Green computing

Computación sostenible y para la sostenibilidad

Ivan Rodero Castro
Francesc Guim Bernat

PID_00191922

Universitat Oberta de Catalunya

Índice

Introducción

Este módulo didáctico se plantea como una breve introducción a la *green computing* y pone un énfasis especial en la eficiencia energética para los sistemas de altas prestaciones. Servirá de referencia, pero se espera que se complemente con materiales adicionales y lecturas de artículos de búsqueda relacionados.

Green computing es un término muy amplio que cubre desde el desarrollo de arquitecturas de bajo consumo hasta técnicas de eficiencia energética, como por ejemplo la utilización de fuentes de energía renovables.

En primer lugar, repasaremos los fundamentos básicos de este concepto, como las definiciones asociadas y las métricas que se relacionan con las mismas. A continuación, estudiaremos varias técnicas para la mejora de la eficiencia energética en el contexto de diversos tipos de sistemas. También veremos una breve introducción a algunos conceptos de sostenibilidad a partir de casos de uso ilustrativos. Finalmente, encontraremos una lista de lecturas recomendadas de los temas más relevantes que tratamos durante el módulo.

En el primer módulo didáctico de la asignatura, hemos visto las principales motivaciones por las que la eficiencia energética es clave en la computación de altas prestaciones. Hay que recordar, sin embargo, que sin una mejora de órdenes de magnitud en la eficiencia energética no se espera que sea posible desarrollar la próxima generación de supercomputadores –que tienen que llegar a proporcionar rendimiento de exaflop en unos pocos años–, a raíz de su creciente y desmesurado aumento de la potencia y de las energías necesarias para operarlos.

Objetivos

Los materiales didácticos de este módulo contienen las herramientas necesarias para permitiros alcanzar los objetivos siguientes:

1. Conocer los fundamentos y las métricas utilizados habitualmente en el contexto de la *green computing* y de la eficiencia energética.

2. Conocer las características y las técnicas principales de gestión de potencia/energía eléctrica en general y para sistemas de altas prestaciones.

3. Saber diferenciar entre computación sostenible y computación para la sostenibilidad, y conocer sus características.

4. Conocer las principales líneas de búsqueda en el ámbito de la *green computing* y la eficiencia energética.

5. Adquirir una visión crítica de artículos de búsqueda relacionados con la *green computing*.

1. Fundamentos y conceptos básicos

Green computing es un término que normalmente hace referencia a la utilización eficiente de los recursos informáticos, pero se trata de un concepto muy amplio. Aun así, la motivación principal está relacionada con la utilización de recursos informáticos para minimizar el impacto ambiental, maximizar la viabilidad económica y la sostenibilidad y garantizar las obligaciones sociales. La *green computing* también está muy relacionada con otros movimientos similares como, por ejemplo, la reducción del uso de materiales ambientalmente peligrosos como los CFC, la promoción de los materiales reciclables, la minimización del uso de componentes no biodegradables y el fomento de los recursos sostenibles.

Aunque actualmente hay mucho interés en soluciones que permitan utilizar fondos de energía más verdes y sostenibles, nosotros nos centraremos en la eficiencia energética. Podemos hablar de eficiencia energética en varios entornos, desde la producción (por ejemplo, con tecnología solar), distribución (en la que se pierde una gran cantidad de energía) o consumo (en el que, en la práctica, se desaprovecha un gran porcentaje de la electricidad que se consume).

1.1. Métricas

En este apartado, veremos las métricas y los fundamentos más básicos relacionados con la *green computing* y, más concretamente, con la eficiencia energética. Así pues, a continuación veremos las definiciones más básicas que utilizaremos en este módulo y que hay que tener presentes.

En el mundo de la física, la **energía eléctrica** se puede definir como la forma de energía que resulta de la existencia de una diferencia de potencial entre dos puntos, lo que permite establecer una corriente eléctrica entre estos puntos cuando se les pone en contacto por medio de un conductor eléctrico y así obtener trabajo. La energía eléctrica puede transformarse en otras muchas formas de energía, como por ejemplo la energía térmica en el contexto de los circuitos electrónicos.

La **potencia eléctrica** es la relación del paso de la energía de un flujo por unidad de tiempo, es decir, la cantidad de energía liberada o absorbida por un elemento en un tiempo determinado. Así pues, la energía es una función de integración de la potencia a lo largo del tiempo y, por este motivo, reducir la potencia no tiene que significar de manera necesaria una reducción de energía.

En el sistema internacional, las unidades de medida para estos dos conceptos son el julio (J) para la energía y el vatio (W) para la potencia.

Podemos definir muy genéricamente la **eficiencia energética** como la obtención de un resultado que minimiza el consumo de energía o, de manera complementaria, todas aquellas acciones que tienden a reducir su consumo.

Las **métricas** son esenciales para medir cuantitativamente y, por lo tanto, para evaluar la eficiencia del consumo de energía. Las métricas forman la base para la toma de decisiones y, de hecho, en los últimos años se han propuesto distintas métricas que se utilizan en la actualidad.

A continuación, veremos dos tipos de métricas diferentes: para equipos individuales y para sistemas paralelos.

1.1.1. Métricas para equipos individuales

La métrica más básica de eficiencia energética proviene de la comunidad de diseño de circuitos y es la fórmula ED^n. En esta fórmula, E es la energía consumida durante la ejecución de una aplicación, D es el tiempo necesario para completarla, y n es un parámetro entero no negativo que caracteriza el equilibrio entre E y D. Así pues, esta métrica combina energía y tiempo.

ED2P, que es el caso específico de ED^n cuando $n = 2$, es la variante más utilizada de esta métrica, especialmente cuando se utilizan técnicas de DVFS[1]. Con esta métrica, se puede anular la influencia del escalado de la frecuencia, puesto que E es proporcional al cuadrado de la frecuencia, mientras que D^2 es proporcional a la inversa del cuadrado de la frecuencia. *ED2P* considera el rendimiento y el consumo de energía, pero no tiene en cuenta las necesidades de los diferentes sistemas.

(1) *DVFS* es la sigla de *dynamic voltage and frequency scaling*, o escalado dinámico de voltaje y frecuencia.

Para generalizarla, también se puede formular la métrica *ED2P* con peso (***weighted ED2P***), tal y como se muestra a continuación:

$$Weighted \quad ED2P = E^{(1-\partial)} \times D^{2(1+\partial)}$$

En esta métrica, $|\partial| \leq 1$ es un factor de peso determinado por las preferencias del usuario. Esta métrica intenta favorecer el rendimiento cuando $0 < \partial$, y la energía cuando $\partial < 0$. Si $\partial = 0$, el rendimiento y la energía se tratan del mismo modo y resulta la *ED2P* convencional.

1.1.2. Métricas para sistemas paralelos

Cuando el valor de n en ED^n es muy grande, se produce un sesgo a favor de los sistemas masivamente paralelos. De hecho, la variante inversa de ED^n, es decir, $1/ED^n$, representa $Rendimiento^n/Potencia$ o $Flops^n/W$. Los flops, como ya sabemos, hacen referencia al número de operaciones en coma flotante por segundo. Cuando un supercomputador tiene s procesadores y cada uno de estos procesadores proporciona F flops con P vatios, la métrica $Flops^n/W$ se puede reformular de la manera siguiente.

$$\frac{Flops^n}{W} = \frac{(s \cdot F)^n}{s \cdot P} = s^{n-1} \cdot \frac{F^n}{P}$$

En la lista Green500, que veremos más adelante, se utiliza esta métrica con $n = 1$, puesto que con $n > 1$ el valor de esta métrica aumenta de manera exponencial con el número de procesadores s y, por lo tanto, no sería fiable.

Ved también

La lista Green500 se ve en el subapartado 1.3 de este módulo didáctico.

El ***TCO***[2] es el coste total de propiedad de un sistema informático, y hace referencia al coste total del sistema durante su vida, incluidos los costes de adquisición, mantenimiento, energía consumida y eliminación. En los últimos años, el coste de la energía ha estado afectando al *TCO* casi al mismo nivel que la compra inicial. Además, hay costes de capital relacionados con la ocupación del espacio que pesan de manera considerable en el último cálculo del *TCO*. La eficiencia energética, la densidad, la refrigeración líquida, la fiabilidad, etc., son factores que contribuyen al bajo coste de mantenimiento y, por lo tanto, al *TCO*.

(2) *TCO* es la sigla de la expresión inglesa *total cost of ownership*.

La ***PUE***[3] significa eficacia en el uso de la energía. Se mide a partir de la cantidad de energía eléctrica que entra en un clúster o centro de datos (normalmente, se utiliza más bien en centros de datos), por lo general se usa para hacer cálculos y, por lo tanto, es absorbida por los sistemas de tecnología de la información (*TI*[4]). La *PUE* se define de la manera siguiente:

(3) *PUE* es la sigla de la expresión inglesa *power usage effectiveness*.

(4) *TI* es la sigla con la que se conocen los sistemas de tecnología de la información.

$$PUE = \frac{\text{Potencia total de la instalación}}{\text{Potencia de los equipos } TI}$$

Esta fórmula también se puede expresar de manera más detallada como se muestra a continuación:

$$PUE = \frac{\text{Refrigeración} + \text{Pérdidas de potencia} + \text{Iluminación} + TI}{TI}$$

La *PUE* teórica perfecta es igual a 1, y la *PUE* media de los centros de datos es aproximadamente de 2,13.

