de abstracción	Medidas de rendimiento
Generadores de programas.	• Número de Instrucciones de alto nivel. • Complejidad del algoritmo (Ω).
Programas de aplicación	• Tiempo de ejecución (Tej). • Tiempo de respuesta de usuario. • Tiempo de CPU (T_{CPU}). • Operaciones por segundo. • Rendimiento (*performance*). • Productividad o *throughput*. • Tiempo de respuesta.

- **Software de sistema**

Niveles de abstracción	Medidas de rendimiento
Compiladores e intérpretes.	• Número de instrucciones máquina por programa (N).
Sistemas operativos	• Número de programas en paralelo. • Número de *threads* en paralelo.

- **Interfaz software (SW)/hardware (HW)**

Niveles de abstracción	Medidas de rendimiento
Arquitectura del conjunto de instrucciones.	• Ciclos por instrucción (CPI). • Instrucciones por segundo (MIPS). • Instrucciones de coma flotante por segundo (MFLOPS).
Organización de componentes hardware	• Tiempo de acceso a memoria (Tacc, TAM). • Latencia. • Ancho de banda (megabytes/segundo). • Transacciones Por Segundo (TPS).

- **Hardware**

Niveles de abstracción	Medidas de rendimiento
Nivel digital.	• Frecuencia de reloj (F). • Periodo o ciclo de reloj (t, T). • Inferencias lógicas por segundo (KLIPS).
Nivel físico.	• Número de transistores por componente.

Tiempo de ejecución

El tiempo que tarda un programa en ser ejecutado por un computador puede ser difícil de medir, pero es la más fiables para medir el rendimiento, debido a los Sistemas Operativos Multitarea y a los dispositivos de E/S, que tienen tiempos de respuesta que son independientes de la frecuencia de reloj del ordenador. El tiempo de ejecución de un programa lo dividiremos en las siguientes componentes:

- **Tiempo de respuesta**: Es el tiempo necesario para completar una tarea, incluyendo los accesos al disco, a la memoria, las actividades de E/S y los gastos del S.O. Es el tiempo que percibe el usuario.

- **Tiempo de CPU**: Es el tiempo que tarda en ejecutarse un programa, sin tener en cuenta el tiempo de espera debido a la E/S o el tiempo utilizado para ejecutar otros programas. Se divide en:

  - ✓ *Tiempo de CPU* utilizado por el usuario*:* Es el tiempo que la CPU utiliza para ejecutar el programa del usuario. No se tiene en cuenta el tiempo de espera debido a la E/S o el tiempo utilizado para ejecutar otros programas

  - ✓ *Tiempo de CPU* utilizado por el S.O.: Es el tiempo que el S.O. emplea para realizar su gestión interna.

El *tiempo de respuesta* se utiliza como medida del rendimiento del sistema (con el sistema no cargado), mientras que el rendimiento de la CPU normalmente hace referencia al tiempo de CPU del usuario sobre un sistema no cargado.

Otros parámetros de rendimiento

- **MIPS** (Millones de Instrucciones Por Segundo)

1. Los MIPS dependen del repertorio de instrucciones, por lo que resulta un parámetro difícil de utilizar para comparar máquinas con diferente repertorio
2. Varían entre programas ejecutados en el mismo computador.
3. Pueden variar inversamente al rendimiento, como ocurre en máquinas con hardware especial para punto flotante, que emplean menor tiempo que las que no disponen de este hardware y por tanto tienen mejor rendimiento. Sin embargo, presentan un menor valor de los MIPS porque ejecutan menor número de instrucciones.

Pero, los MIPS pueden fallar al dar una medida del rendimiento, puesto que no reflejan el tiempo de ejecución. Por esta razón se han venido utilizando los MIPS relativos, que miden cuantas veces más tarda en ejecutarse un programa en la máquina a medir que en una de referencia.

- **MFLOPS** (Millones de operaciones en punto Flotante Por Segundo)

Existen operaciones en coma flotante rápidas (como la suma) o lentas (como la división), por lo que puede resultar una medida poco significativa. Por ello se han utilizado los MFLOPS normalizados, que dan distinto peso a las diferentes operaciones.

