Tendencias y Aplicaciones en Supercomputación

Enrique S. Quintana-Ortí

HPC&A
High Performance Computing and Architectures

Índice

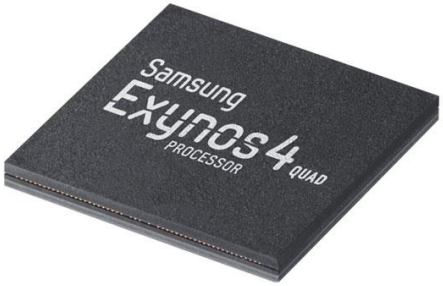
- Estado actual de la arquitectura de computadores
- Aplicaciones en HPCA@UJI
- Líneas de investigación en HPCA@UJI

Tecnologías de Supercomputación

- ¿Arquitecturas paralelas?

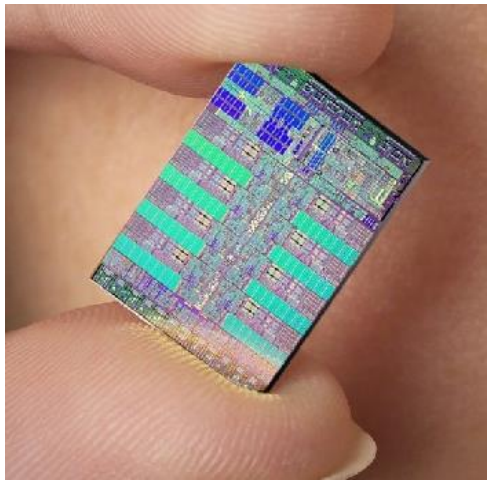
Tecnologías de Supercomputación

- Arquitecturas paralelas



Tecnologías de Supercomputación

- Arquitecturas paralelas

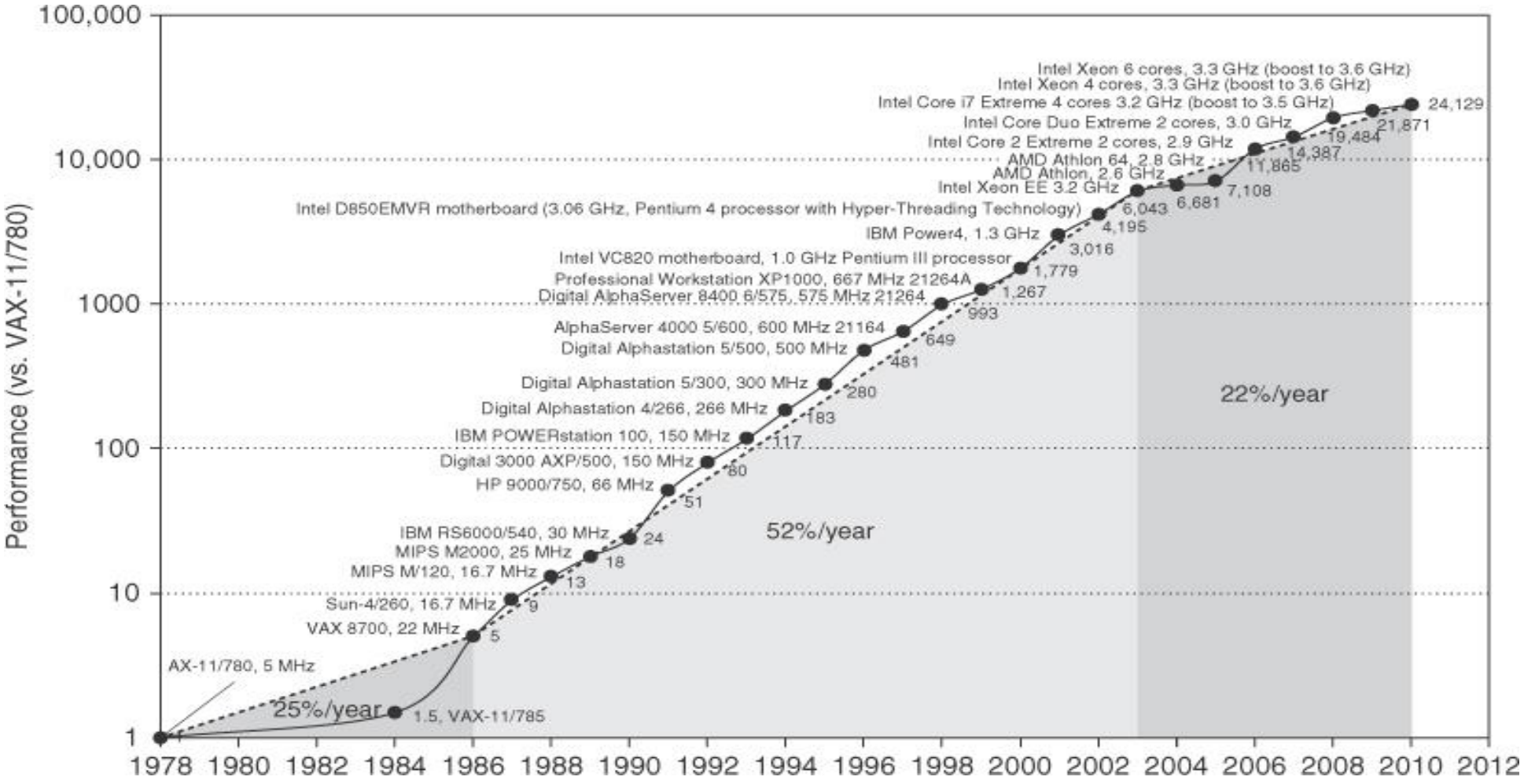


¡Presentación centrada en el procesador!

Evolución del Procesador

- "The attack of the killer micros" (E. Brooks, 1989)
  o "Cray on a chip" (Intel i860)
  - Arquitectura RISC
  - Tecnología CMOS
  - Cauce segmentados y superescalares
  - Alta frecuencia
  - Cachés grandes con varios niveles
  - Unidades FP SIMD

Evolución del Procesador



Evolución del Procesador

- "The free lunch is over" (H. Sutter, 2005)
  - Fin de la carrera de los megahercios:
    - Consumo es proporcional a f^3
    - La energía consumida es calor a disipar
  - No más paralelismo a nivel de instrucción
    - Media de 1 salto cada 5 instrucciones
  - Latencia de acceso a memoria sin cambios
    - 1 acceso a memoria ≈ 240 ciclos (2008)

Evolución del Procesador

- ¡...pero la Ley de Moore sigue siendo válida!

"Cramming more components onto integrated circuits", G. E. Moore, 1965:
The complexity for minimum component costs has increased at a rate of roughly a factor of two per year ... Certainly over the short term this rate can be expected to continue, if not to increase. Over the longer term, the rate of increase is a bit more uncertain, although there is no reason to believe it will not remain nearly constant for at least 10 years. That means by 1975, the number of components per integrated circuit for minimum cost will be 65,000. I believe that such a large circuit can be built on a single wafer.

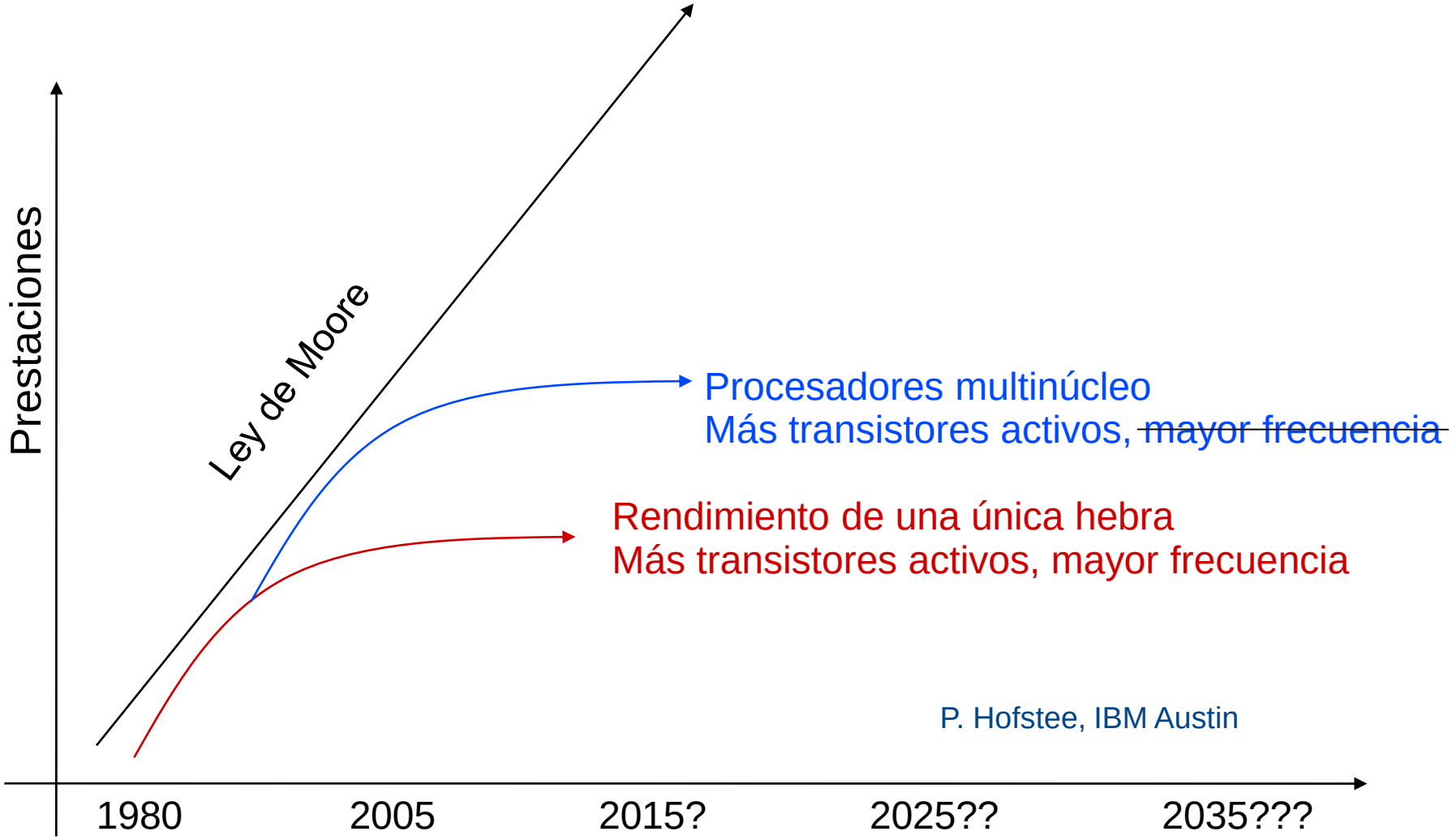
≡ El número de transistores que pueden integrarse en un dispositivo se dobla cada 1,5 años, manteniéndose constante el coste económico

Evolución del Procesador

- En la actualidad, 22nm… en 2028, 1nm
- ¿Qué hacemos con los transistores disponibles?
  - No podemos reducir el tamaño para subir la frecuencia
  - No podemos aumentar la complejidad del procesador pues no hay más ILP
  - No podemos hacer cachés más grandes porque la latencia no mejora

¿Cómo seguir vendiendo microprocesadores?

Evolución del Procesador



Procesadores Multinúcleo

- Intel

Procesador	#núcleos	f (GHz)	L3 (MB)	Tecnología (nm)	Consumo (W)
E7-8870	10	2,4	30	32	130
E5-4650L	8	2,6	20	32	115
E3-1290	4	3,6	8	32	95

- AMD

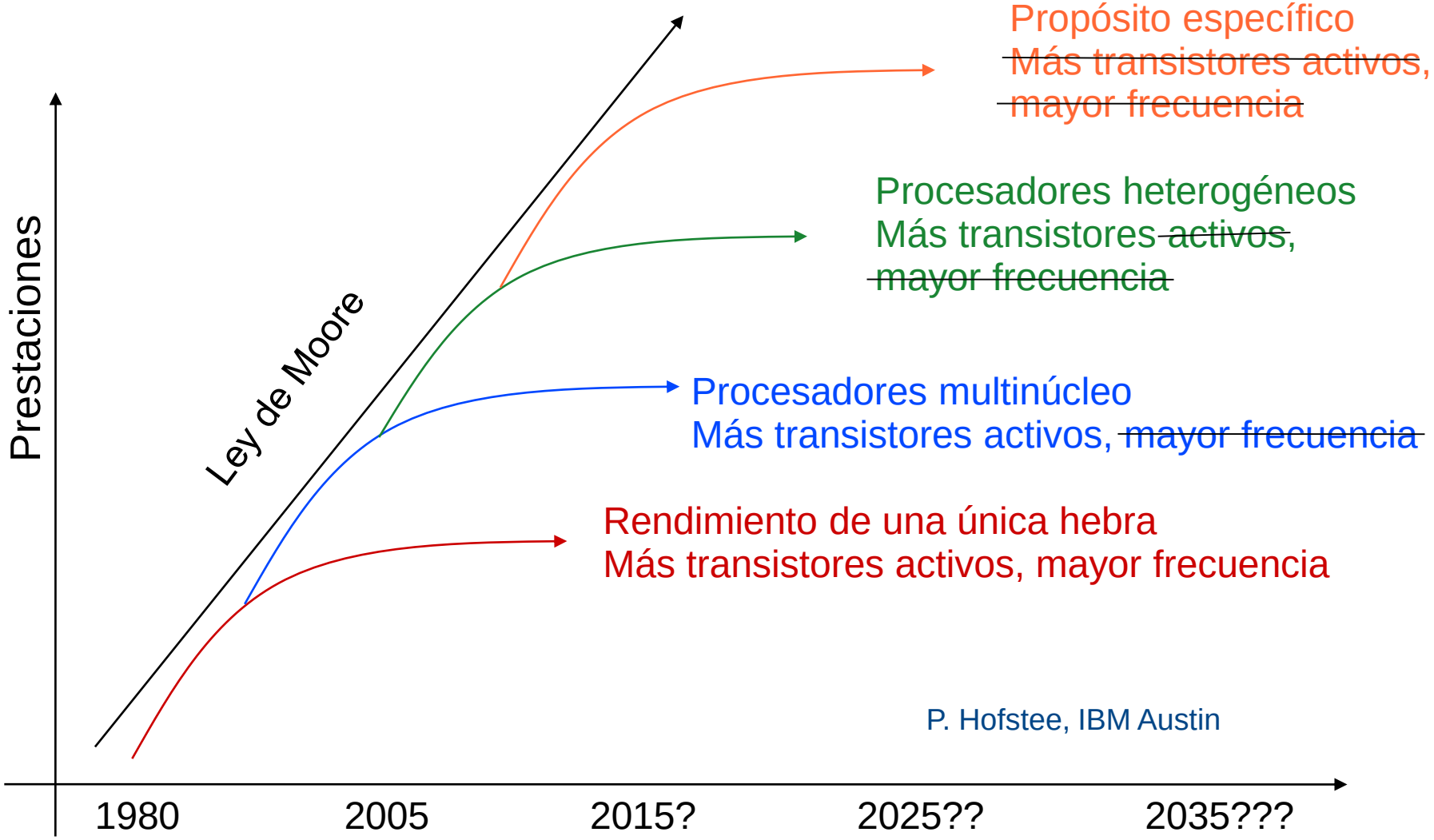
Procesador	#núcleos	f (GHz)	L3 (MB)	Tecnología (nm)	Consumo (W)
6386 SE	16	2,8	16	45	140
6220	8	2,6	16	45	115
6132 HE	8	2,2	12	45	85

Procesadores Multinúcleo y…

“The HIPEAC vision for advanced computing in Horizon 2020", M. Duranton et al., 2013:

> Yet even the shift to universal parallelism is not enough. The increasing number of components on a chip, combined with decreasing energy scaling, is leading to the phenomenon of “dark silicon”, whereby the chip’s power density is too high to use all components at once. This puts an even greater emphasis on efficiency, and is driving chips to use multiple different components, each carefully optimized to efficiently execute a particular type of task. This era of heterogeneous parallel computing presents an even greater challenge for software developers.