La ***ERE***[5] hace referencia al aprovechamiento del calor generado por los sistemas informáticos. Esta técnica se denomina energía térmica, y la reutilización implica que *PUE* < 1, lo que matemáticamente no tiene sentido. La *ERE* es una métrica que pretende recoger esta idea de energía total ahorrada, y se define de la manera siguiente:

(5)*ERE* es la sigla de la expresión inglesa *energy reuse effectiveness*.

$$ERE = \frac{\text{Energía total} - \text{Energía reutilizada}}{\text{Energía consumida por } TI}$$

La ***CUE***[6] representa la eficacia en el uso de carbono. Mide el total de emisiones de CO_2 causadas por el centro de datos y dividido por la energía de la carga del sistema, que es la energía consumida por los servidores. La fórmula se puede expresar de la manera siguiente:

(6)*CUE* es la sigla de la expresión inglesa *carbon usage effectiveness*.

$$CUE = \frac{CO_2 \text{ emitido } (KgCO_2eq)}{\text{Unidad de energía (Kwh)}} \times \frac{\text{Energía total del sistema}}{\text{Energía consumida por } TI}$$

Es decir:

$$CUE = CEF \times PUE$$

En esta última expresión, *CEF* es el factor de emisión de carbono (kgCO2eq/kWh) del sistema, de acuerdo con los datos publicados por el Gobierno de la región de operación. El *CEF* depende de la mezcla de producción de energía que en última instancia alimenta al centro de datos. Este indicador puede ser muy bajo; por ejemplo, en el caso de un centro de datos que funciona a partir de la electricidad generada por una central hidroeléctrica. En realidad, el *CEF* cambia de país a país y se actualiza anualmente. En EE. UU., cambia incluso entre diferentes estados y la media es de 0,59 kgCO2eq/kWh.

Además de estas métricas, también hay otras dos que se acostumbran a utilizar para medir la eficiencia energética en relación con la productividad. La primera es la productividad de la tecnología de la información por vatio (*IT-PEW*[7]). La segunda es el índice de eficiencia energética y productividad (*DC-EEP*[8]). Estas métricas se definen con las ecuaciones siguientes:

(7)*IT-PEW* corresponde a la expresión inglesa *IT productivity per embedded watt*.

(8)*DC-EEP* corresponde a la expresión inglesa *data center energy efficiency and productivity*.

$$IT - PEW = \text{Productividad} \,/\, Embedded \quad watt$$

$$DC\text{-}EEP \quad Index \;=\; PEW/PUE \;=\; \text{Productividad} \,/\, \text{Potencia total cluster}$$

Aquí, la *Productividad* es la producción de servicio del sistema, y *Embedded watt* es la potencia de los sistemas de tecnología de información. Así pues, mientras que IT-PEW indica la eficiencia energética de los equipos informáticos, *DC-EEP index* señala la correspondiente de todo el clúster o centro de datos.

Igualmente, encontramos otras métricas más relacionadas con los centros de datos en Internet, como por ejemplo rendimiento por vatio, rendimiento por *TCO* y rendimiento por coste de la energía y refrigeración.

1.2. Caracterización del consumo energético

Para hacer una gestión eficiente de la energía, es preciso conocer y caracterizar diferentes patrones de uso de la energía del sistema. En otras palabras, necesitamos saber dónde y cuándo se ha consumido la energía y quién es responsable de su uso. La construcción de un perfil de estas características se denomina **perfil de consumo de energía**. A partir de estos perfiles de consumo de energía, se han podido desarrollar diferentes técnicas de análisis, como por ejemplo las basadas en simulaciones, el modelado analítico, la medida directa del consumo de energía, el análisis de consumo de energía basado en la monitorización o técnicas de muestreo de la potencia y software de instrumentación, que pasamos a detallar a continuación:

1) En **las técnicas basadas en simulaciones**, las características de consumo de energía están integradas en un simulador, que se basa en las características de potencia derivadas de medidas de una muestra. Estos simuladores estiman el rendimiento y el consumo de energía mediante el seguimiento de la ejecución de las aplicaciones. Las técnicas de simulación se han utilizado para varios tipos de entornos y de componentes, como por ejemplo el procesador, la memoria, el disco duro o incluso máquinas enteras. El mecanismo de simulación es útil para caracterizar la actividad y la potencia, pero puede haber un desajuste entre la simulación y los sistemas reales debido a la inexactitud de los modelos de simulación. Además, las simulaciones pueden llegar a tardar bastante tiempo en ejecutarse.

2) Las **técnicas analíticas de modelado** abstraen el consumo de energía mediante funciones analíticas sobre un conjunto de parámetros. El consumo de energía se estima a partir de correlaciones entre la potencia eléctrica y las variables proporcionadas. Este tipo puede servir para modelizar, por ejemplo, el almacenamiento compuesto por un conjunto de discos redundantes. En este caso, los parámetros relacionados con la energía son: (a) la potencia necesaria para cada disco en modo activo o listo y (b) la energía y el tiempo necesarios para que un disco pueda ponerse en marcha y apagarse. Teniendo en cuenta los parámetros de los discos y sus otras características, el modelo predice el comportamiento de consumo eléctrico de los sistemas de almacenamiento, la política de gestión de disco y las cargas de trabajo asociada. Es posible utilizar ecuaciones lineales, así como cuestiones de gestión térmica de los sistemas. Un ejemplo es el desarrollo de modelos que emulan la temperatura a partir de las características de los componentes electrónicos. El modelado analítico tiene la ventaja de ser una solución basada en software. Aun así, la exactitud de la solución depende de la granularidad del modelo.

3) La **medida en tiempo real del consumo de energía** es una solución directa al problema de la caracterización del consumo eléctrico. Para llevarla a cabo, hay que emplear herramientas de medida adicionales –por ejemplo, multímetros– que midan y graben el consumo de energía en tiempo de ejecución. Un tema clave es la selección de la aplicación apropiada que sirva como referencia[9] y que sea representativa. Sin embargo, este enfoque no es práctico para la mayoría de los sistemas, sobre todo para sistemas de una cierta escala en los que se necesitarían miles de multímetros, lo cual no siempre es viable desde el punto de vista económico.

(9)En inglés, *benchmark*.

4) La monitorización **basada en el análisis de consumo de potencia** se lleva a cabo a partir de los contadores de rendimiento (PMC) que hay integrados en los procesadores actuales. A partir de estos contadores, podemos extraer perfiles de consumo de energía bastante cuidadosos que se han utilizado para la gestión de energía y la térmica. Sin embargo, la exactitud de los contadores, y por lo tanto la caracterización del tipo de perfil, está limitada por los tipos de eventos que pueden monitorizarse. Por ejemplo, las medidas relacionadas con operaciones que se llevan a cabo fuera del procesador son habitualmente mucho menos precisas.

5) Las **técnicas basadas en el muestreo** intentan buscar correlaciones entre el comportamiento del software/sistema y el consumo de energía del sistema. Un ejemplo consiste en medir de manera periódica el consumo de energía junto con el contador de programa (PC) y el proceso ID (PID), y vincular el consumo de energía de la muestra de nuevo a los PID (es decir, los procesos) y los PC (es decir, las fases de ejecución). No es preciso que las técnicas de muestreo dependan del hardware, pero en este caso es más difícil determinar qué componentes utilizan la mayor parte de la energía, y el muestreo quizá no sea lo suficientemente efectivo.

6) Las **técnicas basadas en instrumentación** añaden fragmentos adicionales en el código de las aplicaciones, con el objetivo de recoger información sobre la alimentación y el contexto de la ejecución del programa. Algunos ejemplos pueden ser: insertar código de perfiles en el código ensamblador que mide el consumo de energía, o utilizar herramientas de instrumentación para insertar de manera dinámica rutinas de generación de perfiles en el código binario. La instrumentación se puede utilizar para construir un modelo de energía de grano fino, pero requiere un compilador de propósito especial o el apoyo de herramientas adicionales para modificar el código binario o fuente.

1.3. Lista Green500

La lista Green500 agrupa los supercomputadores ordenados por su eficiencia energética. El objetivo de esta lista es dejar patentes los esfuerzos para mejorar la capacidad de computación sin penalizar el consumo energético.

A diferencia de los Top500, cuya métrica principal es el rendimiento (medido en megaflops), la lista de Green500 se centra en la eficiencia energética (medida en mflops/W).

Ved también

Podéis ver la lista de los Top500 en el subapartado 4.5.2 del módulo "Introducción a la computación de altas prestaciones" de esta asignatura.

La tabla 1 muestra la lista de los 10 supercomputadores energéticamente más eficientes. En general, vemos que no coincide en absoluto con la lista Top500, excepto en el caso de Titan, que es el tercero en la lista Green500 y se trata del computador más potente en la lista Top500 de noviembre del 2012.

Del mismo modo que en los Top500, la lista se elabora a partir de la ejecución del *bechmark* HPL[10]. La diferencia reside en el hecho de que hay unas normas para la monitorización de la potencia del sistema que incluyen utilizar multímetros cuando el sistema ejecuta HPL en todo el sistema y este ha llegado a su temperatura operativa (al menos, después de 15 minutos).

(10)*HPL* es la sigla de la expresión inglesa *high performance linpack*.