- **Productividad (throughput)**

Sería el número de tareas ejecutadas en la unidad de tiempo.

❖ Patrones de medida (Benchmarks)

El procedimiento consiste en ejecutar un conjunto de **programas de prueba** llamados benchmarks. Se pone a prueba el equipo con tareas muy exigentes y variadas, con la intención de medir cómo de bien las cumple. De esta forma, se puede estimar bajo qué tareas o estímulos un determinado smartphone, tablet u ordenador se comporte de una manera fiable y efectiva.

En la actualidad, en el benchmark estándar del mercado SPEC CPU2006, así como en su versión anterior CPU2000, se usa como máquina de referencia un servidor diseñado en 1997, llamado Sun Ultra Enterprise 2, con un procesador UltraSPARC II a 296 MHz

Entonces para evaluar el rendimiento de un computador podemos utilizar diferentes técnicas:

- Modelos analíticos (matemáticos) de la máquina
- Modelos de simulación (algorítmicos) de la máquina
- La máquina real

Las dos primeras técnicas se utilizan cuando la máquina no está disponible. Los modelos analíticos tienen limitado su ámbito de utilización por la dificultad de expresar en forma de ecuaciones matemáticas el comportamiento detallado de la máquina y su carga de trabajo. Se utilizan en fases muy tempranas de diseño para realizar estimaciones generales del rendimiento.

En la tercera técnica es la máquina o máquinas reales que se evaluaran con el fin de disponer de algún criterio de elección. En este caso será necesario disponer de un conjunto de programas representativos de la carga real de trabajo que vaya a tener la máquina, y con respecto a los cuales se realicen las medidas.

Se clasifican las técnicas por:

1. **Su ámbito de aplicación**, el tipo de recursos computacionales que mayoritariamente intervienen en la evaluación. En este sentido los clasificaremos en:
    - *Enteros*: aplicaciones en las que domina la aritmética entera, incluyendo procedimientos de búsqueda, operaciones lógicas, etc. Por ejemplo, SPECint2000.
    - *Punto flotante*: aplicaciones intensivas en cálculo numérico con reales. Por ejemplo, SPECfp2000 y LINPACK.

- *Transacciones*: aplicaciones en las que dominan las transacciones on-line y off-line sobre bases de datos. Por ejemplo, TPC-C.

2. **La naturaleza del programa que implementan**:

- Programas reales: Compiladores, procesadores de texto, etc. Permiten diferentes opciones de ejecución. Con ellos se obtienen las medidas más precisas

- Núcleos (Kernels): Trozos de programas reales. Adecuados para analizar rendimientos específicos de las características de una determinada máquina: Linpack, Livermore Loops

- Patrones conjuntos (benchmarks suits) Conjunto de programas que miden los diferentes modos de funcionamiento de una máquina: SPEC y TPC.

- Patrones reducidos (toy benchmarks): Programas reducidos (10-100 líneas de código) y de resultado conocido. Son fáciles de introducir y ejecutar en cualquier máquina (Quicksort, …)

- Patrones sintéticos (synthetic benchmarks): Código artificial no perteneciente a ningún programa de usuario y que se utiliza para determinar perfiles de ejecución. (Whetstone, Dhrystone)

❖ Influencia en el rendimiento de las alternativas de diseño

En muchos casos las alternativas se contemplan junto con datos reales de su presencia en el código máquina que generan los compiladores para programas reales. El resultado será la caracterización de un repertorio (ISA) que definirá las propiedades generales de los procesadores de tipo RISC.

Tipo de elementos de memoria en la CPU

Tres alternativas:

Tipo de máquina	Ventajas	Inconvenientes
Pila		
Acumulador	• Instrucciones cortas	• Elevado tráfico con Memoria

Registros	• Mayor flexibilidad para los compiladores • Más velocidad (sin acceso a Memoria)	• Instrucciones más largas

El tráfico con memoria es uno de los cuellos de botella más importantes para el funcionamiento del procesador. Se disminuye este tráfico con instrucciones que operen sobre registros, ya que el acceso es más rápido y las referencias a los registros se codifican con menor número de bits (instrucciones más cortas).