Procesadores Multinúcleo y…



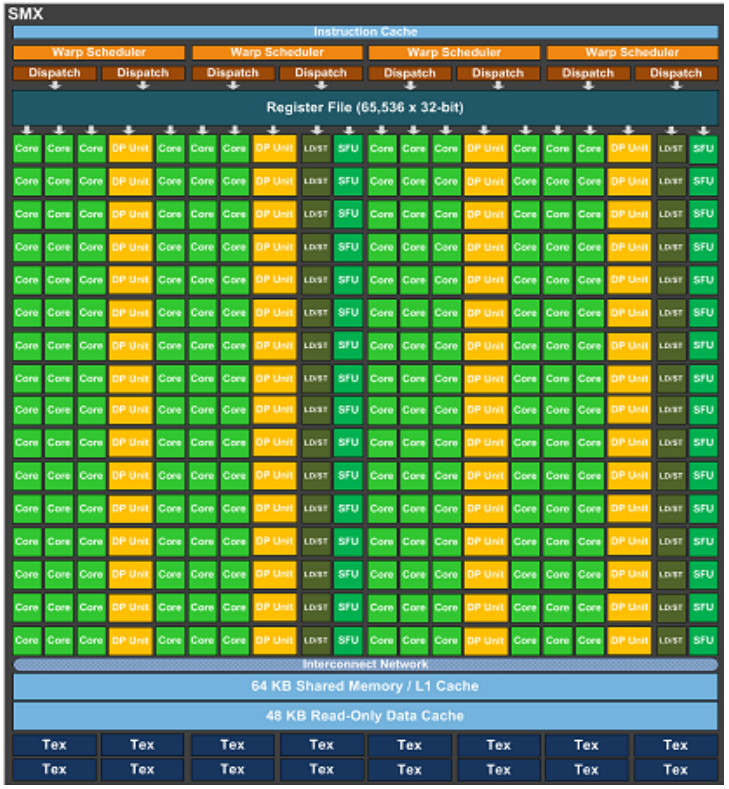
Procesadores Multinúcleo y…

- Aceleradores hardware (many-core):
  - NVIDIA GPUs
  - AMD FireStream y APUs
  - Intel Xeon Phi

¡La fabricación de tecnología de supercomputación no es un mercado económicamente viable por sí solo!

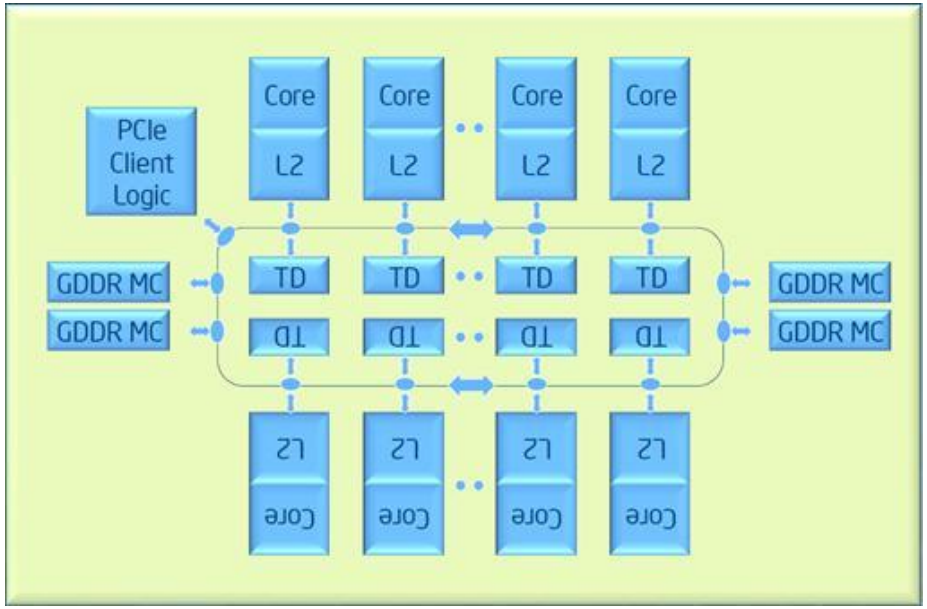
Procesadores Multinúcleo y…

- NVIDIA Kepler GK110
  - 13-15 streaming multiprocessors (SMX units)
  - 192 SP cores por cada SMX
  - 1.536 KB L2 cache



Procesadores Multinúcleo y…

- Intel Xeon Phi 510P
  - Hasta 61 cores y 240 threads con ejecución en orden
  - ISA x86 con SIMD (512 bits)
  - Coherencia Hw. de cachés
  - Anillo bidireccional de interconexión



Comparativa

- Multicore

Procesador	#núcleos	f (GHz)	L2/L3 (MB)	Tecnología (nm)	Consumo (W)
Intel E7-8870	10	2,4	30	32	130
AMD 6386 SE	16	2,8	16	45	140

- Aceleradores

Procesador	#núcleos	f (GHz)	Memory (GB)	Tecnología (nm)	Consumo (W)
NVIDIA GK110	2.496	0,7	5	28	122
Intel Xeon Phi 7120P	61	1,2	16	22	300

Comparativa

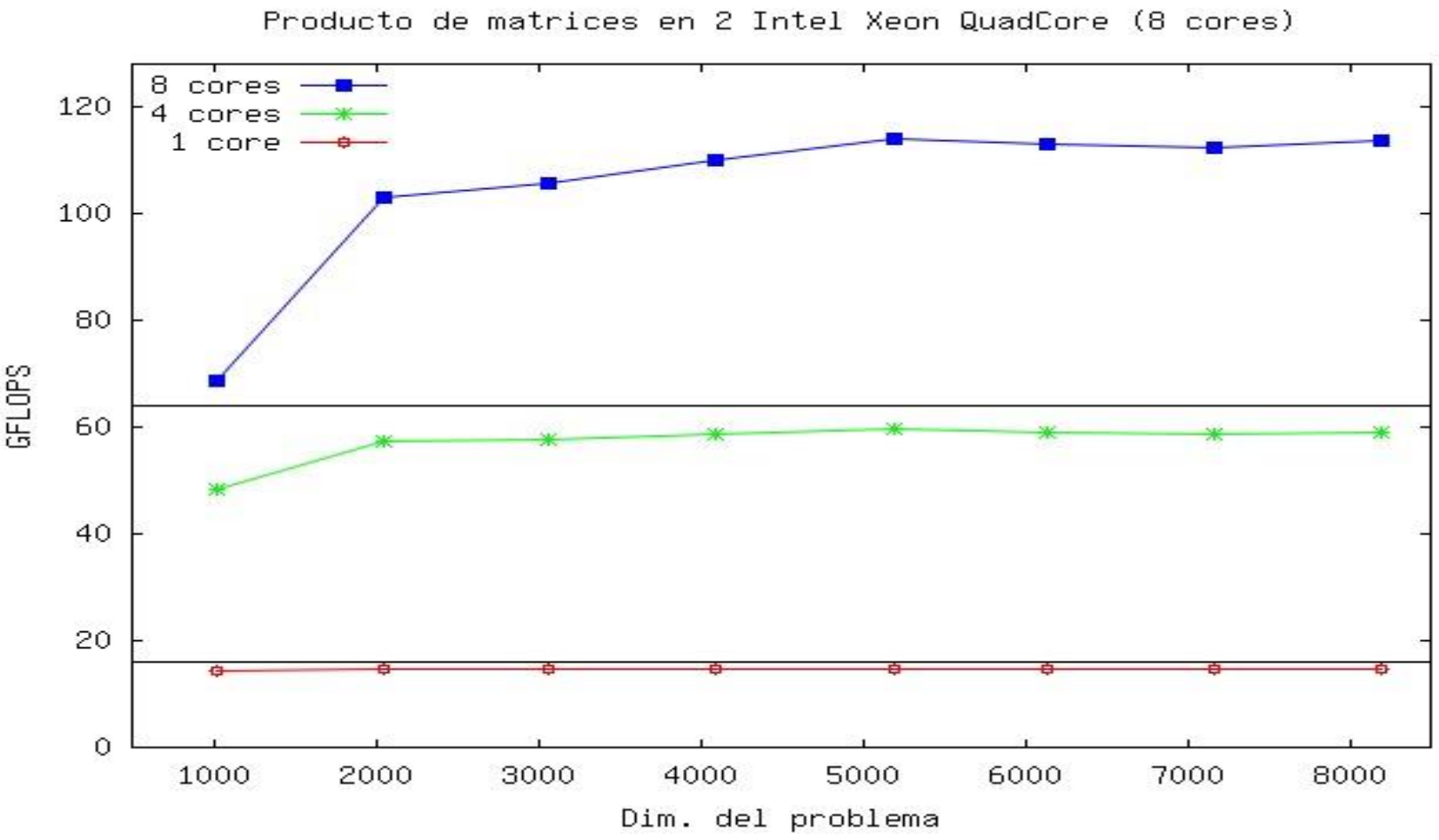
- ¿Cuál es factor que determina el rendimiento?

Procesador / Acelerador	f (GHz)	#núcleos	Velocidad pico SP (TFLOPS)
Intel E7-8870	2,4	10	0,38
NVIDIA GK110	0,7	2.496	3,52
Intel Xeon Phi 7120P	1,2	61	2,44

La velocidad real depende fuertemente
de la aplicación y de su tamaño
Ratio: velocidad-consumo/coste/programabilidad

Comparativa

- Velocidad real: `sgemm` de Intel MKL 10.0



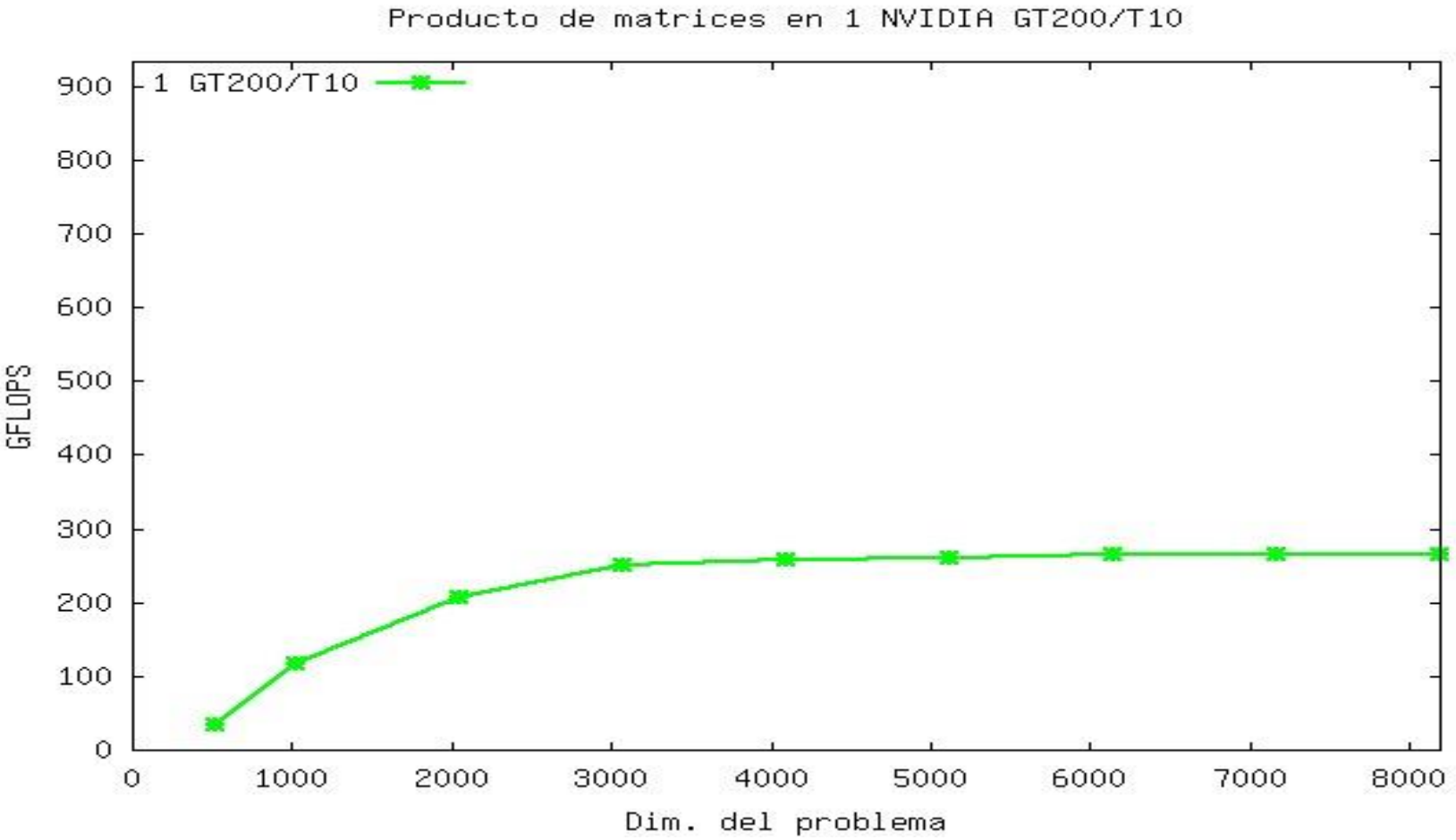
89% pico

92% pico

92% pico

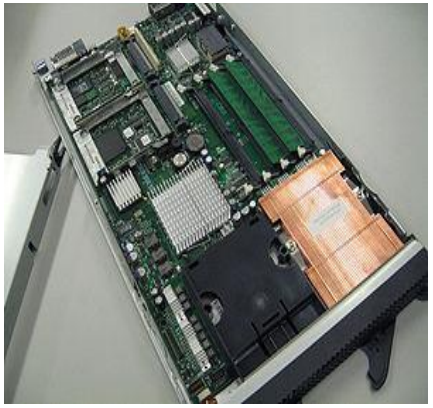
Comparativa

- Velocidad real: `sgemm` de NVIDIA CUBLAS 2.0



28% pico

Supercomputadores



- Los supercomputadores se construyen “fácilmente” a partir de procesadores y una “buena” red
- “The Beowulf cluster” (D. Becker, T. Sterling, 1994):
  - Mejor ratio precio-prestaciones
- Actualmente se construyen a partir de tecnologías estándar
  - La supercomputación no es un mercado económicamente viable por sí solo