Tabla 1. Lista de los 10 supercomputadores más eficientes energéticamente de la lista del Green500 de noviembre del 2012

#	MFLOPS/W	Institución	Computador	Potencia (kW)
1	2.499,44	National Institute for Computational Sciences/University of Tennessee	Beacon - Appro GreenBlade GB824M, Xeon E5-2670 8C 2.600GHz, Infiniband FDR, Intel Xeon Phi 5110P	44,89
2	2.351,10	King Abdulaziz City for Science and Technology	SANAM - Adtech ESC4000/FDR G2, Xeon E5-2650 8C 2.000GHz, Infiniband FDR, AMD FirePro S10000	179,15
3	2.142,77	DOE/SC/Oak Ridge National Laboratory	Titan - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x	8.209,00
4	2.121,71	Swiss Scientific Computing Center (CSCS)	Todi - Cray XK7 , Opteron 6272 16C 2.100GHz, Cray Gemini interconnect, NVIDIA Tesla K20 Kepler	129,00
5	2.102,12	Forschungszentrum Juelich (FZJ)	JUQUEEN - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect	1.970,00
6	2.101,39	Southern Ontario Smart Computing Innovation Consortium/University of Toronto	BGQdev - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect	41,09
7	2.101,39	DOE/NNSA/LLNL	rzuseq - BlueGene/Q, Power BQC 16C 1.60GHz, Custom	41,09
8	2.101,39	IBM Thomas J. Watson Research Center	BlueGene/Q, Power BQC 16C 1.60GHz, Custom	41,09

Fuente: The Green 500

#	MFLOPS/W	Institución	Computador	Potencia (kW)
9	2.101,12	IBM Thomas J. Watson Research Center	BlueGene/Q, Power BQC 16C 1.60GHz, Custom	82,19
10	2.101,12	Ecole Polytechnique Federale de Lausanne	CADMOS BG/Q - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect	82,19

Fuente: The Green 500

1.4. Casos de uso

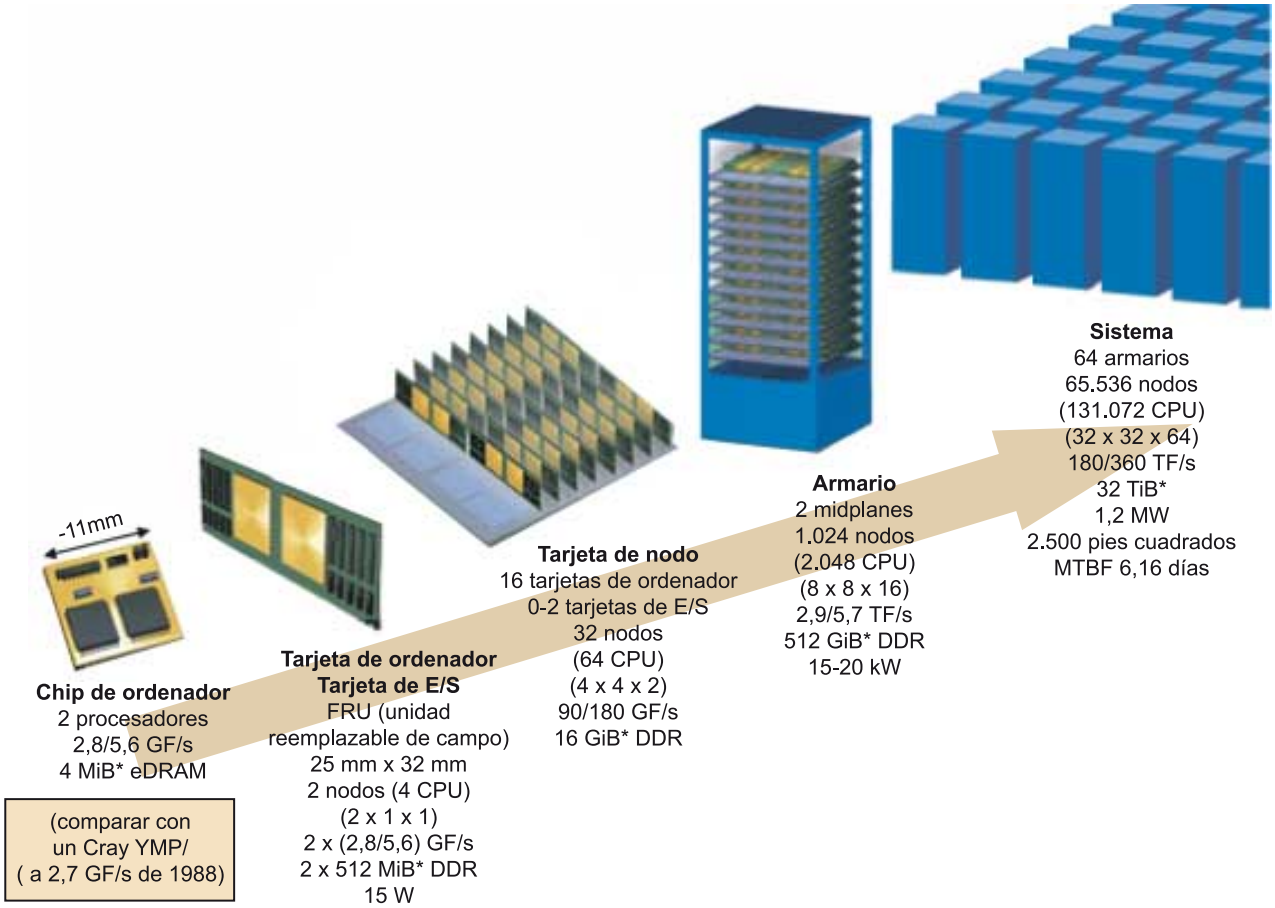
En este subapartado, utilizaremos dos casos de uso específicos de sistemas que se diseñaron con el objetivo de ser muy eficientes energéticamente, aun teniendo características muy distintas. Sin embargo, se trata de casos bastante representativos.

1.4.1. BlueGene/Q

Tal y como se puede observar en la tabla 1, la mitad de los diez supercomputadores más eficientes energéticamente son BlueGene/Q, que diseñó IBM y que precede a BlueGene/L y a BlueGene/P. La idea fundamental de este diseño es obtener un diseño muy compacto –tal y como se puede observar en la figura 1– y que permita obtener un alto rendimiento (petaflops) con un consumo reducido.

Además, BlueGene/Q dispone de un software para gestionar la energía (IBM Systems Director Active Energy, que es el eje central de gestión de la energía). Este software mide, gestiona y controla la potencia y el uso de energía térmica, y también se integra en la infraestructura y los paquetes de gestión empresarial. Para conseguir escalabilidad hasta millones de núcleos y lograr rendimiento de exaflop, la gestión de recursos subyacente a la arquitectura de software tiene que proporcionar un mecanismo flexible con el objetivo de soportar la gran cantidad de aplicaciones de diferentes características y tipos de carga de trabajo que se ejecutan en esta plataforma. Para hacer su tarea, LoadLeveler, el planificador de IBM, consciente de la energía, establece la frecuencia del procesador de manera óptima en el conjunto de nodos en los que se ejecuta un trabajo, así como la frecuencia de los nodos para que consuman el mínimo posible de energía cuando estos no tienen tareas.

Figura 1. Diagrama de la jerarquía de la arquitectura BlueGene (en concreto, se muestra la de BlueGene/P)



Fuente: IBM.

1.4.2. Proyecto Montblanc

El proyecto Montblanc es un proyecto europeo, inicialmente de tres años de duración, que tiene por objetivo diseñar un supercomputador basado en la tecnología de bajo consumo que se utiliza actualmente en tabletas y en telefonía móvil. Con esta tecnología, los investigadores pretenden diseñar un ordenador de prestaciones idénticas, pero con un consumo energético entre cuatro y diez veces menor.

Montblanc integra tecnología de bajo consumo de ARM y aceleradores diseñados para dispositivos móviles con el objetivo de desarrollar una nueva clase de supercomputador igual de potente pero mucho más eficiente en el uso de energía.

Si consideramos que la potencia eléctrica disponible marca el límite del rendimiento de un supercomputador, esta tecnología nos permitiría disponer de sistemas entre cuatro y diez veces más potentes. Aun así, hay retos muy importantes que es preciso solucionar, como por ejemplo la programabilidad, la jerarquía de memoria y la red de interconexión de este tipo de arquitectura.

Esto se debe a que tendrá que soportar un nivel de paralelismo muy superior al de otros tipos de arquitecturas basadas en procesadores más potentes y con más requerimientos de potencia.

En este proyecto, financiado por la UE, hay empresas europeas líderes en el sector tecnológico, como Bull y ARM, y también están los centros de supercomputación de más peso en Europa.

2. Eficiencia energética

En este apartado, nos centraremos en la eficiencia energética y las técnicas de gestión de potencia y energía eléctrica tanto en sistemas de altas prestaciones como en sistemas computacionales en general, como son los centros de datos en Internet.

Aunque veremos técnicas de eficiencia energética para diferentes tipos de dispositivos y en distintos niveles, incluso en el de software, primero estudiaremos la fuente principal de potencia de los circuitos electrónicos y después nos centraremos en las técnicas que nos permiten gestionar su eficiencia.

2.1. Fuentes de potencia en los circuitos electrónicos actuales

El consumo de energía eléctrica en un circuito electrónico proviene de dos fuentes: el consumo estático y el consumo dinámico.

El **consumo estático** es el que se produce debido a las corrientes de fuga[11] que están presentes en los transistores.

[11] En inglés, *leakage*.

El consumo estático es inherente al circuito, incluso cuando el circuito está inactivo. Con el avance de la tecnología, este componente de la potencia es cada vez más importante. El valor de este consumo depende de las características de la tecnología que se emplea, el número de transistores y la temperatura de funcionamiento del circuito.

La potencia estática, $P_{Estática}$, se define como el producto del voltaje de la fuente de alimentación V_s por la corriente estática del circuito i_0, tal y como indica la ecuación siguiente:

$$P_{Estática} = \sum_{1}^{n} i_0 \cdot V_s; \qquad i_0 = i_s\left(e^{\frac{qV_{Diodo}}{KT}} - 1\right)$$

Aquí, i_s es la corriente inversa de saturación o corriente de fuga de los diodos; V_{Diodo} es el voltaje del diodo; q es la unidad de carga; K es la constante de Boltzmann; y T es la temperatura.

El **consumo dinámico** se produce por la carga y descarga de la capacidad de los transistores y las conexiones, y depende de la actividad del circuito.