Referencias a memoria en instrucciones ALU

Tres alternativas:

Tipo de máquina	Ventajas	Inconvenientes
Registro-Registro	• Ninguna referencia a Memoria • Codificación fija => formato simple • Generación de código simple	• Mayor número de instruccionespor programa
Registro-Memoria	• Menor número de instrucciones	• Mayor tráfico con Memoria • Formato más complejo
Memoria-Memoria	• Muchos tipos de direccionamiento • Número mínimo de instrucciones porprograma	• Mucho acceso a memoria • Formato complejo

Dentro de las máquinas con registros de propósito general, las que operan en las instrucciones ALU de registro - registro (RR) son las que optimizan el uso de registros, quedando el acceso a memoria limitado a las instrucciones de carga y almacenamiento.

Orden de ubicación de los datos

Dos alternativas:

big-endian	Selección basada en motivos de compatibilidad
litle-endian	

Es indiferente desde el punto de vista del rendimiento. Pero serán motivos de compatibilidad con otros procesadores los que determinen una elección.

Alineamiento de datos

Dos alternativas:

acceso alineado	• Más rápido
acceso no alineado	• Más lento en general • Mayor flexibilidad

Son los datos alineados, o si el procesador permite a los no alineados, será el compilador quien genere siempre datos alineados.

Direccionamientos

Pueden reducir significativamente el número de instrucciones de un programa. Sin embargo, añaden complejidad al repertorio, aumentando con ello el CPI (número medio de ciclos por instrucción).

Una máquina eficiente, que favorezca los casos frecuentes debería soportar:

- Direccionamientos registro + desplazamiento y el inmediato
- Tamaños de los desplazamientos de 12 a 16 bits
- Tamaño del dato inmediato de 8 a 16 bits
- La supresión de los modos complejos no afecta decididamente al rendimiento

Datos operando

Cuando se les aplica los benchmarks se obtiene accesos a datos de longitud palabra o doble palabra dominan sobre los demás. Si se añaden la necesidad de

acceder al elemento mínimo que define la resolución del direccionamiento, así como el soporte del tipo carácter, es decir, el byte; y la existencia en la mayoría de los procesadores de operaciones hardware en punto flotante, se tiene que los:

- enteros de 16 y 32 bits
- flotantes de 64 bits
- caracteres de 8 bits

Operaciones

Las operaciones más simples son las que más se repiten en los programas, en especial las operaciones de carga, salto condicional, comparación, almacenamiento, suma, and, resta, transferencia entre registros y saltos-retornos de subrutina, que se llevan el 96% de las ocurrencias.

Entonces el repertorio ISA de un procesador eficiente no deberá incluir muchas más operaciones que las ya mencionadas anteriormente.

Sentencias de salto condicional

Es importante diseñar de forma eficiente el mecanismo de generación de condiciones y salto sobre el valor. Si analizamos la distribución de los saltos condicionales se ha obtenido que más de un 70% son saltos sobre igual o diferente, y un gran número son comparaciones con cero. Haciendo que algunos procesadores incorporen un registro cuyo contenido es siempre igual a cero.

Lo recomendable seria agregar instrucciones que integren el test sobre la condición y el correspondiente salto. También es necesario tener un registro cuyo contenido es inalterable igual a cero y también se podría hacer un desplazamiento de 8 bits para tener un mejor rendimiento.

Llamadas a procedimientos (subrutinas)