Supercomputadores: Top500 (noviembre 2013)

Puesto	Sistema	#Núcleos	Procesador/Red	LINPACK (TFLOPS)
1	Tianhe-2 National Supercomputer Center Guangzhou	3.120.000	Intel Xeon E5, Intel Xeon Phi/ Infiniband	33.862*
2	Titan DOE/SC/ORNL	560.640	Opteron 6274, NVIDIA K20x/ Cray Gemini	17.590
3	Sequoia DOE/NNSA/LLNL	1.572.864	Blue Gene/Q / IBM	17.173

*1 día Tianhe-2 = 1.405 años (24 x 7) de la población mundial (6.000 millones) equipados con calculadora

Supercomputadores: Top500 por segmentos

Segmento	#Sistemas	%
Industria	282	56,4
Investigación	103	20,6
Académico	83	16,6
Gobierno	16	3,2
Vendedores	12	2,4
Clasificado	3	0,6
Otros	1	0,2

- Total (15º, 110.400 cores)
- EDF (46º, 65.536 cores)
- Amazon (65º, 26496 cores)

Supercomputadores: Top500 por fabricantes

Familia de procesadores	#Sistemas	%
Intel Xeon E5 (SandyBridge)	307	61,3
Intel Xeon 5600 (Westmere-EP)	55	11,0
Intel Xeon E5 (Ivy Bridge)	34	6,8
IBM BQC	24	4,8
AMD Opteron 6100 (Magny-Cours)	17	3,4
AMD Opteron 6200 (Interlagos)	16	3,2

Supercomputadores: Top500 por fabricantes

Interconexión	#Sistemas	%
Infiniband	207	41,4
Gigabit Ethernet	135	27,0
10 Gigabit Ethernet	77	15,4
A medida	50	10,0
Otros (Cray, SP, NUMAlink,…)	31	6,2

Supercomputadores: Top500 en España

Puesto	Sistema	#Núcleos	Red	LINPACK (TFLOPS)
34	BSC MareNostrum – Intel Xeon E5	48.896	Infiniband FDR	925
139	ITER – Intel Xeon E5	16.384	Infiniband QDR	273

Aplicaciones para Supercomputadores

Aplicaciones para Supercomputadores

- DOE/LANL Roadrunner
  - Kinetic Thermonuclear Burn Studies with VPIC
  - Multibillion-Atom Molecular Dynamics Simulations of Ejecta Production and Transport
  - Saturation of Backward Stimulated Scattering of Laser In The Collisional Regime
  - Instabilities Driven Reacting Compressible Turbulence
  - Three-Dimensional Dynamics of Magnetic Reconnection in Space and Laboratory Plasmas
  - Cellulosomes in Action: Peta-Scale Atomistic Bioenergy Simulations

Aplicaciones para Supercomputadores

- Repsol YPF - BSC

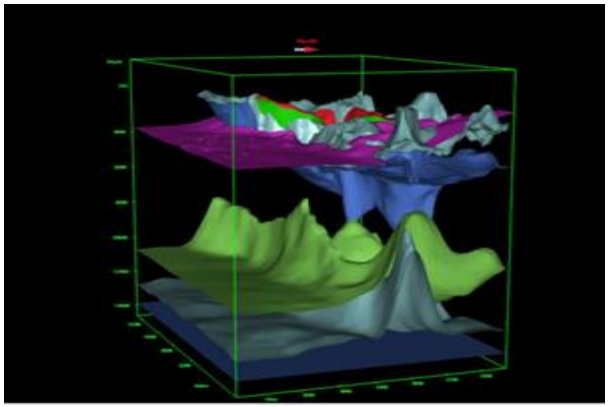
Aplicaciones para Supercomputadores

- Las reservas de petróleo cada vez están en lugares más inaccesibles
- Perforar un pozo puede costar 40 millones de euros

¡...como para equivocarse de sitio!

Aplicaciones para Supercomputadores

- La decisión de dónde perforar se basa en el análisis sísmico: ondas de choque (explosiones, vibraciones) que exploran el subsuelo
- La precisión de los resultados es proporcional a la cantidad de datos a procesar



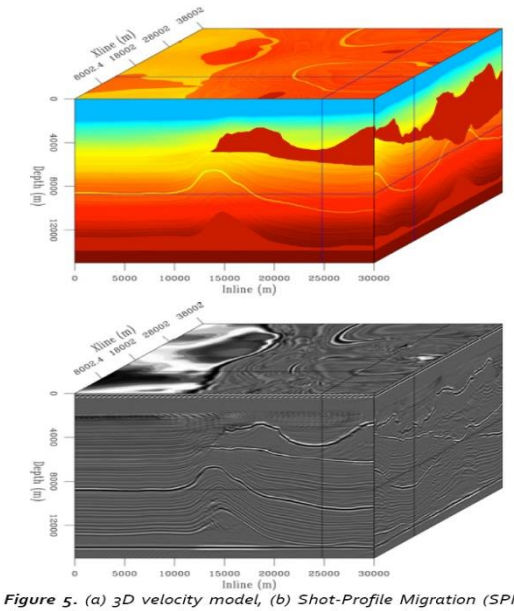


Figure 5. (a) 3D velocity model, (b) Shot-Profile Migration (SPM) of the modeled data with 4,047 shot gathers and limited cross-line apertura. AGC has been applied to the image

Aplicaciones para Supercomputadores

- Repsol YPF – Houston:
  - Cluster BladeCenter QS22
  - Nodos PowerXCell 8i 3.2 Ghz
  - 1800 núcleos
  - 26.38 TFLOPS (puestos 429-431 Top500-Nov. 2008)
  - Red Infiniband
- BSC MareNostrum (2008)
  - 5120 procesadores PowerPC de doble núcleo
  - 63,83 TFLOPS

Sumario

- El futuro es paralelo, heterogéneo y especializado, a menos que se descubra nueva tecnología (nanotubos, computación cuántica)
- La supercomputación avanza gracias a los desarrollos de otros mercados (juegos, PCs, sistemas empotrados)
- La programación paralela es una de las claves y queda mucho por hacer…

Índice

- Estado actual de la arquitectura de computadores
- Aplicaciones en HPCA@UJI
  - HPC4NGS
  - Simulación macromolecular
  - Imágenes hiperespectrales
  - Reducción de modelos
  - ILUPACK
- Líneas de investigación en HPCA@UJI

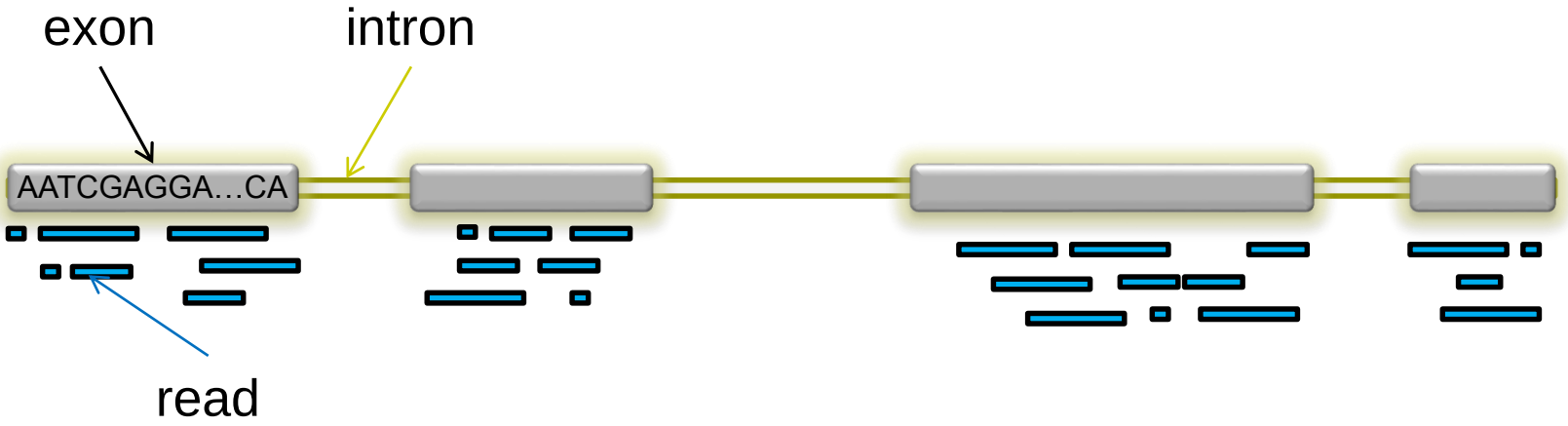
HPC4NGS: RNA-SEQ

- Ana Conesa
- Joaquín Dopazo
- Ignacio Medina
- Joaquín Tárraga

- Sergio Barrachina
- Maribel Castillo
- Héctor Martínez
- Enrique S. Quintana-Ortí

HPC4NGS: RNA-SEQ

- Dependiendo de la tecnología, Next Generation Sequencing (NGS) proporciona millones de lecturas cortas (“reads”) con 35-150 (Illumina), 50-100 (SOLiD) o 400 (Roche) nts
- Después de secuenciarlos, los fragmentos deben mapearse contra un genoma de referencia



- Objetivo: obtener un mapa de tránscritos

HPC4NGS: RNA-SEQ

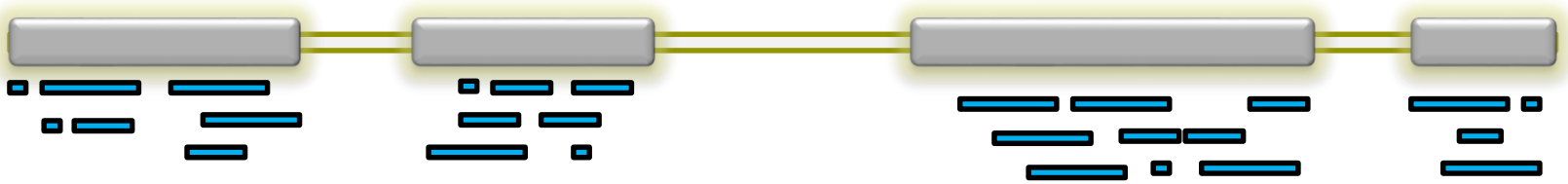
- El mapeado ofrece información sobre:
  - Expresión de los genes, tránscritos, exons
  - Variaciones en los nucleótidos
  - Fusión de genes
  - …

HPC4NGS: RNA-SEQ

- El mapeado es un proceso biológico complejo:
  - Número de “mismatches” (EIDs)
  - Número de multi-mapeos
  - Uniones de exones (splice junctions)
  - Distancia entre exones
  - …

HPC4NGS: RNA-SEQ

- El mapeado es un proceso informático complejo:
  - Gran cantidad de datos (out-of-core)
  - Etapas de cálculo intensivo, acceso a memoria intensivo
  - Esquemas híbridos de concurrencia: paralelismo de datos, tareas, bucles, SIMD, etc.
  - …



HPC4NGS: RNA-SEQ

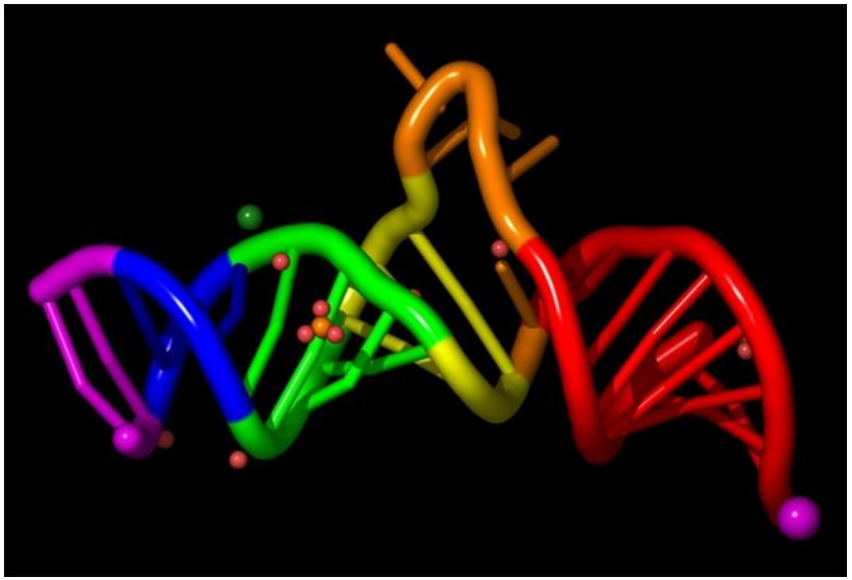
- Estado del arte:
  - Tophat 2 (+Bowtie 2)
  - Mapsplice
  - SpliceMap
  - RUM
  - STAR
  - y docenas de otros (BLASTN, Bowtie, ELAND, QPALMA…)

“Those are my principles, and if you don't like them... well, I have others.”