Es decir, pasa únicamente durante las transiciones, cuando las puertas conmutan. Por lo tanto, es proporcional a la frecuencia de conmutación, y cuanto mayor sea el número de conmutaciones, mayor será también la potencia dinámica. La ecuación siguiente muestra la potencia dinámica:

$$P_{Dinámica} = P_{Conmutación} + P_{Carga}$$

En esta expresión, *a es* la actividad de conmutación, *C* es la capacidad en cada nodo que conmuta, *f* es la frecuencia de reloj y V_s es el valor del potencial de alimentación.

La potencia dinámica tiene dos componentes, como muestra la ecuación siguiente: la potencia de conmutación[12] y la potencia de carga[13]. La primera se debe a las corrientes que van de la fuente de alimentación a tierra cuando el transistor cambia de estado, mientras que la de carga está causada por la corriente necesaria para cargar las capacidades de los elementos conectados a la salida.

(12) En inglés, *crowbar*.

(13) En inglés, *load*.

$$P_{Dinámica} = P_{Conmutación} + P_{Carga}$$

2.2. Mecanismos de gestión de la energía

Hay dos tipos básicos de mecanismos de gestión de la energía, el escalado dinámico de velocidad y el adormecimiento dinámico de recursos:

1) El **escalado dinámico de velocidad**[14] es un mecanismo que cambia de manera dinámica el estado de funcionamiento del componente en cuestión para reducir la potencia; es decir, se ralentiza para reducir el consumo de energía y se acelera cuando es necesario, pero a expensas de un consumo de energía mayor.

(14) En inglés, *dynamic speed scaling*.

Algunos ejemplos de escalados dinámicos de velocidad

Un ejemplo típico es el escalado dinámico de voltaje y frecuencia, o *DVFS* (del inglés *dynamic voltage and frequency scaling*). En este caso, la reducción en el consumo de energía se efectúa mediante la reducción del voltaje de alimentación o la frecuencia de reloj. La mayoría de los procesadores actuales soportan este mecanismo. Algunos ejemplos de esto son los procesadores Intel Xeon o los procesadores/coprocesadores AMD.

La regulación térmica es otro ejemplo de lo mismo. En este caso, lo que se controla es la temperatura del procesador. Del mismo modo que antes, esto se hace mediante la modulación del ciclo de trabajo del reloj del procesador o la reducción de la frecuencia de funcionamiento y del voltaje del procesador.

Otros ejemplos de DSS incluyen memorias multifrecuencia, en las que se puede escalar la frecuencia de trabajo de manera dinámica –y, por lo tanto, la velocidad de acceso a los datos–, así como discos con múltiples velocidades.

Sin embargo, en todos estos métodos la transición entre diferentes estados de funcionamiento consume energía adicional y causa sobrecarga en la latencia.

2) El **adormecimiento dinámico de recursos**[15] es un mecanismo que duerme (o hiberna) componentes de modo dinámico para ahorrar energía y los despierta cuando son necesarios. Cada componente puede estar en un estado activo, en uno de los estados de sueño, o bien en estado de apagado. Tal y como veremos más adelante, en el estándar de la industria ACPI el estado activo se denota C0 y los estados de sueño, C1, C2... Cn. Cada uno de los estados de sueño consume menos energía que el estado C0 sin actividad. Cuanto más profundo es el estado de sueño del procesador menos potencia consume, pero más energía se necesita para despertar. Los controladores de memoria también pueden cambiar la gestión dinámica de la potencia y los discos también pueden soportar los estados activo, listo y en espera. De hecho, todo el equipo se puede administrar también como si fuera un componente que puede estar en los estados activo, suspendido, hibernación o apagado, pero las transiciones entre estados consumen energía y tardan un cierto tiempo.

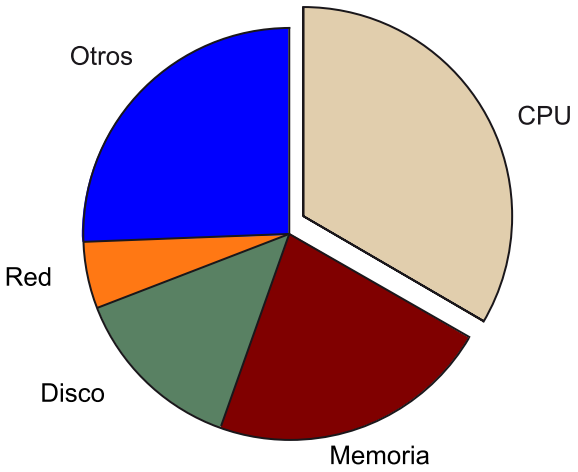
(15) En inglés, *dynamic resource sleeping*.

Aunque el procesador es el principal subsistema que disipa energía eléctrica en los computadores actuales, los otros subsistemas están aumentando cada vez más su demanda de potencia. La figura 2 muestra una posible distribución de la potencia disipada por los distintos subsistemas de un computador estándar.

Hay fuerza técnicas que intentan hacer un uso eficiente de estos subsistemas, tal y como veremos en los subapartados siguientes, y entre las cuales destacan las arquitecturas de baja potencia, los DVFS[16], el apagado de subsistemas (incluso de todo el computador) y la utilización de estados de bajo consumo o planificación eficiente de la carga, entre otros.

(16) *DVFS* es la sigla de *dynamic voltage and frequency scaling*.

Figura 2. Distribución de potencia eléctrica de un computador por subsistema



2.3. Gestión de energía en un ámbito de sistema operativo

En este subapartado tratamos una de las técnicas de gestión de la energía más extendida y que desde hace tiempo está soportada por los sistemas operativos de manera transparente para el usuario. Es complementaria a otras técnicas que se exponen en este módulo didáctico.

La especificación de las **interfaces avanzadas de configuración y energía** (ACPI[17]) fue desarrollada para establecer interfaces comunes en la industria que permitieran la gestión de energía y su configuración en dispositivos que delegan esta responsabilidad en el sistema operativo. En la industria, esto último es conocido con la sigla de OSPM[18].

(17) *ACPI* es la sigla de la expresión inglesa *advanced configuration and power interface.*

(18) OSPM es la sigla de *operating system-directed configuration and power management*.

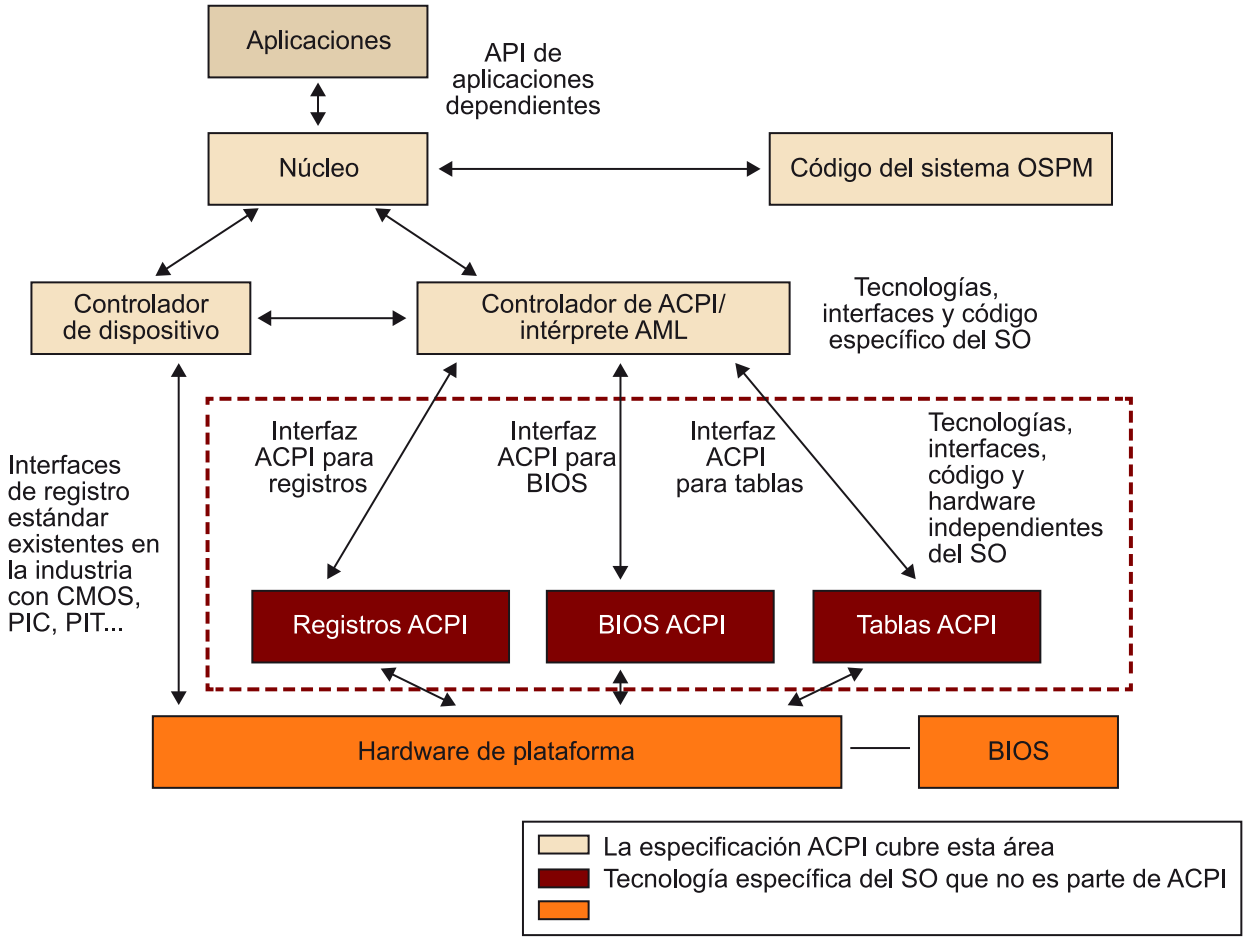
La ACPI es la consolidación de varios intentos y aproximaciones anteriores, y trata de responder a sus mismas necesidades desde un punto de vista menos heterogéneo y más flexible. La ACPI agrupa y sustituye rutinas de código localizadas en la BIOS en una especificación de interfaz muy definida, aunque extensa y muy compleja. La ACPI también proporciona los mecanismos necesarios para llevar a cabo una transición ordenada entre el hardware más antiguo y el más reciente.