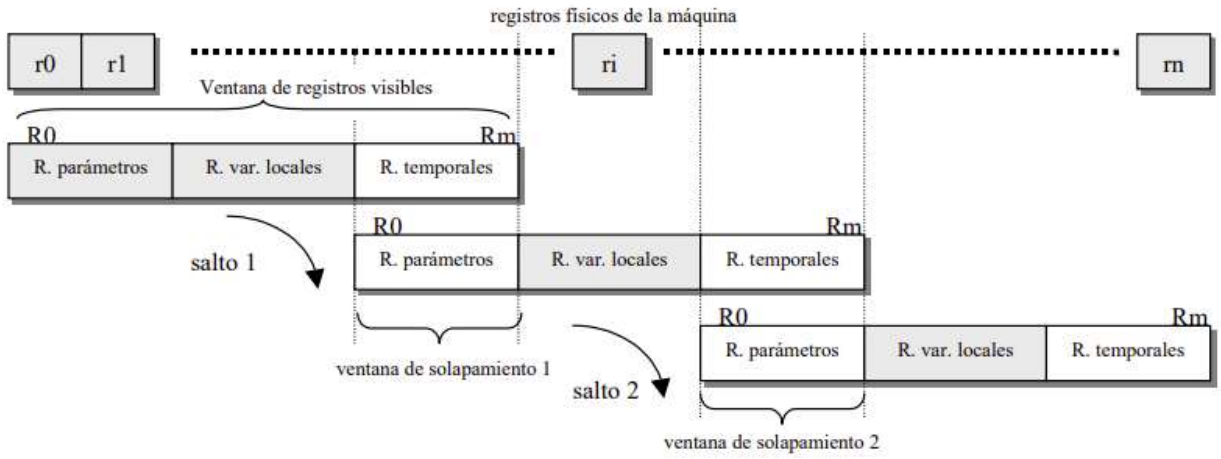
Aunque las instrucciones de llamada a procedimientos no son tan frecuentes como las de bifurcación condicional, su consumo de tiempo en los LAN es elevado debido a la gestión de los registros de activación (RA) que deben de realizar. Siendo conveniente optimizar la

ejecución de estas instrucciones. También se deberá tomar en cuenta que la variación del nivel de anidamiento en la que se mueve un programa se mantiene relativamente pequeña a lo largo de una secuencia de llamadas/retorno.

Este comportamiento de los programas en lo relativo a las llamadas a procedimientos se ha explotado en algunos procesadores (por ejemplo, SPARC) utilizando una ventana de registros para soportar los entornos, marcos o registros de activación (RA).

Registros de propósito general: ventanas para soportar llamadas a procedimientos

El procesador dispone de *n* registros físicos de los que en cada momento sólo son visibles para el programador (ventana) *m* de ellos. Cuando no se ha realizado ningún salto a subrutina, los registros visibles coinciden con los m primeros registros físicos. Cuando se ejecuta una instrucción de salto, los registros visibles se desplazan sobre los físicos de tal manera que se da un solapamiento entre los últimos antes del salto y los primeros después del salto. Es en los registros de solapamiento donde se realiza el paso de parámetros a la rutina y la devolución de resultados.



❖ Influencia de los compiladores de lenguajes de alto nivel

Al ejecutar un programa dependen significativamente del compilador que se utilice. Cuando se diseña el repertorio de instrucciones se tiene en mente en todo momento la tecnología de diseño de compiladores. Las decisiones arquitectónicas afectan directamente a la complejidad de diseño de un buen compilador.

Un compilador consta de una serie de etapas, las cuales transforman representaciones abstractas de alto nivel, en representaciones de más bajo nivel, hasta llegar al repertorio de instrucciones. Un compilador busca en primer lugar la exactitud ***de la traducción*** (mantenimiento de la semántica del programa en la transformación), en segundo lugar la ***velocidad del código*** generado, en tercer lugar el ***tamaño del código***, en cuarto lugar la ***velocidad del compilador***, y en quinto lugar el ***soporte a la depuración***. Desde el punto de vista del rendimiento que tenemos, la velocidad del código generado es el factor por optimizar. Las optimizaciones realizadas por los compiladores modernos son las siguientes:

- **Optimización de alto nivel,** realizadas en el código fuente**:**
  - Integración de procedimientos: sustituye la llamada a un procedimiento por el cuerpo de éste (expansión de macros).
- **Optimizaciones locales,** afectan a fragmentos de código lineal (sin bifurcaciones)**:**
  - Eliminación de subexpresiones comunes.
  - Propagación de constantes.
  - Reducción del tamaño de la pila en la evaluación de expresiones reorganizando su estructura.
- **Optimizaciones globales,** afectan al programa completamente que son más complejas de implementar**:**
  - Optimización de bucles.
  - Eliminación de subexpresiones comunes de alcance global (incluyendo saltos).
  - Propagación de copias: sustituye todas las instancias de una variable asignada.
- **Optimización del uso de registros,** es una de las que reporta mayor incremento de rendimiento.