-- Groucho Marx

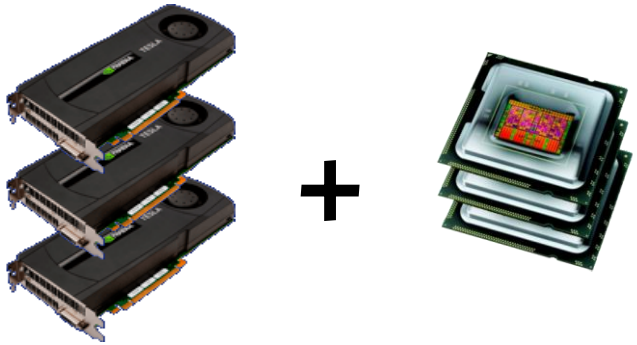
HPC4NGS: RNA-SEQ

- HPG Aligner W:
  - Mapear lecturas cortas contra un genoma de referencia con “splice junctions”
  - Alta fiabilidad
  - Velocidad



HPC4NGS: RNA-SEQ

- HPG Aligner W. Velocidad:
  - Dividir el problema en etapas/tareas
  - Mapear las tareas sobre diferentes recursos computacionales
  - Computación CPU-GPU



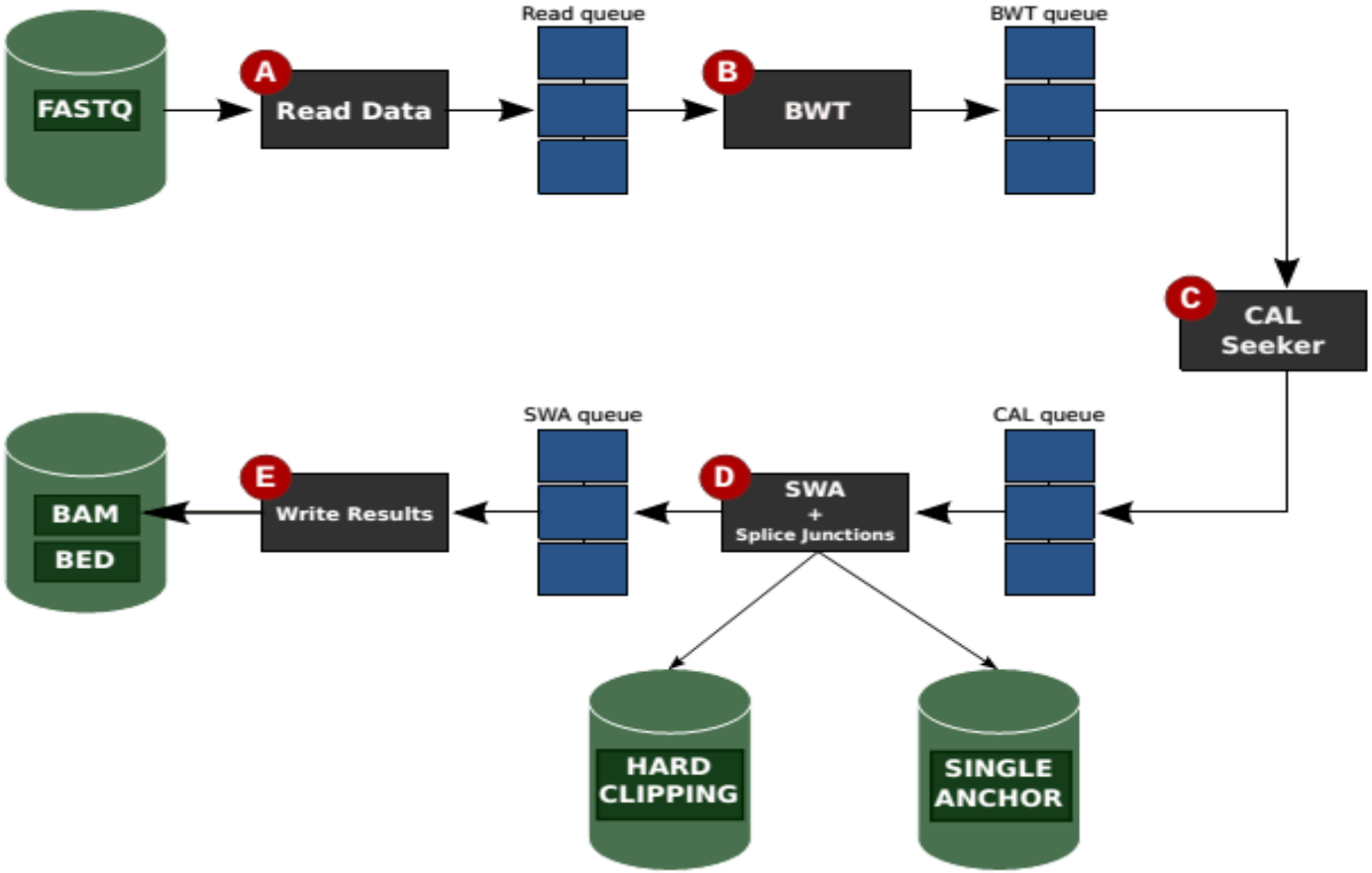
Need for Speed!

HPC4NGS: RNA-SEQ

- HPG Aligner W… pero no olvidar el movimiento de datos

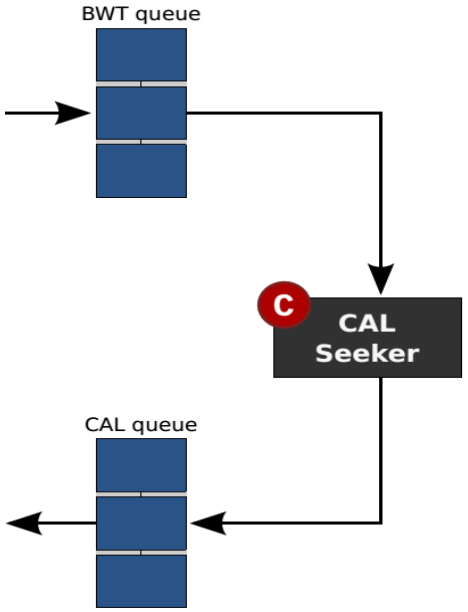
HPC4NGS: RNA-SEQ

- HPG Aligner W



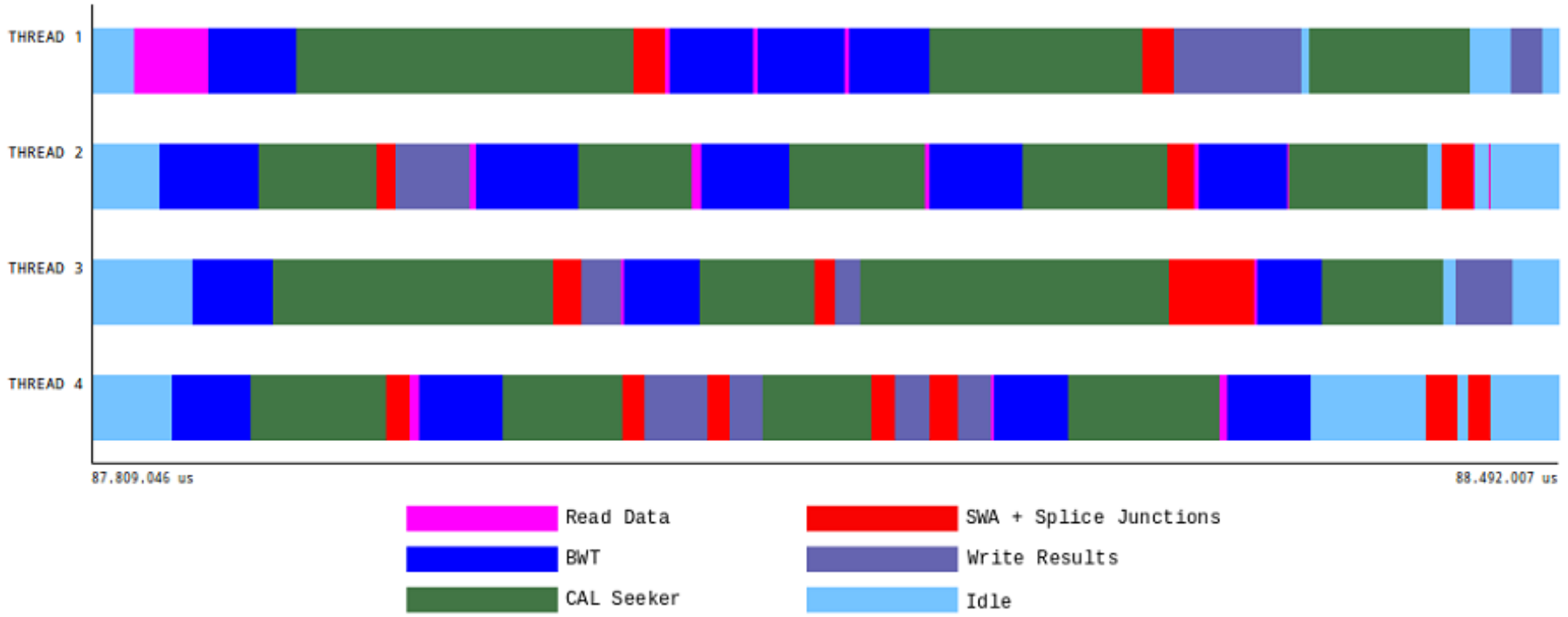
HPC4NGS: RNA-SEQ

- HPG Aligner W:
  - Extracción de paralelismo a nivel de tarea (batch de lecturas)
  - “Pool” de threads activos, capaces de ejecutar cualquier tipo de tarea
  - Sincronización a través de colas compartidas



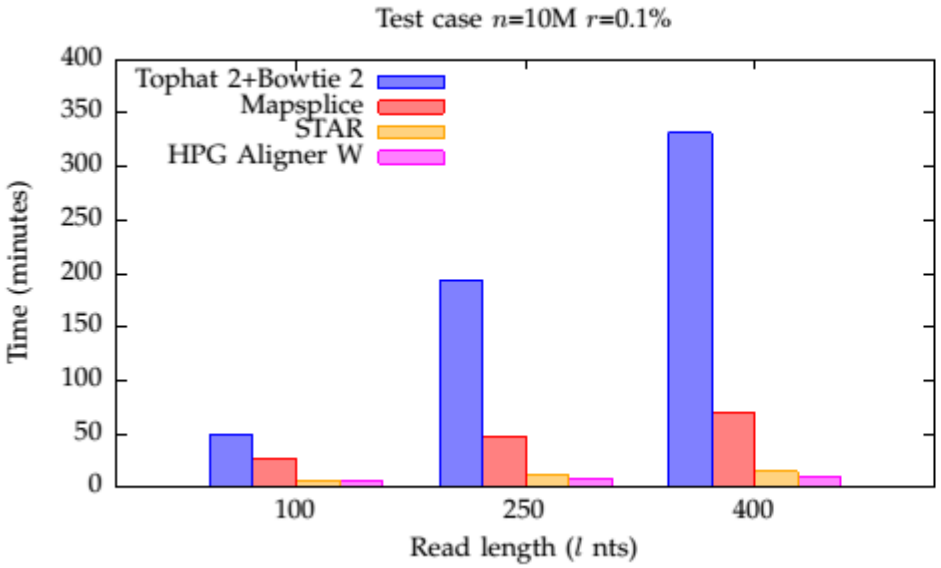
HPC4NGS: RNA-SEQ

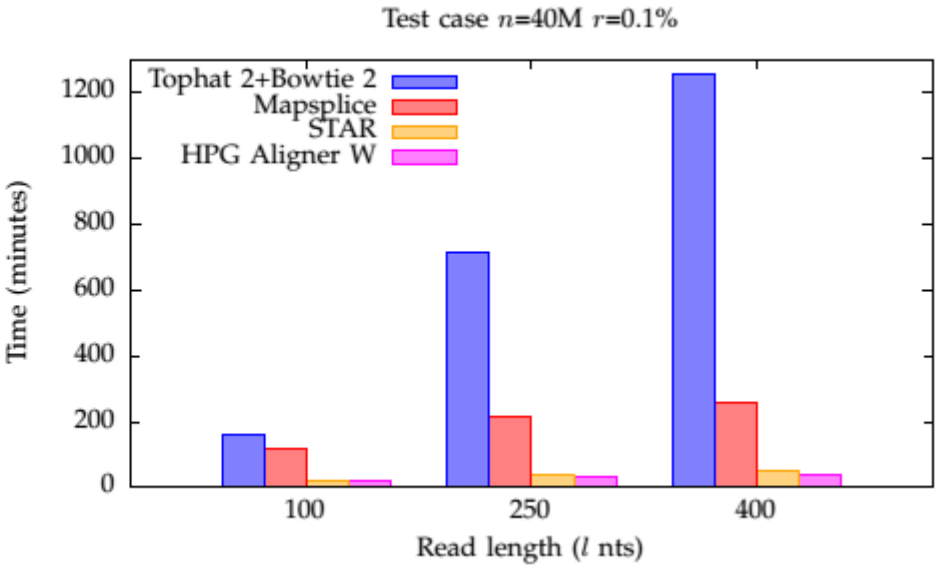
- Procesamiento superescalar



HPC4NGS: RNA-SEQ

- Velocidad: comparable a STAR y superior a Mapsplice y Tophat 2





HPC4NGS: RNA-SEQ

- Sensibilidad: superior a otros mapeadores

			HPG Aligner W			STAR			MapSplice			TopHat 2+Bowtie 2		
n	r	l	NMR	RCS	Time	NMR	RCS	Time	NMR	RCS	Time	NMR	RCS	Time
10M	0.01	100	0.37	95.57	5.8	0.98	87.18	8.0	0.53	92.37	26.0	35.05	64.20	50.0
		250	0.60	95.42	8.6	0.47	86.70	12.6	0.21	62.25	46.5	88.53	11.40	163.8
		400	1.05	94.96	10.4	1.25	84.12	16.2	0.03	42.84	74.0	98.79	1.19	313.4
	0.1	100	0.40	95.50	5.9	0.98	87.08	7.0	0.58	91.94	27.7	36.71	62.57	49.5
		250	0.70	95.21	8.8	0.63	87.99	11.1	0.26	57.25	47.5	89.67	10.25	193.0
		400	1.24	94.79	10.4	1.57	84.13	15.2	0.04	43.61	69.6	99.00	0.98	331.4
	1	100	0.78	94.43	6.0	2.24	85.09	7.8	1.33	86.92	27.3	52.44	46.99	53.2
		250	1.73	93.47	9.0	2.09	84.15	11.1	0.32	52.66	48.9	96.55	3.41	206.3
		400	3.47	91.74	12.0	6.50	75.16	15.2	0.05	42.54	70.9	99.87	0.13	333.4