Las interfaces y el concepto OSPM que se incluyen en la especificación son transversales a las distintas implementaciones de ordenadores genéricas que hay en la industria. Por ejemplo, ordenadores personales, estaciones de trabajo, teléfonos móviles o servidores son implementaciones que pueden beneficiarse de esta especificación OSPM/ACPI. Hay que decir que los sistemas operativos más difundidos y empleados en el mercado de ordenadores (por ejemplo, Microsoft Windows, GNU/Linux, BSD, etc.) disponen por defecto de soporte para ACPI habilitado en mayor o menor medida.

La especificación utiliza el concepto de conservación de la energía por medio de la transición de dispositivos en estados de bajo consumo, o incluso de no consumo cuando no hacen trabajo útil.

La ACPI describe las interfaces de hardware, software y las estructuras de datos que, una vez implementadas, activan el soporte para ejecutar OSPM y permiten hacer la gestión de energía desde el propio sistema operativo (SO) empleando una interfaz abstracta entre el sistema operativo y el hardware. ACPI también incluye la semántica de estas interfaces. La figura 3 muestra los componentes de software y hardware más relevantes en OSPM/ACPI.

Figura 3. Esquema global del sistema OSPM/ACPI



Fuente: Adaptado de la especificación ACPI.

Como se observa en la figura 3, la especificación describe las interfaces entre componentes, los contenidos de las tablas de descripción de sistema ACPI y la semántica relacionada del resto de los componentes.

Las tablas de descripción de sistema ACPI, que describen un hardware de plataforma concreto, son el corazón de la implementación ACPI al mismo tiempo que el microsoftware de sistema ACPI lleva a cabo, entre sus responsabilidades, el suministro de tablas ACPI (que son independientes de la tecnología) y no así el de una interfaz nativa, que sería menos flexible.

La **responsabilidad funcional del OSPM** es, por lo tanto, disponer del control de acceso directo y exclusivo sobre las funciones de configuración y gestión de energía del hardware.

De este modo, durante la inicialización, el OSPM es responsable de gestionar los eventos de configuración generados por el hardware y de controlar el consumo de energía, el rendimiento y el estado térmico del sistema teniendo en cuenta siempre las preferencias de usuario, las peticiones en un ámbito de aplicación y los objetivos de usabilidad y calidad de servicio.

Las áreas funcionales que permiten al OSPM llevar a cabo estas funciones se describen a continuación:

- *System power management*. La ACPI define mecanismos para permitir que el ordenador pase a estados de bajo consumo. Esto se puede hacer a escala de sistema o de dispositivo.

- *Device power management*. Las tablas ACPI describen el hardware y sus estados de energía y permiten poner un dispositivo en diferentes estados de bajo consumo.

- *Processor power management*. Mientras el sistema operativo está en reposo, la ACPI permite poner el procesador en estados de bajo consumo.

- *Device and processor performance management*. Mientras el sistema se encuentra activo, el OSPM permite efectuar transiciones entre estados para dispositivos y procesadores, con la intención de obtener el balance deseado entre rendimiento y conservación de la energía.

- *Configuration / Plug and play*. La ACPI especifica información que se utiliza para enumerar y configurar el hardware.

- *System events*. La ACPI define mecanismos muy flexibles para encaminar eventos a la lógica de cada dispositivo en hardware.

- *Battery management*. La política de gestión de batería mueve desde la BIOS hacia la ACPI. Las baterías compatibles con ACPI disponen de una pequeña interfaz definida para métodos de control.

- *Thermal management*. La ACPI soporta gestión térmica, de modo que proporciona un modelo escalable a fabricantes que les permite definir zonas, indicadores y métodos de control para una correcta gestión térmica.

- *Embedded controller / SMBus controller*. La ACPI define un hardware estándar y un software de comunicaciones que hace de interfaz entre controladores

del sistema operativo y un controlador SMBus que permite a los fabricantes proporcionar funcionalidades para que puedan ser utilizadas por el sistema operativo y las aplicaciones.

La ACPI reconoce cinco estados globales, o *global states*, y un número de estados que suceden en alguno de los cinco estados globales mencionados, tal y como se muestra en la figura 4. Estos se pueden clasificar con las categorías siguientes.

1) *Global states* (G0-G3). Son estados globales con la definición siguiente.

- G0: estado de trabajo o *working state*.
- G1: estado durmiente o *sleeping state*.
- G2: apagado por software o *soft-off state*.
- G3: apagado mecánico o *mechanical-off state*.

También encontramos un quinto estado global, no considerado dentro de este grupo, y que se denomina *legacy*. Este último estado representa el estado del sistema cuando no soporta el modelo ACPI.

2) *C-states*. Se producen en el contexto de G0 (*global working state*). C0, uno de los *C-states* (*CPU-state*), hace referencia al estado de ejecución. En cambio, si queremos ahorrar energía cuando el procesador está parado, son preferibles estados con un valor más alto. De hecho, ninguna instrucción se ejecuta en C1, C2 y C3. Hay que decir que ACPI sustituye el típico bucle de *idle* por defecto para poder entrar en C1, C2 y C3.

3) *P-states*. En el contexto de G0 (*global working state*) y C0 (*CPU executing state*), se producen los *P-states* (*performance states*). Estos estados sirven para modular la frecuencia y el voltaje del procesador cuando está ejecutando instrucciones. Son estados muy efectivos y tienen un subsistema propio en el dominio de la gestión de energía del núcleo Linux (*cpufreq*). Juegan un papel notable en la técnica de *throttling*, en la que el núcleo trata de obtener un compromiso entre la frecuencia, la potencia y el rendimiento deseado.

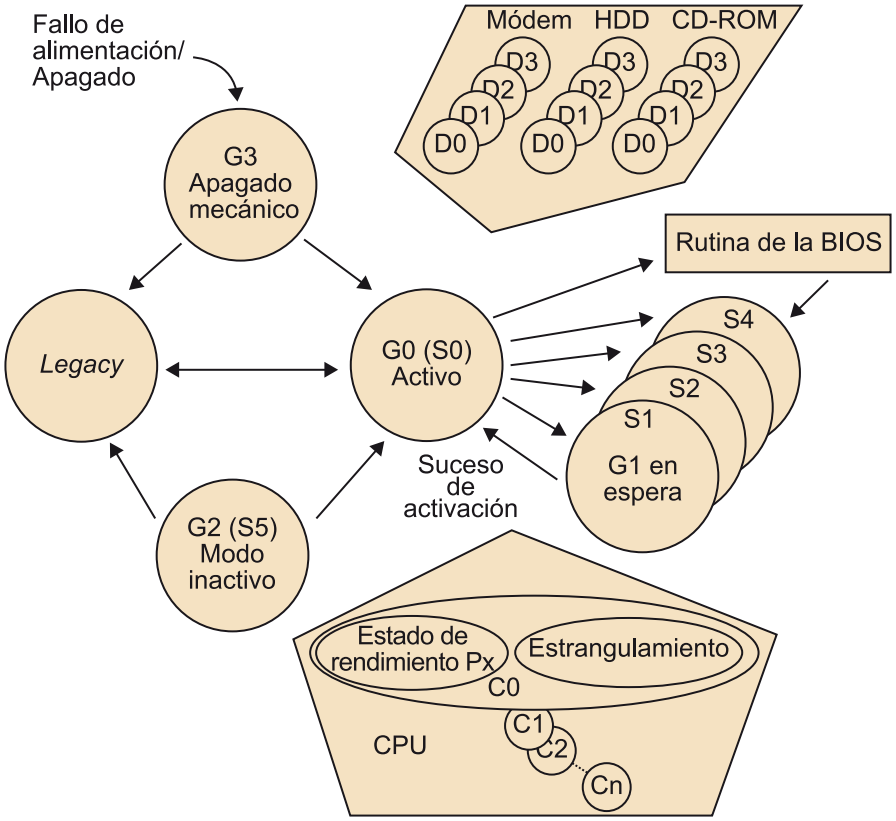
4) *Sleep states*. ACPI dispone de los estados S0-S5 con el significado siguiente:

- S0 es un estado en el que no duerme (*non-sleep state*).
- S1 es el estado *stand-by*, en el que el procesador está parado y la pantalla, apagada.
- S2 no se utiliza.
- S3 consiste en suspender el sistema operativo y mantenerlo en memoria RAM.
- S4 es para hibernar la imagen del sistema operativo en disco.
- S5 es por apagado por software (*soft-power off state*).

4) *Device states* (D0-D3). ACPI define estados de bajo consumo para los dispositivos de la manera siguiente:

- D0 está encendido.
- D3 está apagado.
- D1 y D2 son estados intermedios.

Figura 4. Estados de energía *global system* y transiciones en ACPI global del sistema OSPM/ACPI



Fuente: Adaptado de la especificación ACPI.

2.4. Gestión de energía en centros de procesado de datos

En centros de datos comerciales y en Internet, hay varias técnicas de gestión de energía. Algunas de estas se basan en el encendido y apagado de servidores de modo que se conserva energía al mismo tiempo que se respeta la calidad de servicio, que se define a partir de los SLA[19]). Otras técnicas se basan en el uso de DVFS, de modo que se puede mejorar la eficiencia energética sin penalizar el rendimiento; por ejemplo, en la manera de limitar la potencia del sistema. En el último caso, normalmente se hace con un controlador. Incluso se utilizan sensores de temperatura, flujo de aire, etc., para hacer una gestión térmica eficiente y, por lo tanto, energética.