Es responsabilidad del compilador cuya misión es mantener en registros (y no en memoria) los operandos necesarios para el mayor número posible de cálculos, minimizando las operaciones de carga/almacenamiento de registros que requieren el acceso a memoria.

El proceso de asignación óptimo de registros se realiza en dos fases:

1) *Diseño del grafo de interferencias*, entre las variables del programa:

   nodos: las variables

   arcos: entre variables que están activas simultáneamente en el programa

2) *Coloreado del grafo*

   Se intenta colorear el grafo con *n* colores, siendo *n* el nº de registros disponibles en la máquina. A los nodos no coloreados se le asignan posiciones de memoria y se utilizan instrucciones de carga/almacenamiento.

- **Optimizaciones dependientes de la máquina**, aprovechan el conocimiento de las arquitecturas específicas**:**
  - Multiplicación por una constante y sustitución por sumas y desplazamientos
  - Elección del desplazamiento más corto en los saltos

Tenemos como dos importantes ayudas de la arquitectura de un procesador, que nos puede prestar al diseño del compilador son:

- *Regularidad del repertorio*, es decir, ortogonalidad de sus elementos para simplificar la generación de código.
- *Proporcionar funciones primitivas*, no soluciones codificadas en las instrucciones, pues estas resultan difíciles de utilizar cuando el caso se aparta ligeramente del que originó su diseño.

Conclusión:

Al diseñar una arquitectura de un computador, aparte de que queremos que funcione correctamente, también queremos que tenga el máximo rendimiento posible para las diferentes aplicaciones que podrían utilizarse. Y podemos conocer el rendimiento del diseño a través del tiempo de ejecución, siendo el más fiable, permitiendo compararlo con otro diseño y deducir cual es más rápida en cuestión. Mediante los benchmarks, en el que se le aplican diversas pruebas que se ejecutan, también nos permite conocer el rendimiento del diseño de la arquitectura hecha o de un dispositivo físico, demostrando que cumple con

los tiempos de ejecución acordados. Y si el diseño de nuestra arquitectura no trabaja óptimamente, quiere decir, que debemos mejorar el rendimiento de la arquitectura, y por eso hay que tomar en cuenta que influye demasiado la forma en la que mejoraremos el rendimiento, como son las del diseño y las del compilador de lenguajes. Por eso cuando se diseña la arquitectura se debe de tomar en cuenta en que aplicaciones se va a usar para hacer directamente las conexiones en que funcionen con el máximo rendimiento posible, ya que existen diferentes tipos de conexiones, en la que más adelante sería difícil cambiar estas conexiones físicamente, también hay que tomar en cuenta el compilador, porque la arquitectura tiene que llegar a aceptar el compilador, que hace como nuestro convertidor de ordenes de alto nivel para pasarlas de bajo nivel, en donde estos cambios para mejorar el rendimiento serian dentro del repertorio de instrucciones del compilador. Finalmente, para poder diseñar una arquitectura de un computador optimo debemos de tomar en cuenta las conexiones y diseño (físico) de la arquitectura que tendrá que aceptar el compilador y el compilador mismo tendrá que poder adaptarse a la arquitectura para que funciones correctamente el computador, y sabremos que mejorar al fijarnos que componentes o tareas utilizan más tiempo, hacer las mejoras necesarias sobre el componente especifico.

Bibliografía

Frutos, A. y Frutos. (2021). ¿Qué es un benchmark y para qué sirve? Consultado el 27 de noviembre de 2021 en https://computerhoy.com/noticias/moviles/que-es-benchmark-que-sirve-40273

UCM. (s/f). Tema 4: Rendimiento del procesador. Consultado el 27 de noviembre de 2021 en http://www.fdi.ucm.es/profesor/jjruz/web2/temas/ec4.pdf

Universidad Europea de Madrid. (s/f). *Medidas de rendimiento* (Articulo). Consultado el 27 de noviembre de 2021

Universidad de Oviedo (s/f). Medición del Rendimiento de Computadores. Consultado el 27 de noviembre de 2021 en http://www.atc.uniovi.es/inf_superior/atc/PARALELAS/4atc_2rend_enfriadas.pdf