Simulación Macromolecular

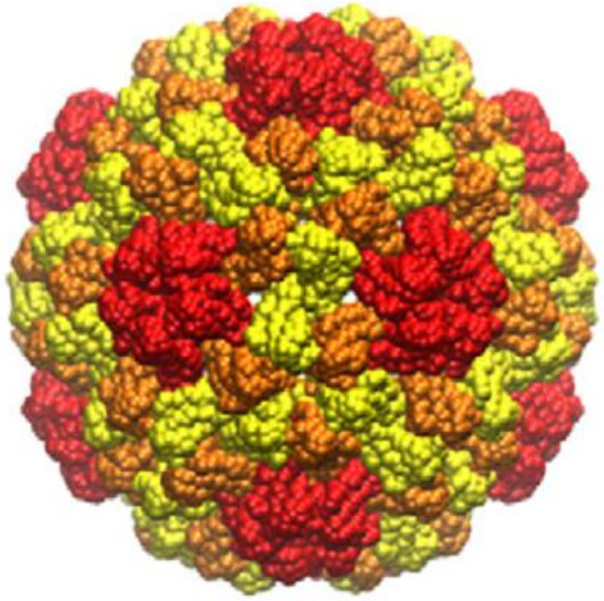
- José R. López Blanco
- Pablo Chacón

- Rosa M. Badia

- José I. Aliaga
- Enrique S. Quintana-Ortí
- Ruymán Reyes

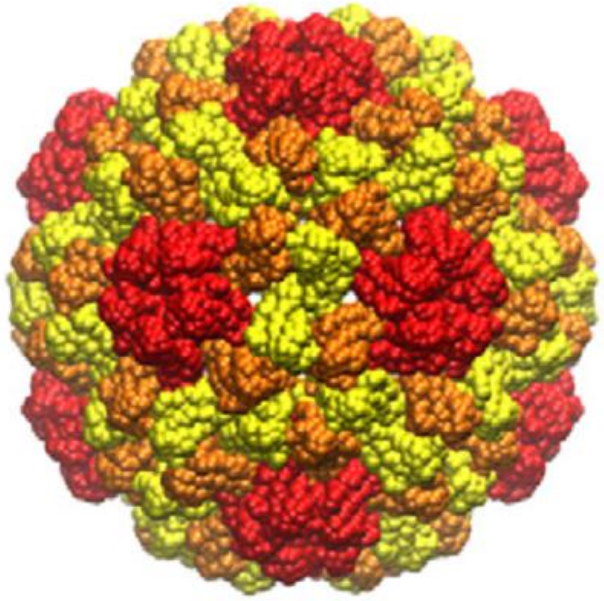
Simulación Macromolecular

- Los componentes más importantes de las células vivas son proteínas y ácidos nucleicos (AC), formadas de cadenas de aminoácidos y nucleótidos
- Las proteínas y AC se ensamblan en grandes macromoléculas, con importantes funciones biológicas



Simulación Macromolecular

- A nivel molecular, la actividad biológica de estos componentes se estudia mediante el análisis de su dinámica e interacciones
- Desgraciadamente, el tamaño y la escala temporal de los movimientos hace estas simulaciones demasiados costosas



Simulación Macromolecular

- En los últimos años, el uso de modelos de grano grueso (coarse-grain, CG), junto con análisis modal normal (NMA) ha permitido la simulación de macromoléculas de dimensión moderada
- La clave es la resolución de un problema generalizado simétrico definido de valores propios:

$$HX = \Lambda TX$$

Simulación Macromolecular

- La resolución de problemas generalizados de valores propios es clásico en álgebra lineal, y se conocen numerosos métodos:
  - Subespacios de Krylov
  - Reducción a forma tridiagonal + algoritmo iterativo QR
  - División espectral (función signo, descomposición polar, etc.)
  - ...

Simulación Macromolecular

- La resolución de problemas generalizados de valores propios es clásico en álgebra lineal, y se conocen numerosos métodos:
  - Subespacios de Krylov
  - Reducción a forma tridiagonal + algoritmo iterativo QR
  - División espectral (función signo, descomposición polar, etc.)
  - ...

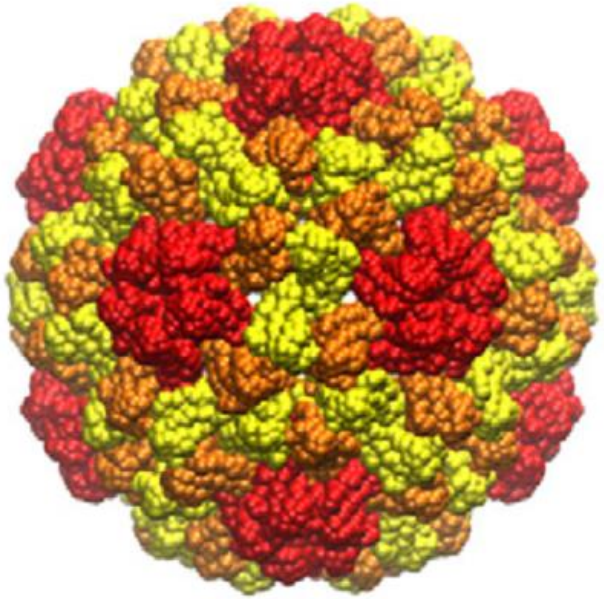
- …pero en todos los casos el coste computacional es cúbico en la dimensión del problema (10.000-300.000)
  - 300.000 $\rightarrow 27 \cdot 10^{15}$ flops,
  - Intel core 2.0 GHz $\rightarrow 16 \cdot 10^{9}$ flops/seg.

¡Casi 20 días!

Simulación Macromolecular

- Subespacios de Krylov en clusters de computadores

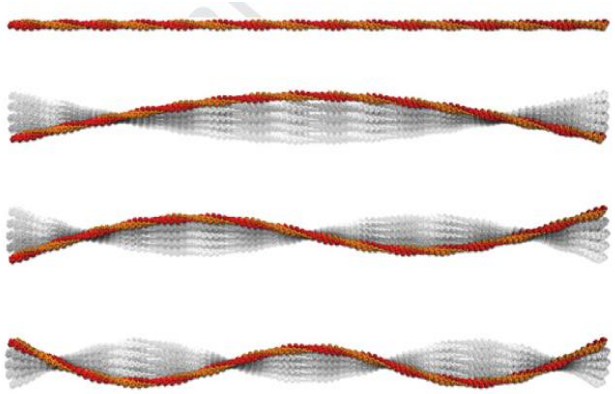
Acronym	DOFs	Time
GroEL/ES	15,870	16.21
ER	36,488	57.65
CCMV	56,454	124.64
HBV	66,234	172.55
VP	119,486	723.39
F-actin	141,297	1,027.81
MT	148,111	1,267.07
HK97	149,231	1,306.95
NwV	149,506	1,315.61



Simulación Macromolecular

- Reducción a forma tridiagonal + algoritmos iterativo QR y descomposición polar en plataformas CPU-GPU con almacenamiento en disco (escala:25.000-31.000 DOFs aprox.)

Case	Two-stage	SD&C-A			SD&C-B			SD&C-C		
	Time	Time	#iter	split	Time	#iter	split	Time	#iter	split
UTUBSEAM40	1534.3	3087.9	7	8678	2402.8	7	712	2428.2	7	1024
UTUBSEAM10	2536.3	4652.3	7	9006	3871.2	7	936	3877.8	7	1034
RIBOTIPRE	2426.4	5868.2	9	11779	3420.6	6	284	4736.9	7	11448
1CWP	2523.1	5949.8	9	7005	4276.4	7	1412	8264.1	12	16721
1QGT	2622.9	6503.7	10	7362	5525.9	9	1562	9650.2	12	20952
UTUBSEAM20	2780.9	7263.4	10	9288	4521.4	7	815	5937.8	9	2511



Imágenes Hiperespectrales

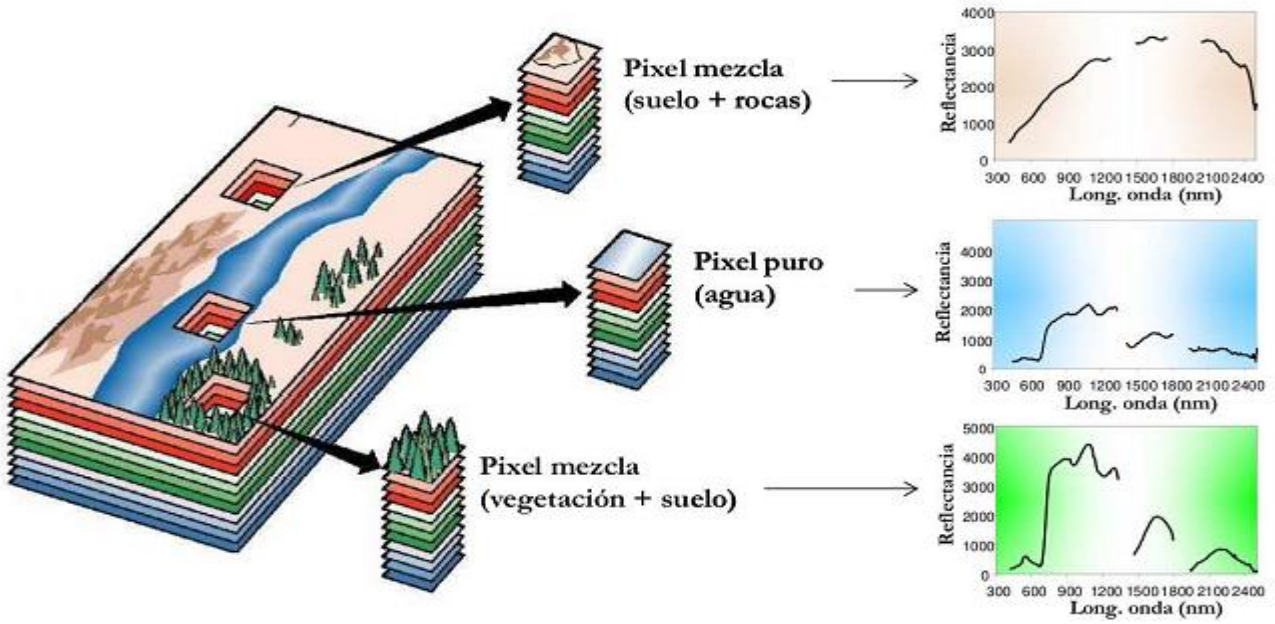
- Sergio Sánchez
- Antonio Plaza

- Francisco D. Igual

- Maribel Castillo
- Juan C. Fernández
- Germán León
- Enrique S. Quintana-Ortí
- Alfredo Remón

Imágenes Hiperespectrales

- Resolución de unas pocas docenas de metros y un tiempo de revisita de entre 3 a 5 días
- El volumen de datos justifica el procesado de imágenes a bordo (satélites o aviones no tripulados), pero requiere tiempo real



Imágenes Hiperespectrales

- Desmezclado hiperespectral: modelo lineal de mezclas en forma compacta matricial:

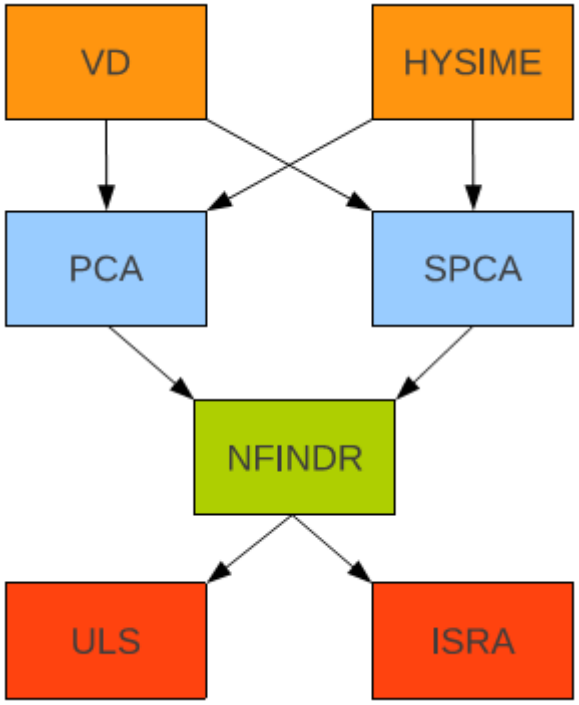
$$Y = EA + N,$$

donde Y representa la imagen hiperespectral, compuesta de m píxels (columnas) y n bandas (filas); E es la matriz de endmembers; A es la matriz de abundancias de los endmembers y N es el ruido

1. Estimación del número de endmembers (p)
2. Reducción dimensional
3. Identificación de endmembers
4. Estimación de las abundancias

Imágenes Hiperespectrales

- Diferentes métodos en cada etapa llevan a diversas cadenas de desmezclado



1. Estimación del número de endmembers (*p*)
2. Reducción dimensional
3. Identificación de endmembers
4. Estimación de las abundancias

Imágenes Hiperespectrales

- La mayoría de los problemas numéricos que subyacen en los algoritmos de desmezclado hiperespectral son operaciones de álgebra lineal:
  - Resolución de sistemas de ecuaciones lineales
  - Problemas de valores propios
  - Problemas de valores singulares

Imágenes Hiperespectrales

- GPUs de diferentes capacidades: Carma, Fermi, Kepler

Cuprite

System	Chain #1					Chain #2				
	VD	PCA	NFINDR	ULS	4 stages	HYSIME	SPCA	NFINDR	ISRA	4 stages
	Time				Total	Time				Total
Carma	1.23	1.35	1.53	0.40	4.51	11.28	1.45	0.91	5.06	18.70
Fermi	0.18	0.11	0.08	0.05	0.42	0.81	0.12	0.06	0.92	1.91
Kepler	0.16	0.08	0.07	0.04	0.35	0.63	0.10	0.05	0.87	1.65
	Avg. power				Avg.	Avg. power				Avg.
Carma	28.3	19.5	19.6	22.8	22.2	20.6	19.9	20.0	42.6	26.4
Fermi	248.3	213.9	208.8	218.2	228.2	226.6	220.0	212.9	323.1	272.2
Kepler	251.5	255.9	204.4	211.4	238.5	215.5	199.3	203.1	261.1	238.1
	Max. power				Max.	Max. power				Max.
Carma	32.5	22.5	19.6	23.1	32.5	32.8	28.4	22.5	45.6	45.6
Fermi	322.0	232.3	218.9	236.7	322.0	281.1	243.1	218.1	336.7	336.7
Kepler	273.4	264.2	208.5	214.7	273.4	237.4	218.8	205.0	270.1	270.1
	Energy				Total	Energy				Total
Carma	34.8	26.3	30.0	9.1	100.2	232.4	28.9	18.2	215.6	495.0
Fermi	44.7	23.5	16.7	10.9	95.8	183.5	26.4	12.8	297.3	520.0
Kepler	40.2	20.5	14.3	8.5	83.4	135.8	19.9	10.2	227.2	393.0
	Net Energy				Total	Net Energy				Total
Carma	19.4	9.5	10.9	4.1	43.8	91.4	10.7	6.8	152.3	261.2
Fermi	27.1	12.8	8.9	6.0	54.7	104.2	14.6	6.9	207.1	332.9
Kepler	38.2	19.5	13.4	8.0	79.1	78.1	11.7	5.5	191.8	287.1

Imágenes Hiperespectrales

- Procesadores multinúcleo de diferentes capacidades (solo OSP-GS e ISRA)

Data set	Processor	Best time	Norm.	Energy	Best energy	Norm.	Time
Cuprite	TI DSP	5.58	(10.33)	12.93	**12.93**	(1.00)	5.58
	Intel Xeon	1.14	(2.11)	107.22	103.81	(8.03)	1.70
	Intel Atom	16.96	(31.41)	125.14	125.14	(9.68)	16.96
	AMD Opteron	**0.54**	(1.00)	128.98	123.16	(9.53)	0.61
	ARM Cortex	22.94	(42.48)	41.29	36.89	(2.85)	23.18
WTC	TI DSP	15.27	(6.01)	36.89	**36.89**	(1.00)	15.27
	Intel Xeon	4.24	(1.67)	394.08	384.39	(10.42)	6.32
	Intel Atom	62.27	(24.52)	493.88	493.88	(13.39)	67.27
	AMD Opteron	**2.54**	(1.00)	578.83	556.97	(15.10)	2.80
	ARM Cortex	87.76	(34.56)	168.17	137.13	(3.72)	87.83

Reducción de Modelos

- Peter Benner
- Alfredo Remón

- Pablo Ezzatti

- Enrique S. Quintana-Ortí

Reducción de Modelos

- Dado un modelo de un proceso físico, reemplazarlo por otro más simple
  - Diseño de controladores
    - Tiempo real solo es posible con controladores de baja complejidad
    - Controladores simples son más robustos
  - Simulación
    - ¡Reducir una vez, simular muchas!