[19] *SLA* es la sigla de la expresión inglesa *service level agreements*.

En centros de datos virtualizados también podemos encontrar técnicas específicas, como por ejemplo provisión, planificación y consolidación de máquinas virtuales teniendo en cuenta la eficiencia energética (por ejemplo, a partir de los perfiles de las aplicaciones), así como aspectos térmicos como el control de los sistemas de refrigeración o ubicación de máquinas virtuales inteligentes.

Dado que este tipo de técnicas se cubren en las lecturas recomendadas (en el último apartado de este módulo didáctico), no las desarrollaremos aquí, puesto que el foco de este módulo es la computación de altas prestaciones.

2.5. Gestión de energía en computación de altas prestaciones

Un sistema de altas prestaciones típico se puede dividir en tres subsistemas: los nodos *front-end*, los nodos computacionales y los nodos de almacenamiento.

a) Los **nodos *front-end*** ofrecen una interfaz al usuario para que pueda acceder al sistema y de este modo enviar trabajos a los sistemas y monitorizar su estado.

b) Los **nodos de computación** proporcionan recursos de computación de alto rendimiento para procesar los trabajos.

c) Los **nodos de almacenamiento** proporcionan una capacidad de almacenamiento masiva. Estos tres subsistemas están integrados en un sistema fuertemente acoplado por una red de interconexión de alta velocidad.

A diferencia de los clústeres comerciales, las aplicaciones científicas normalmente son altamente acopladas[20] y generalmente implican muchos nodos –incluso un número masivo de los mismos– que trabajan de manera coordinada. La comunicación y la sincronización se entrelazan con el cálculo, y el tiempo de ejecución es relativamente largo. En este contexto, la CPU domina el consumo de energía en el sistema de altas prestaciones (podéis ver la figura 2). Estas características de los sistemas de altas prestaciones imponen ciertos retos y oportunidades para la gestión de su energía, que se pueden clasificar en dos categorías básicas:

[20] En inglés, *tight-couple*.

- Análisis de la fase de ejecución.
- Planificación y asignación de trabajos.

2.5.1. Análisis de la fase de ejecución

En términos generales, las aplicaciones paralelas consisten en computación con la CPU, acceso a memoria, entrada/salida y comunicación/sincronización. En un supercomputador, normalmente la entrada/salida habitual se imple-

menta mediante la interfaz de comunicación para acceder a un subsistema de almacenamiento (sistema de ficheros distribuidos). Por lo tanto, este será un caso especial de comunicación.

Así pues, podemos decir que hay tres tipos de fases durante la ejecución de la aplicación: basadas en CPU, en memoria y en comunicaciones. Cuando una fase está basada en un componente, normalmente los otros se pueden poner en estados de bajo consumo sin degradar de manera significativa el rendimiento de la ejecución de la aplicación. Se han desarrollado una serie de técnicas para aprovechar los recursos al mismo tiempo que se hace una gestión eficiente de la energía. A continuación, estudiaremos algunas de las técnicas más típicas que se centran en cada uno de los subsistemas que acabamos de comentar.

Análisis de la actividad de la CPU

Una de las técnicas más típicas para la gestión de potencia es utilizar DVFS basándose en el estrés de la CPU. Por ejemplo, Linux utiliza CPU como métrica para determinar la implicación de la CPU. En cambio, hay otras posibilidades como por ejemplo utilizar el número de millones de instrucciones ejecutadas por segundo (MIPS). La carga de trabajo se puede descomponer en dos partes: la carga de trabajo del chip (su rendimiento depende de la frecuencia de la CPU) y la carga de trabajo de fuera del chip (su rendimiento no depende de la frecuencia de la CPU).

El cambio de frecuencia de la CPU tiene un cierto impacto en el tiempo de ejecución. Este se puede modelizar de la manera siguiente.

$$1+\delta=\frac{T(f)}{T(f_{máx})}\approx\frac{mips\left(f_{máx}\right)}{mips(f)}\approx\beta\frac{f_{máx}}{f}+(1-\beta)$$

En esta expresión, β cuantifica el nivel de intensidad de la carga de trabajo fuera del chip, δ es la desaceleración relativa del rendimiento, $T(f)$ es el tiempo de ejecución del trabajo en la frecuencia de CPU f, $mips(f)$ es la media de los MIPS para la frecuencia de CPU f y f_{max} es la frecuencia máxima de la CPU.

También se pueden aprovechar las fases intensivas de memoria para seleccionar de la manera adecuada la frecuencia de la CPU. Los fallos de memoria caché de nivel más bajo son buenos indicadores de si una fase de ejecución es intensiva de memoria. Una ejecución se puede dividir en una secuencia de ventanas en la que cada ventana contiene un número fijo de ciclos de reloj.

Al final de cada ventana, se pueden determinar las características de la fase correspondiente. Por ejemplo, es posible utilizar una ratio de fallos de memoria caché de nivel 3 (exterior) con valor 0,4 como umbral para detectar fases intensivas de memoria. Por encima de este umbral, la nueva frecuencia de la CPU f_{new} se calcula de la manera siguiente.

$$f_{New} = f_{Old} \times \frac{\text{Instrucciones ejecutadas en la ventana activa}}{\text{Instrucciones ejecutadas en la última ventana} \times 100} \times 100$$

Si la fase no es intensiva de memoria, la CPU funciona a máxima frecuencia. Otras técnicas también combinan los MIPS con la ratio de fallos de memoria caché para determinar el tipo de fase.

Tal y como hemos visto durante el resto de los módulos didácticos, las aplicaciones en computación de altas prestaciones son aplicaciones paralelas, normalmente MPI. Así pues, muchas de las técnicas de gestión de energía para computación de altas prestaciones están basadas en la detección de fases en las que se puede sacar provecho de los mecanismos de control de potencia de los diferentes subsistemas, como por ejemplo la CPU.

Ved también

Podéis ver los *MPI* (*message passing interface*) en el subapartado 3.2.1 del módulo "Introducción a la computación de altas prestaciones" de esta asignatura.

En este sentido, una de las técnicas es la de detectar fases en programas MPI en las que se aplicarán niveles inferiores de consumo en la CPU, basados en la memoria. Estas fases de la ejecución de la aplicación MPI se pueden reconocer a partir de dos pasos.

1) En el primer paso, el programa se divide en bloque utilizando dos reglas:

- Cualquier operación MPI determina el límite de un bloque.
- Si la presión en la memoria cambia de manera abrupta, el límite de un bloque sucede en este cambio.

Para implementar la primera regla, se pueden interceptar llamadas MPI (por ejemplo, mediante la interfaz PMPI), y para la segunda, se pueden utilizar las operaciones por fallo de memoria caché como indicadoras de la presión en la memoria.

2) En el segundo paso, los bloques se unen en fases. Dos bloques adyacentes se unen si la presión que ejercen sobre la memoria está entre el mismo umbral. A partir de esto, se puede aplicar un algoritmo para establecer la frecuencia de la CPU más apropiada que optimice la eficiencia energética para aquella fase determinada.

Estas técnicas determinan si la CPU limita o no el rendimiento del sistema a partir de las actividades de la CPU como por ejemplo la utilización de la CPU, MPIS o el ratio de fallos de memoria caché. Estas son efectivas para aquellas fases en las que operaciones externas bloquean el rendimiento de la CPU. En

cambio, en algunas situaciones, el rendimiento de la CPU no es el límite del rendimiento de todo el sistema. Por ejemplo, la CPU podría estar ocupada en un bucle a la espera de otros eventos y, por lo tanto, no es preciso que opere a la frecuencia máxima. Las técnicas anteriores no detectan esta situación, y se puede perder la oportunidad de ahorrar energía.

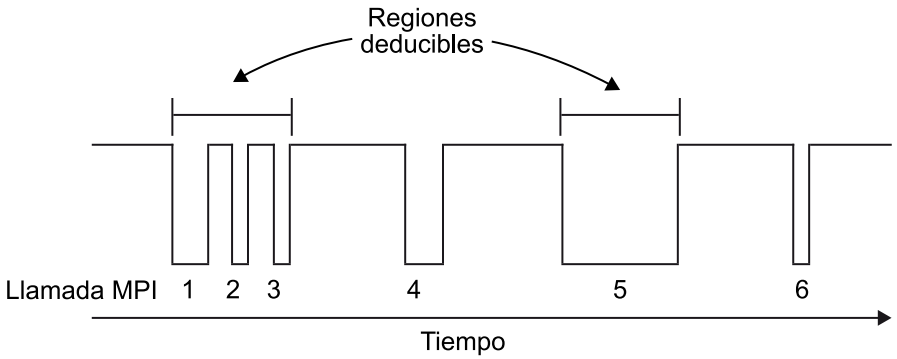
Análisis de la fase de comunicación

Una **ejecución delimitada por comunicación** sucede cuando un nodo tiene que esperar a otro para acabar la comunicación, ya sea porque el dispositivo es más lento o porque se ha dado más trabajo al nodo.

Se han desarrollado técnicas para reconocer estas situaciones y tomar medidas para ahorrar energía. A continuación, se tratan dos de las más comunes, que además lo son de la gestión de energía para aplicaciones de altas prestaciones.