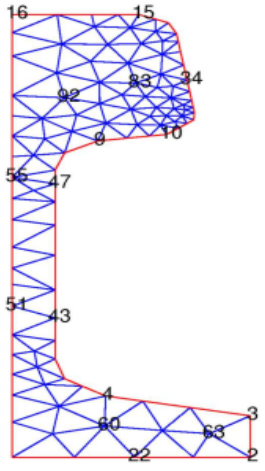
Reducción de Modelos

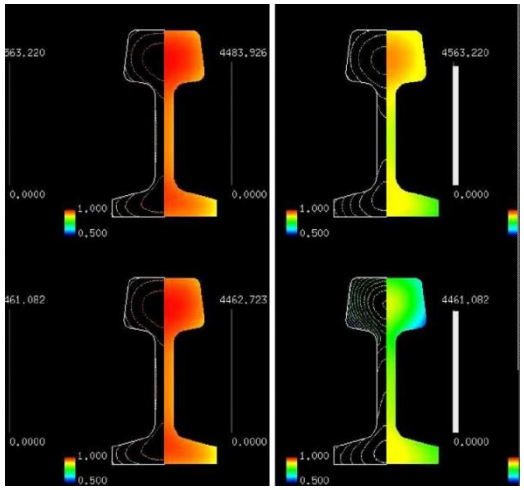
- Enfriado óptimo de railes de tren
  - Reducir la temperatura rápidamente (productividad)
  - Evitar grandes gradientes de temperatura (durabilidad)

- Ecuación de Riccati

$$X := F(A,S,Q) = 0,$$

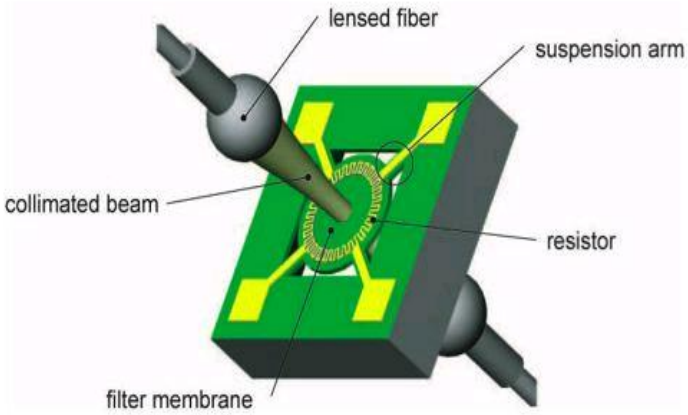
$A \rightarrow n \times n$ con n = 79,841, dependiendo del mallado





Reducción de Modelos

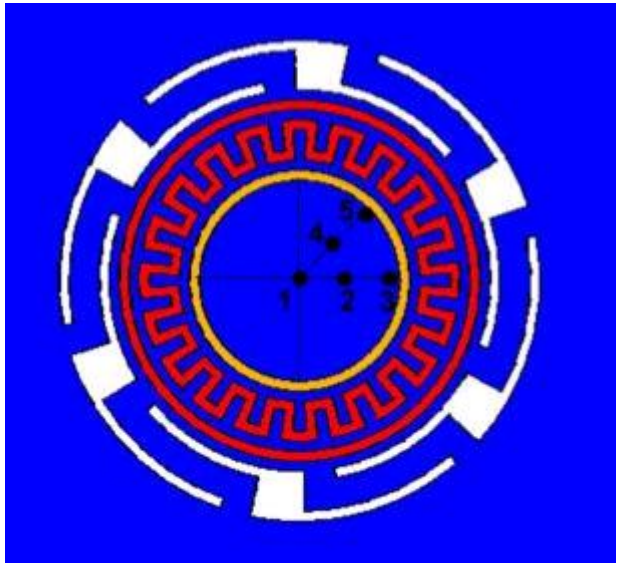
- Filtro óptimo adaptable:
  - Determinar la potencia eléctrica a apliar para alcanzar la temperature crítica y una distribución de temperature óptima



- Ecuación de Riccati

$$X := F(A,S,Q) = 0,$$

$A \rightarrow n \times n$ con n = 108,373



Reducción de Modelos

- Método de la función signo:

  Invertir $A \rightarrow n \times n$

  $\approx 2n^3$ flops

- Intel Xeon: 4 DP flops/ciclo,

  e.g., f=2.0 GHz

n	Tiempo 1 core	Tiempo 8 cores	Tiempo cluster de 16 nodos, 8 cores por nodo (192 cores)
100	25,0 ms	--	--
1.000	0,25 s	--	--
10^4	> 4 m	31,2 s	--
10^5	> 69 h	> 8 h	> 21 m

Steel profiles →

Optical filter →

¡Métodos computacionales para reducción de modelos son caros desde el punto de vista computacional!

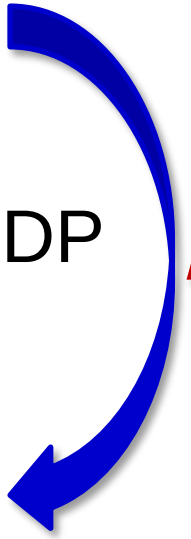
Reducción de Modelos

- Cluster Intel Pentium II, 32 nodos (300MHz. DGEMM: 180 DP MFLOPS/core) con red Myrinet
  - Ecuación de Lyapunov *n*=5,177 (método BT) en 38.5 m.

    "State-space truncation methods for parallel model reduction of large-scale systems".
    P. Benner, E. S. Quintana, G. Quintana.
    Parallel Computing, 2003

- Servidor 8 cores Intel Xeon (2.3 GHz. Pico: 9.2 DP GFLOPS/core) y Tesla C1060 (240 cores):
  - Ecuación de Lyapunov *n*=5,177 (método BT) en 55.5 s.

    "A mixed-precision algorithm for the solution of Lyapunov equations on hybrid CPU-GPU platforms".
    P. Benner, P. Ezzatti, D. Kressner, E. S. Quintana, A. Remón.
    Parallel Computing, 2011



/40

ILUPACK

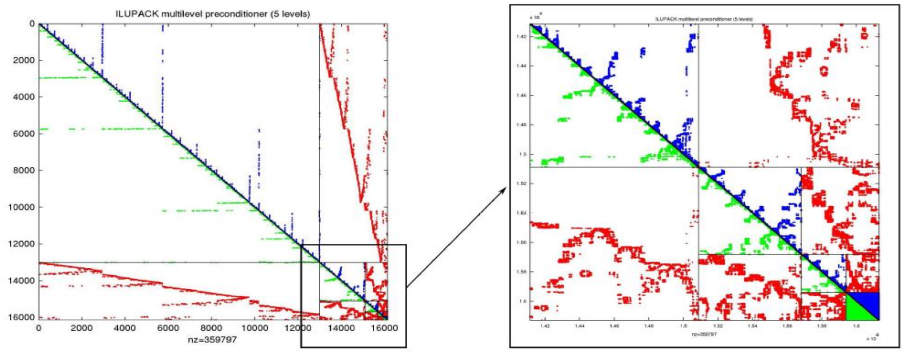
- Matthias Bollhöfer

- ABB Suiza

- José I. Aliaga
- Enrique S. Quintana-Ortí

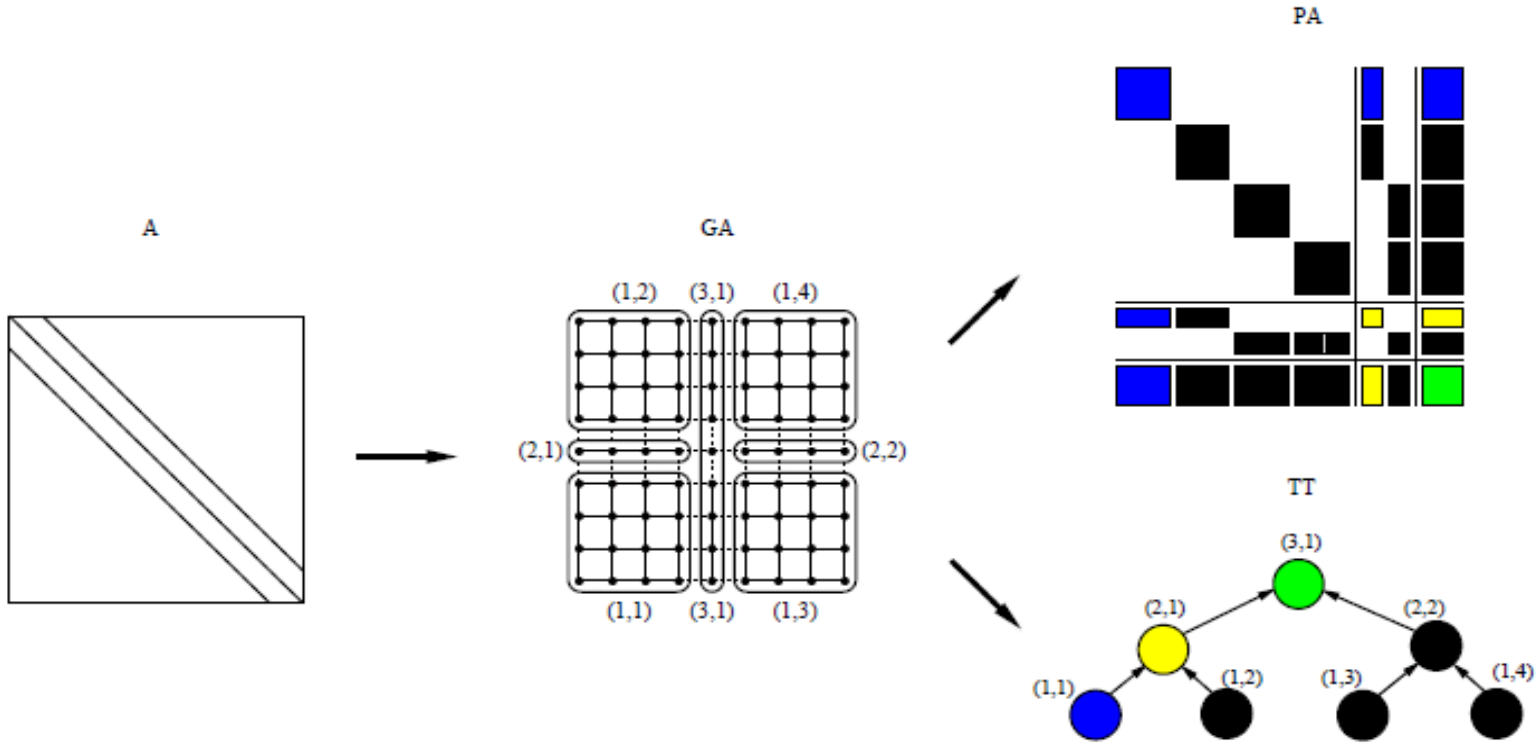
ILUPACK

- Incomplete LU Package (http://ilupack.tu-bs.de)
  - Métodos basados en subespacios de Krylov
  - Precondicionadores ILU multinivel para sistemas de ecuaciones lineales dispersos
  - Basado en ILUs con control del crecimiento de los factores triangulares
  - Competitivos para PDEs 3D



ILUPACK

- Paralelismo de tareas
  - Aprovechamiento en arquitecturas multihebra y clusters



Sumario

- Álgebra lineal densa está en la base de muchas aplicaciones científicas y de ingeniería:
  - Simulación macromolecular
  - Imágenes hiperespectrales
  - Reducción de modelos

- Otros problemas básicos:
  - Ordenación
  - Búsqueda
  - Optimización y matemática discreta
  - …

Índice

- Estado actual de la arquitectura de computadores
- Aplicaciones en HPCA@UJI
- Líneas de investigación en HPCA@UJI
  - Programación paralela
  - Ahorro de energía

Supercomputadores: Top500 (noviembre 2013)

Puesto	Sistema	#Núcleos	Procesador/Red	LINPACK (TFLOPS)
1	Tianhe-2 National Supercomputer Center Guangzhou	3.120.000	Intel Xeon E5, Intel Xeon Phi/ Infiniband	33.862
2	Titan DOE/SC/ORNL	560.640	Opteron 6274, NVIDIA K20x/ Cray Gemini	17.590
3	Sequoia DOE/NNSA/LLNL	1.572.864	Blue Gene/Q / IBM	17.173

Programación Paralela

2013 17 PFLOPS ($17 \cdot 10^{15}$ flops/sec.)

2013 Sequoia

- 10^{10} nivel core
  (PowerPC A2, 1,6 GHz → 12,8 GFLOPS)
- 10^{1} nivel node
  (16 cores/node)
- 10^{5} nivel cluster
  (98.304 nodes)

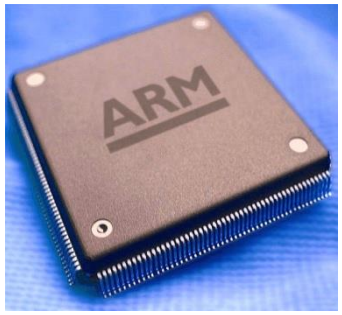
2020 EFLOPS (10^{18} flops/sec.)