Por un lado, en el análisis de regiones intensivas de comunicación, la fase de comunicación es un periodo en la ejecución de una aplicación en el que la comunicación es intensiva pero no intensiva de CPU. La idea es detectar estas fases en tiempos de ejecución y aplicar DVFS a la CPU en los momentos adecuados para ahorrar energía. Esta técnica trata de interceptar y almacenar las secuencias de llamadas MPI durante la ejecución de un programa. Un segmento de un código de un programa se conoce como **región reductible** si hay una concentración elevada de llamadas MPI (por ejemplo, las llamadas 1-3 de la figura 5) o si una llamada MPI es lo bastante larga (por ejemplo, la llamada 5 de la figura 5). Cuando se entra otra vez en una región reductible que se ha podido detectar, el procesador se pone a la frecuencia apropiada más reducida para ahorrar energía sin degradar el rendimiento de la ejecución de la aplicación.

Figura 5. Ejemplo de traza de un programa de MPI



Las llamadas 1-3 forman una región reductible de múltiples llamadas juntas. La región 5 es una región reductible, puesto que es una llamada larga. Las llamadas 4 y 6 no se pueden reducir, puesto que ni están lo bastante juntas ni son lo suficientemente largas.

Por otro lado, también encontramos el análisis de desequilibrio entre nodos. Cuando la carga de cálculo no está muy equilibrada en todos los nodos, un nodo que llega antes a un punto de sincronización debe esperar los otros nodos más lentos. Cuando esto sucede, se dice que hay *slack* en el nodo por des-

equilibrio. Esta situación puede suceder varias veces si el código está en un bucle. Así pues, lo que se intenta es poner el nodo más rápido a una frecuencia más baja para conservar la energía sin pérdida de rendimiento significativa. Se puede calcular el *slack* de un nodo como el tiempo de espera en una iteración dividido por la longitud de la iteración. El *slack* de red de un nodo se calcula como la diferencia entre su propio *slack* y el *slack* mínimo de los nodos.

2.5.2. Planificación y asignación de trabajos

Los principios de conservación de energía por medio de la planificación y asignación de tareas desarrolladas inicialmente para clústeres formados con hardware de gran consumo también se pueden aplicar a la computación de altas prestaciones. Sin embargo, las soluciones técnicas empleadas con éxito en los centros de datos, como por ejemplo la concentración de la carga, son ineficaces para supercomputadores porque sus características de carga de trabajo y la demanda de rendimiento son diferentes. Nos centraremos en la asignación de tareas entre nodos, puesto que dentro de un nodo se pueden aplicar técnicas como las vistas en un ámbito de sistema operativo.

Un ejemplo de gestor de sistemas de colas que tiene un cierto soporte para la gestión de energía en los nodos inactivos es SLURM. Un nodo que permanece inactivo durante un periodo de tiempo puede colocarse en un estado de baja energía y vuelve al estado normal una vez que se asigna un nuevo trabajo a aquel nodo.

Para evitar un aumento instantáneo de la demanda de potencia por el hecho de que el número de nodos activos cambian rápidamente, SLURM puede poner en marcha y apagar los nodos despacio. Sin embargo, no proporciona ninguna política de gestión de energía. El usuario decide cuánto tiempo tiene que pasar antes de cambiar al modo de energía más baja, los estados de potencia deseados y el número máximo de nodos que pueden cambiar de estado por minuto. La gestión de energía de SLURM puede resultar útil, pero se ha llevado a cabo poca investigación sobre cómo utilizarla de una manera eficaz, sobre todo en el modo de asignar tareas a los nodos necesarios para la conservación de energía. Algunas de las iniciativas en el mundo de la investigación que intentan mejorar las políticas de planificación tienen en cuenta técnicas como por ejemplo DVFS, pero todavía no han llegado a implantarse en sistemas reales.

3. Computación sostenible y para la sostenibilidad

En la *green computing*, normalmente se habla de desarrollar soluciones para que la computación sea sostenible (por ejemplo, no superando ciertos límites de potencia eléctrica puesto que podría no estar disponible). En cambio, la computación también se puede utilizar para mejorar la sostenibilidad del medio ambiente, por ejemplo, con soluciones para desarrollar redes eléctricas inteligentes[21] que hagan un uso y transporte más útil de la energía eléctrica, para simular posibles ubicaciones de placas solares o edificios en un barrio o elegir la ubicación de la instalación de manera idónea en función del clima de la zona y los tipos de proveedores de electricidad locales. En esta segunda categoría, podemos encontrar infinidad de ejemplos. Sin embargo, en este apartado trataremos brevemente el primer tipo.

[21] En inglés, *smart grid*.

Una de las primeras manifestaciones del movimiento *green computing* fue el lanzamiento del programa Energy Star en 1992. Energy Star es una etiqueta voluntaria que reconoce los productos informáticos que reducen al mínimo el uso de energía y aumentan al máximo la eficiencia. Energy Star se aplica a productos como monitores de ordenador, televisores y dispositivos de control de temperatura, como por ejemplo refrigeradores, aires acondicionados y artículos similares. Algunas prácticas que se han adoptado incluyen, por ejemplo, desde apagar el monitor cuando no está en uso o la utilización de monitores más eficientes, hasta sistemas de refrigeración más eficientes.

En este apartado, utilizaremos dos casos de uso para ilustrar distintos tipos de técnicas y diseños de sistemas sostenibles. El primero es la técnica denominada *free cooling*, que permite reducir drásticamente el consumo eléctrico del sistema, y el segundo es un caso particular de microcentro de datos basado en energía solar.

3.1. *Free cooling*

Una tendencia actual en el diseño de centros de datos es la utilización del método *free cooling* para la refrigeración, que está teniendo un gran impacto económico y medioambiental.

El ***free cooling*** es un método económico que consiste en la utilización de las bajas temperaturas del aire exterior para ayudar en la refrigeración del agua, que después se utiliza para refrigerar los sistemas.

Hay tres maneras básicas de utilizar el *free cooling*:

1) *Strainer cycle*. La torre de enfriamiento de agua se puede relacionar directamente con el flujo a través del circuito del agua refrigerada. Si la torre de refrigeración está abierta, entonces se requiere un filtro para eliminar cualquier residuo que pueda acumular dentro de la torre. El ahorro de costes se asocia al uso limitado del agua de refrigeración, pero hay un mayor riesgo de corrosión utilizando este método.

2) *Plate and frame heat exchanger*. Con un intercambiador de calor, se transfiere calor directamente desde el circuito de agua de refrigeración a la torre de refrigeración. El intercambiador mantiene el agua de la torre de refrigeración separada del refrigerante que fluye a través de los serpentines de refrigeración. Por lo tanto, el agua de refrigeración se enfría previamente. El ahorro de energía se reduce por el refrigerador de carga y, de este modo, se produce una reducción en el consumo de energía. Hay un aumento en el coste debido a la bomba que se necesita para compensar las diferencias de presión.

3) *Refrigeration migration*. La disposición de una válvula dentro del agua de refrigeración abre un camino directo entre el condensador y el evaporador. El líquido del circuito de refrigeración se vaporiza y la energía se transporta directamente al condensador, donde el agua de la torre de refrigeración la enfría y condensa. Este método está basado en la idea de que el refrigerante tiende a desplazarse hacia el punto más frío en un circuito de refrigeración. El ahorro de costes asociados a este método se debe a la inactividad del compresor, puesto que el ventilador y las bombas están operativos.

Ejemplos de uso de técnicas de *free* cooling

La empresa Google tiene dos centros de datos en Europa (uno en Bélgica y otro en Finlandia), que no utilizan sistemas de refrigeración tradicionales, sino que se refrigeran a partir del agua como fuente de energía natural y sostenible.

Otro ejemplo es el SuperMUC del Centro de Supercomputación de Leibniz, que utiliza una manera nueva y revolucionaria de refrigeración por agua desarrollada por IBM. Este método está relacionado con la tolerancia en aumento de los componentes electrónicos a temperaturas más elevadas, que hace que se necesite menos refrigeración y, en consecuencia, menos energía.

3.2. Caso de uso: Parasol

Parasol es un microcentro de datos solar desarrollado por investigadores de Rutgers, State University of New Jersey, en Estados Unidos. Está formado por un pequeño contenedor, un conjunto de paneles solares y baterías. El contenedor se encuentra en una estructura de acero colocada en la azotea de uno de los edificios del campus. Los paneles solares se montan en la parte superior de la estructura de acero de modo que pueda adquirir la energía solar y proteger el contenedor del sol. El contenedor aloja dos armarios de servidores de bajo consumo (hasta 160) y equipos de red. Utiliza *free cooling* siempre que es posible, y activa el aire acondicionado convencional cuando es necesario.

Además de los paneles solares, Parasol puede obtener energía de sus baterías y/o de la red eléctrica. Tres interruptores manuales permiten configuraciones distintas para el suministro de energía. Por ejemplo, se puede configurar Parasol para operar completamente fuera de la red eléctrica. Parasol también incluye una amplia infraestructura de monitorización para cuantificar la cantidad de energía que se extrae de cada fuente disponible y, de este modo, implementar políticas de eficiencia energética. La figura siguiente muestra una fotografía del exterior de Parasol.

Figura 6. Fotografía del exterior del microcentro de datos Parasol en Rutgers, State Univeristy of New Jersey

Fuente: Página web del proyecto Parasol.

Más información

Podéis encontrar más información sobre el proyecto Parasol en la lista de lecturas recomendadas del apartado 4.

4. Lista de lecturas recomendadas

En este apartado, se presenta una lista de lecturas recomendadas para complementar este módulo didáctico. Se trata básicamente de artículos de investigación actuales que cubren la mayoría de los aspectos relacionados con este módulo y que ofrecen oportunidades de profundizar en el tema a partir de una lectura crítica.

4.1. Conceptos generales

U. Hoelzle; L. A. Barroso (2009). *The datacenter as a computer: An introduction to the design of warehouse-scale machines* (1.ª ed.). Morgan and Claypool Publishers.

L. A. Barroso; U. Hoelzle (2007). "The case for energy-proportional computing". *IEEE computer* (vol. 40, núm. 12, págs. 33-371).