- 10^{10} nivel core
- 10^{3} nivel nodo!
- 10^{5} nivel cluster

Supercomputadores

- Arquitecturas “convencionales”

- Nuevos oponentes…



Programación Paralela

- ¿Álgebra lineal?

¡Qué no cunda el pánico!

  - Determinantes, sistemas lineales,
    ajuste por mínimos cuadrados, FFT, etc.

- Importancia:
  - Intel MKL, AMD ACML, IBM ESSL, NVIDIA CUBLAS,…

Programación Paralela

- Sistema lineal

$$\left.\begin{aligned} 2x + 3y &= 3 \\ 4x - 5y &= 6 \end{aligned}\right\}$$

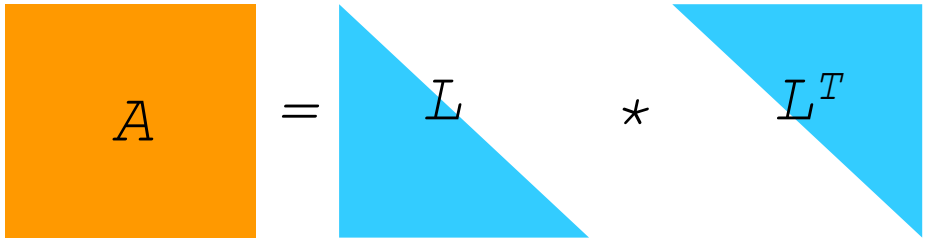
$A\,X = B$, con $A, B \rightarrow n \times n$ densas:

$\approx 2n^3/3 + 2n^3$ flops

- Intel Xeon:

4 DP flops/ciclo, e.g., a f=2.0 GHz

n	Tiempo 1 core	Tiempo 8 cores	Tiempo cluster de 16 nodos, 8 cores por nodo (192 cores)
100	33,33 ms	--	--
1.000	0,33 s	--	--
10^4	333,33 s	41,62 s	--
10^5	> 92 h	> 11 h	> 28 m

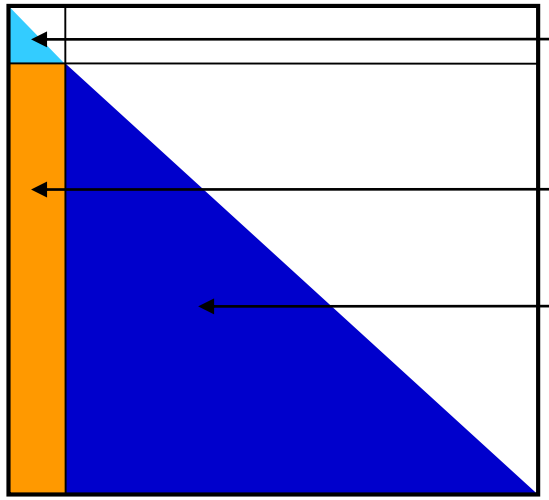


Clave en la solución de sistemas de ecuaciones lineales s.p.d.

$$A\ x\ =\ b \equiv \qquad (LL^T)x\ =\ b$$

$$L\ y\ =\ b \quad \Rightarrow\ y$$

$$L^T x\ =\ y \quad \Rightarrow\ x$$



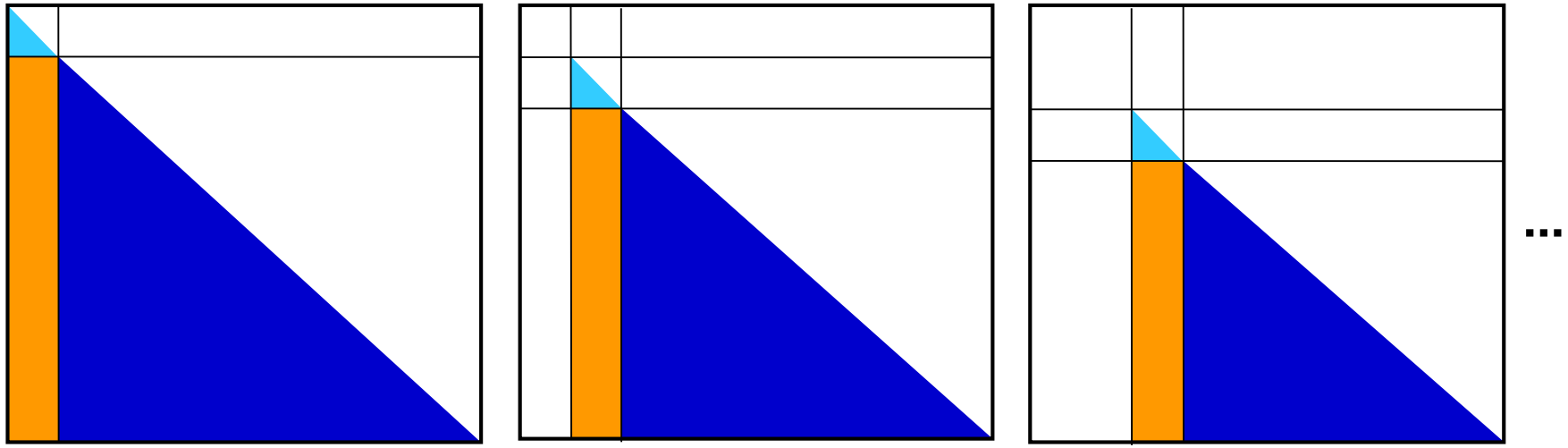
1ª iteración

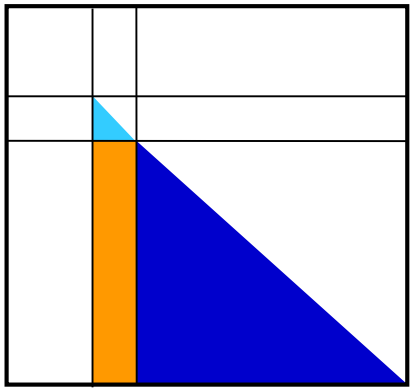
F: $A_{11} = L_{11} * L_{11}^{T}$

T: $L_{21} \leftarrow A_{21} * L_{11}^{-T}$

U: $A_{22} \leftarrow A_{22} - L_{21} * L_{21}^{T}$

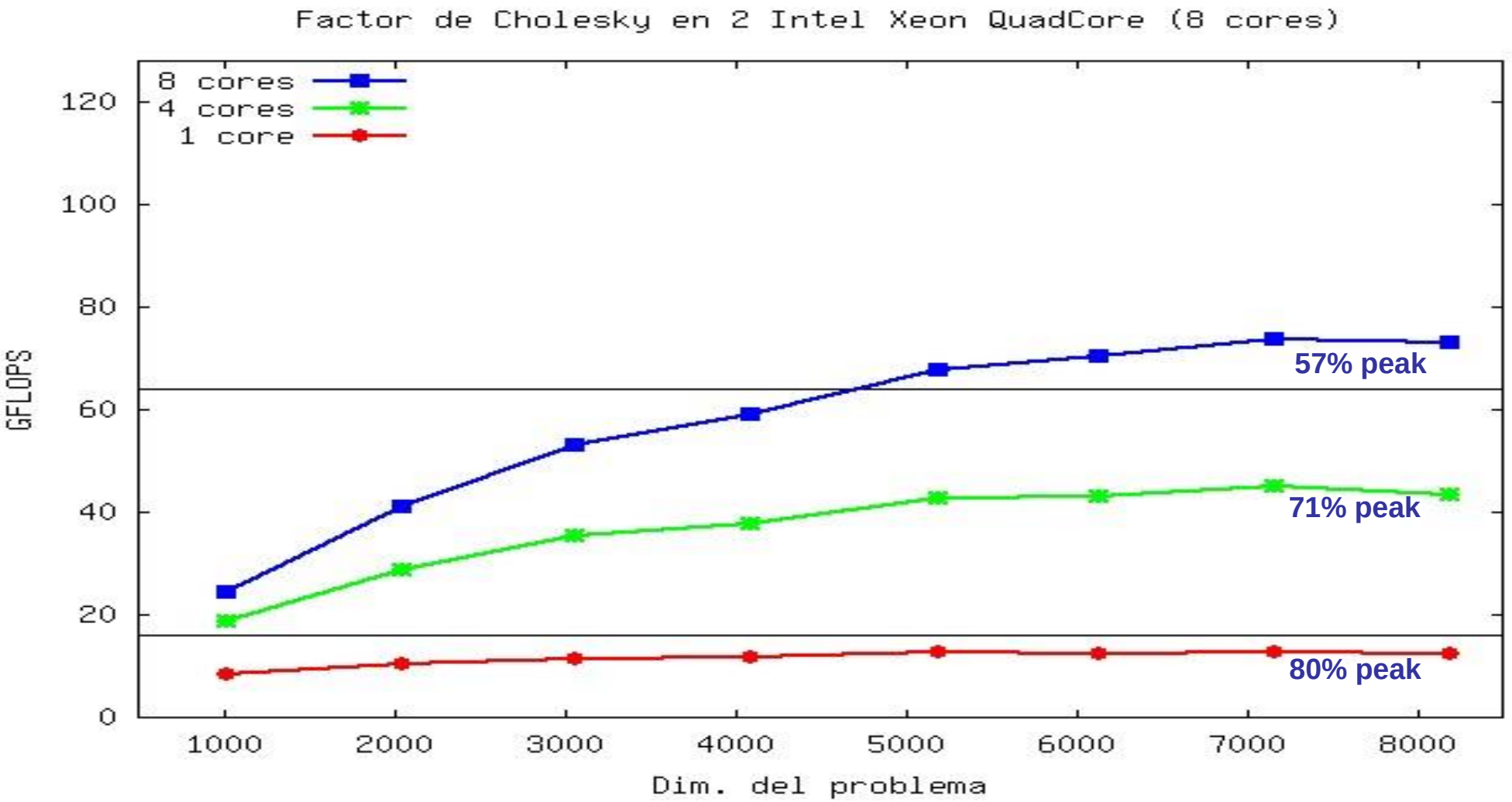
- Reusar datos en cache
- Procesador MT: Emplear una implementación MT de T y P


...
1ª iteración
2ª iteración
3ª iteración



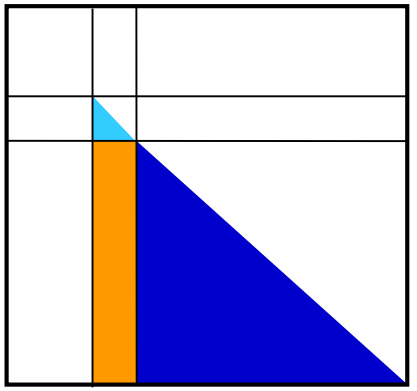
```
for (k=1; k<=n/b; k++){
F:   Chol(A[k,k]);   // A_kk      = L_kk * L_kk^T

   if (k<=n/b){
T:    Trsm(A[k,k], A[k+1,k]);     // L_k+1,k   ← A_k+1,k * L_kk^-T

U:    Syrk(A[k+1,k], A[k+1,k+1]);// A_k+1,k+1 ← A_k+1,k+1
                                  //           - L_k+1,k * L_k+1,k^T
   }
}
```

Programación Paralela



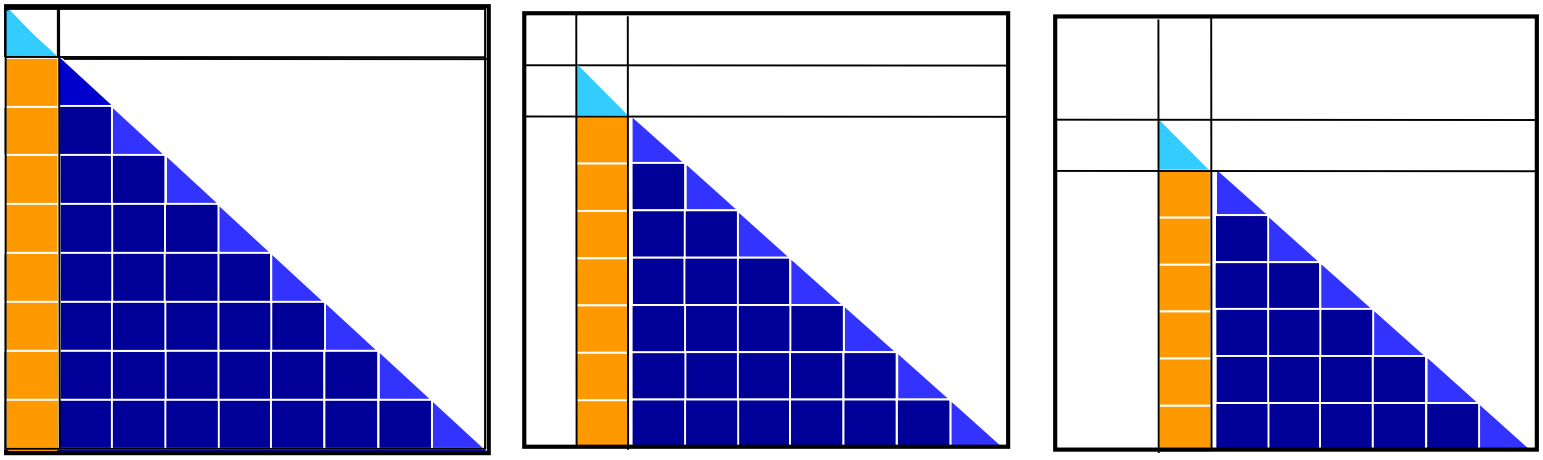
- ¿Por qué?