T. Scogland; B. Subramaniam; W. Feng (2012). "The Green500 list: escapades to exascale". *Computer science - Research and development* (mayo, págs. 1-9). Springer Verlag.

4.2. Arquitectura y gestión de subsistemas

V. Pallipadi; S. B. Siddha (2007). "Processor power management features and process scheduler: Do we need to tie them together?". *LinuxConf Europe*.

K. Choi; R. Soma; M. Pedram (2004). "Dynamic voltage and frequency scaling based on workload decomposition". *Proceedings of International Symposium on Low Power Electronics and Design 2004* (*ISLPED'04*) (9-11 de agosto, págs. 174-179). Newport Beach.

Hur; C. Lin (2008). "A comprehensive approach to DRAM power management". En: *14th International Conference on High-Performance Computer Architecture* (*HPCA*) (págs. 305-316).

L. Xiaodong; L. Zhenmin; Z. Yuanyuan; A. Sarita (2005, agosto). "Performance directed energy management for main memory and disks". *Trans. Storage* (vol. 1, núm. 3, págs. 346-380).

V. Pallipadi; S. Li; A. Belay (2007). "cpuidle-Do nothing efficiently...". En: *Ottawa Linux Symposium* (*OLS'07*).

S. Siddha; V. Pallipadi; A. van de Ven (2007). "Getting maximum mileage out of tickless". En: *Ottawa Linux Symposium* (*OLS'07*) (págs. 201-208).

V. Delaluz; M. Kandemir; N. Vijaykrishnan; A. Sivasubramaniam; M. J. Irwin (2001). "Hardware and software techniques for controlling DRAM power modes". *IEEE Trans. Comput.* (vol. 50, núm. 11, págs. 1154-1173).

D. Colarelli; D. Grunwald (2002). "Massive arrays of idle disks for storage archives" (págs. 1-11). En: *Proceedings of the 2002 ACM/IEEE conference on Supercomputing* (*Supercomputing'02*). IEEE Computer Society Press: Los Alamitos.

D. Rotem; E. Otoo; S.-C. Tsao (2009). "Analysis of trade-off between power saving and response time in disk storage systems". *Fifth Work- shop on High-Performance Power-Aware Computing (HPPAC'09) with IPDPS'09*.

E. Pinheiro; R. Bianchini; C. Dubnicki (2006). "Exploiting redundancy to conserve energy in storage systems". *SIGMETRICS Perform. Eval. Rev.* (vol. 34, núm. 1, págs. 15-26).

4.3. Entornos de computación de altas prestaciones

N. Kappiah; V. W. Freeh; D. K. Lowenthal (2005). "Just in time dynamic voltage scaling: exploiting inter-node slack to save energy in MPI programs". En: *ACM/IEEE conference on Supercomputing* (*SC'05*) (pág. 33).

B. Rountree, D. K. Lownenthal, B. R. de Supinski, M. Schulz, V. W. Freeh; T. Bletsch (2009). "Adagio: making DVS practical for complex HPC applications". En: *23rd international conference on Supercomputing* (*ICS'09*) (págs. 460-469).

W. Freeh; F. Pan; N. Kappiah; D. K. Lowenthal; R. Springer (2005). "Exploring the energy-time tradeoff in MPI programs on a power- scalable cluster". En: *19th IEEE International Parallel and Distributed Processing Symposium* (*IPDPS'05*) (pág. 4).

K. W. Cameron; R. Ge; X. Feng (2005). "High-performance, power-aware distributed computing for scientific applications". *Computer* (vol. 38, núm. 11, págs. 40-47).

M. Y. Lim; V. W. Freeh; D. K. Lowenthal (2006). "Adaptive, transparent frequency and voltage scaling of communication phases in MPI programs". En: *ACM/IEEE conference on Supercomputing* (*SC'06*) (pág. 107).

B. Rountree; D. K. Lowenthal; S. Funk; V. W. Freeh; B. R. de Supinski; M. Schulz (2007). "Bounding energy consumption in large-scale MPI programs". En: *ACM/IEEE conference on Supercomputing* (*SC'07*) (págs. 1-9).

V. W. Freeh; D. K. Lowenthal (2005). "Using multiple energy gears in MPI programs on a power-scalable cluster". En: *ACM SIGPLAN symposium on Principles and Practice of Parallel Programming* (*PPoPP'05*) (págs. 164-173).

4.4. Centros de datos

J. Moore; J. Chase; P. Raanganathan (2006, junio). "Weatherman: Automated, online, and predictive thermal mapping and management for data centers". En: *Proceedings of the Third IEEE International Conference on Autonomic Computing*.

Verma; P. Ahuja; A. Neogi (2008). "pMapper: power and migration cost aware application placement in virtualized systems". En: *9th ACM/IFIP/USENIX International Conference on Middleware* (*Middle-ware'08*) (págs. 243-264).

R. Nathuji; K. Schwan (2007). "VirtualPower: coordinated power management in virtualized enterprise systems". En: *ACM SIGOPS Symposium on Operating Systems Principles* (*SOSP'07*) (págs. 265-278).

Qureshi; R. Weber; H. Balakrishnan; J. Guttag; B. Maggs (2009). "Cutting the electric bill for internet-scale systems". SIGCOMM Comput. Commun. Rev. (vol. 39, núm. 4, págs. 123-134). ACM: Nueva York.

J. Moore; J. Chase; P. Ranganathan (2005, abril). "Making scheduling «cool»: Temperature-aware workload placement in data centers". En: *Proc. of the 2005 USENIX Annual Technical Conference* (*USENIX'05*).

X. Fan; W. Weber; L.A. Barroso (2007). "Power provisioning for a warehouse-sized computer". ISCA.

4.5. Sostenibilidad

C. Li; W. Zhang; C. Cho; T Li (2011). "SolarCore: Solar energy driven multi-core architecture power management". En: *Proceedings of the 2011 IEEE 17th International Symposium on High Performance Computer Architecture* (*HPCA'11*). *IEEE computer society*. Washington (págs. 205-216).

Z. Liu; M. Lin; A. Wierman; S. H. Low; L. H. Andrew (2011, diciembre). "Geographical load balancing with renewables". *SIGMETRICS Perform. Eval. Rev*. (vol. 3, núm. 39, págs. 62-66).

I. Goiri; W. Katsak; K. Le; T. D. Nguyen; R. Bianchini (2013). "Parasol and GreenSwitch: Managing datacenters powered by renewable energy". *18th International Conference on Architectural Support for Programming Languages and Operating Systems* (*ASPLOS 2013*). Houston (16-20 de marzo).

N. Sharma; J. Gummeson; D. Irwin; P. Shenoy (2010). "Cloudy computing: Leveraging weather forecasts in energy harvesting sensor systems". *Sensor Mesh and Ad Hoc Communications and Networks* (*SECON*). *7th Annual IEEE Communications Society Conference on* (págs. 1-9, 21-25 de junio).

C. Stewartand; K. Shen (2009, octubre). "Some joules are more precious than others: Managing renewable energy in the datacenter". En: *Proceedings of the Workshop on Power Aware Computing and Systems*.

K. Ley; R. Bianchiniy; M. Martonosi; T. D. Nguyeny (2009). "Cost and energy-aware load distribution across data centers". *HotPower'09*.

Bibliografía

Advanced Configuration and Power Interface Specification, Intel, Microsoft, Toshiba, Revision 1.0 (22 de diciembre de 1996).

Barroso, L.A; Holzle, U. (2007). "The case for energy-proportional computing". *IEEE computer* (vol. 40, núm. 12, págs. 33-371).

Brown, L.; Keshavamurthy, A.; Shaohua Li, D.; Moore, R.; Pallipadi, V.; Yu, L. (2005). "ACPI in Linux. Intel open source technology center". *Proceedings of the Linux Symposium*.

Chari, S. (2011, junio). "IBM Blue Gene/Q: The most energy efficient solution for high performance computing, a total cost of ownership (TCO) study comparing the IBM Blue Gene/Q with traditional x86 based cluster systems including systems with graphics processing units (GPUs)". *Cabot Partners*.

Hoelzle, U.; Barroso, L.A. (2009). *The datacenter as a computer: An introduction to the design of warehouse-scale machines* (1.ª ed.). Morgan and Claypool Publishers.

Kappiah, N.; Freeh, V.W.; Lowenthal, D.K. (2005). "Just in time dynamic voltage scaling: exploiting inter-node slack to save energy in MPI programs". En: *ACM/IEEE conference on supercomputing* (*SC'05*) (pág. 33).

Lim, M.Y.; Freeh, V.W.; Lowenthal, D.K. (2006). "Adaptive, transparent frequency and voltage scaling of communication phases in MPI programs". En: *ACM/IEEE conference on supercomputing* (*SC'06*) (pág. 107).

Rountree, B.; Lownenthal, D.K.; Supinski, B.R.; Schulz, M.; Freeh, V.W.; Bletsch, T. (2009). "Adagio: making DVS practical for complex HPC applications". En: *23rd international conference on supercomputing* (*ICS'09*) (págs. 460-469).

Pallipadi, V.; Li, S.; Belay, A. (2007). "cpuidle - Do nothing efficiently...". En: *Ottawa Linux Symposium* (*OLS'07*).

Rountree, B.; Lowenthal, D.K.; Funk, S.; Freeh, V.W.; Supinski, B.R. de; Schulz, M. (2007). "Bounding energy consumption in large-scale MPI programs". En: *ACM/IEEE conference on supercomputing* (*SC'07*) (págs. 1-9).

Scogland, T.; Subramaniam, B.; Feng, W. (2012, mayo). "The Green500 list: escapades to exascale" (págs. 1-9). *Computer science - Research and development*. Springer Verlag.