  Excesiva sincronización entre threads



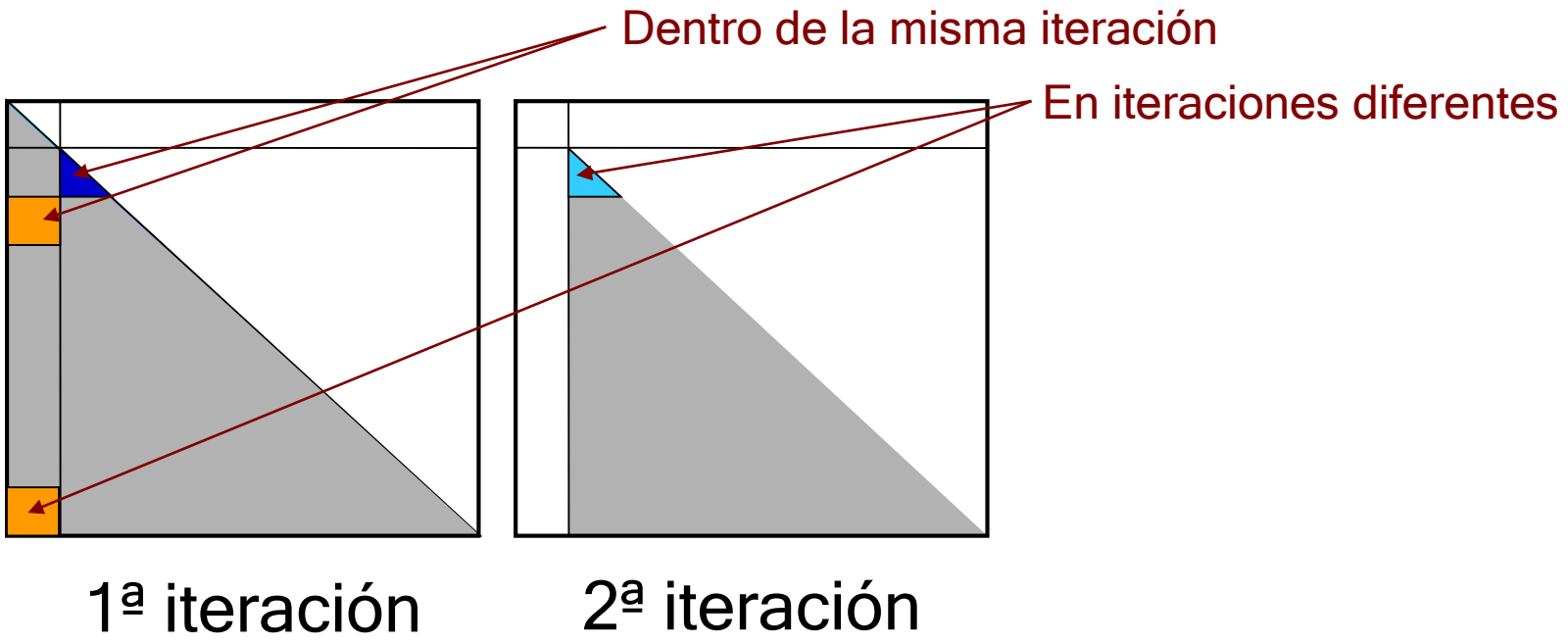
```
for (k=1; k<=n/b; k++){
F:   Chol(A[k,k]);                          // A_kk       = L_kk * L_kk^T

     if (k<=n/b){
T:      Trsm(A[k,k], A[k+1,k]);             // L_k+1,k    <- A_k+1,k * L_kk^-T
U:      Syrk(A[k+1,k], A[k+1,k+1]);         // A_k+1,k+1  <- A_k+1,k+1
                                            //            - L_k+1,k * L_k+1,k^T
     }
}
```

- ¡¡¡pero hay mucho más paralelismo!!!



1ª iteración 2ª iteración 3ª iteración

- ¡¡¡pero hay mucha más concurrencia!!!



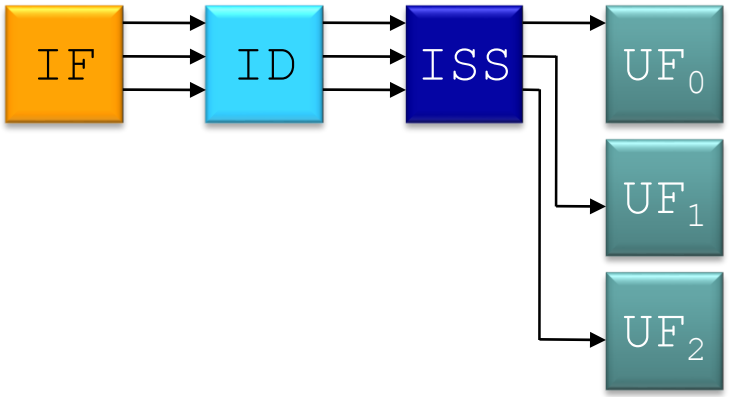
¿Cómo podemos explotarla?

Programación Paralela

Código escalar

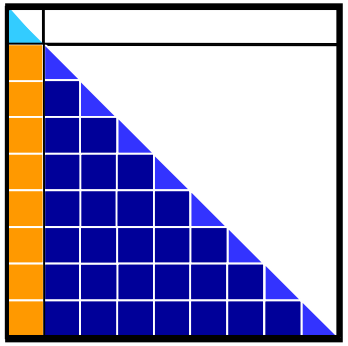
```
loop: ld   f0, 0(r1)
      addd f4, f0, f2
      sd   f4, 0(r1)
      addi r1, r1, #8
      subi r2, r2, #1
      bnez r2, loop
```

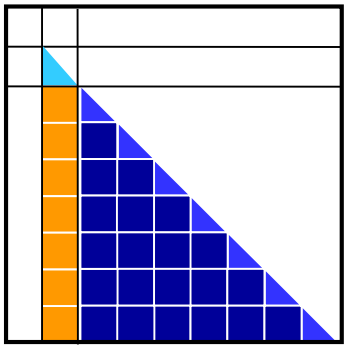
Procesador superescalar



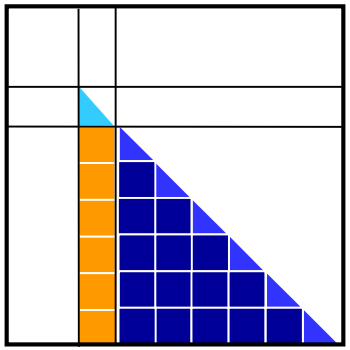
- ¿Algo similar para álgebra lineal?

1ª iter.





2ª iter.

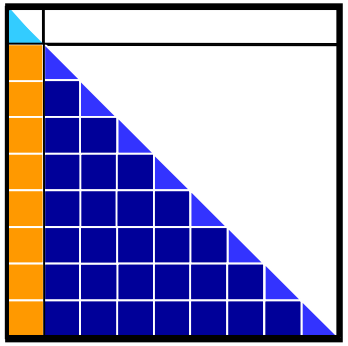


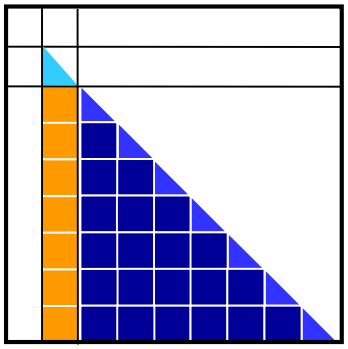
3ª iter.

```
   for (k=1; k<=n/b; k++){
F:   Chol(A[k,k]);
     for (i=k+1; i<=n/b; i++)
T:     Trsm(A[k,k], A[i,k]);
     for (i=k+1; i<nb; i++){
U1:    Syrk(A[i,k],A[i,i]);
       for (j=k+1; j<=i; j++)
U2:      Gemm(A[i,k], A[j,k], A[i,j]);
     }
   }
```

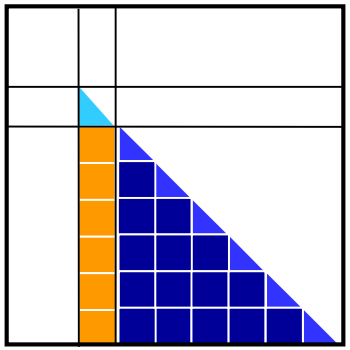
- ¿Algo similar para álgebra lineal?

1ª iter.





2ª iter.



3ª iter.

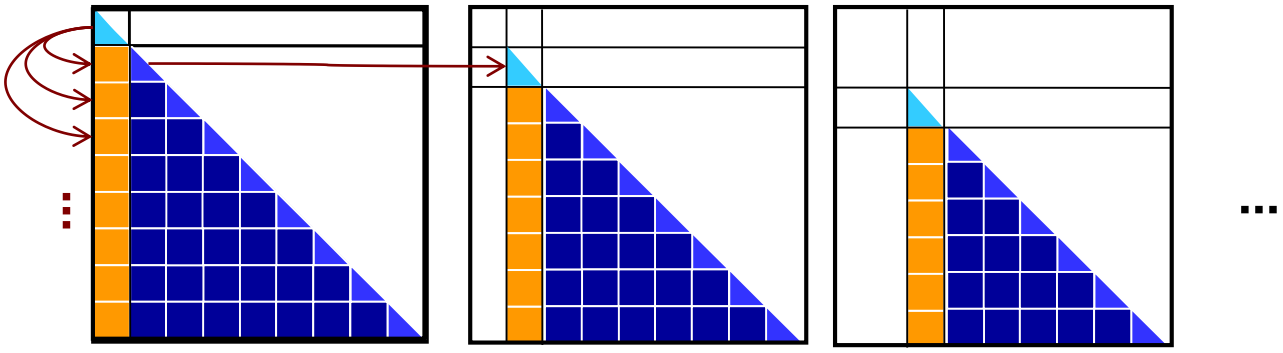
- Aplicar técnicas escalares a nivel de bloque
- Implementación por software
- Paralelismo de tareas/threads
- Objetivo: cores/GPUs de la plataforma

- Bloques leídos/escritos determinan las dependencias, como en el caso escalar

```
loop:  ld   f0, 0(r1)
       addd f4, f0, f2
       sd   f4, 0(r1)
       addi r1, r1, #8 ...
```

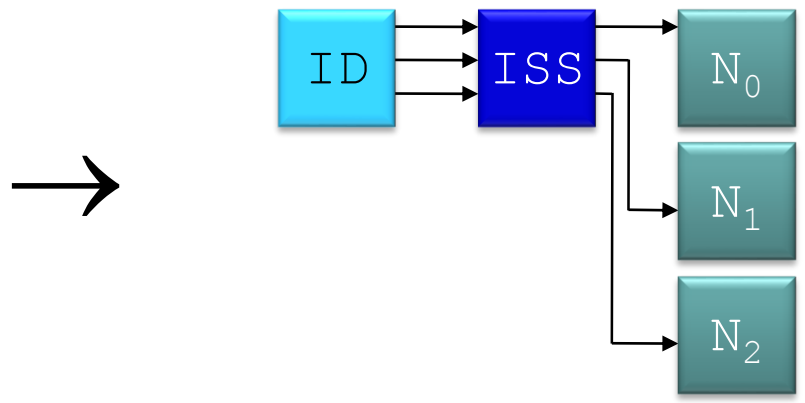
```
for (k=1; k<=n/b; k++){
  Chol(A[k,k]);
  for (i=k+1; i<=n/b; i++)
    Trsm(A[k,k], A[i,k]); ...
```

Las dependencias forman un DAG (árbol de tareas)

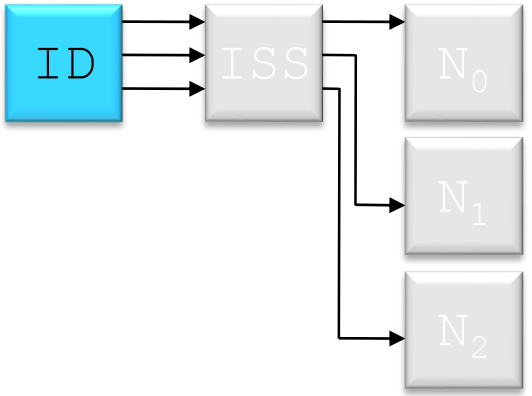


Programación Paralela

- *Runtime*:
  - Decodificación (ID): Generar el árbol de tareas a partir de un “análisis simbólico” del código en tiempo de ejecución
  - Emisión (ISS): Ejecución de las tareas del árbol consciente de la arquitectura



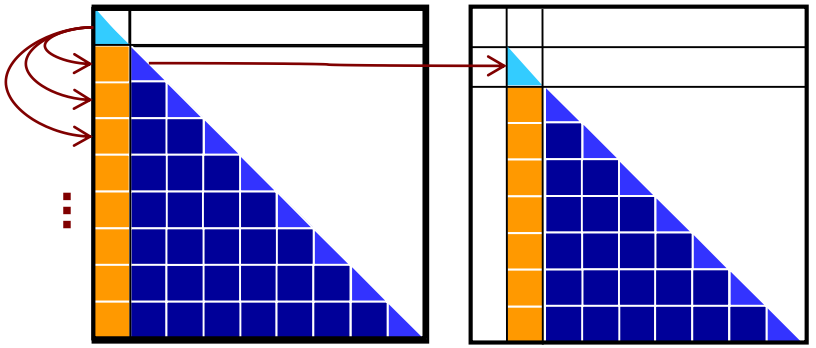
- Etapa de decodificación:
  - "Análisis simbólico" del código



Código por bloques:

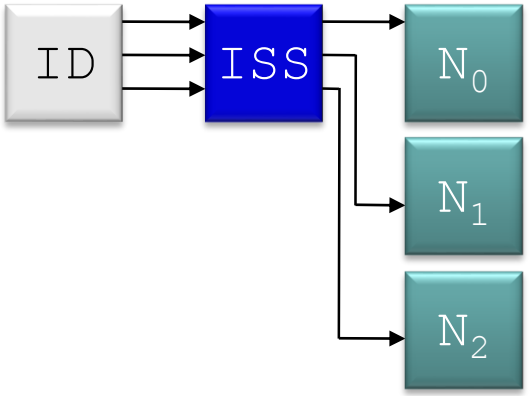
```
for (k=1; k<=n/b; k++){
  Chol(A[k,k]);
  for (i=k+1; i<=n/b; i++)
    Trsm(A[k,k], A[i,k]); …
```

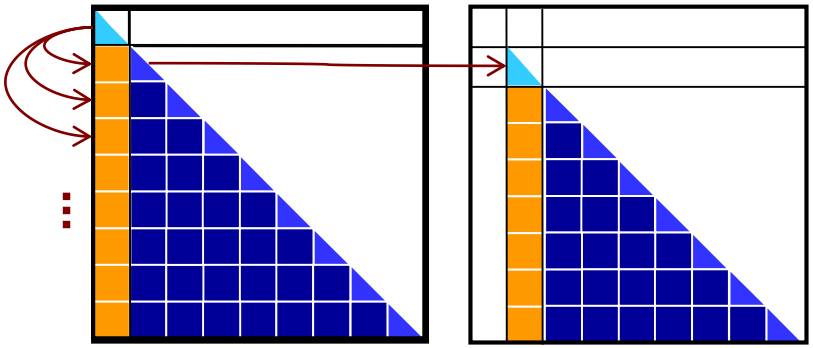
Árbol de tareas:



Programación Paralela

- Etapa de emisión:
  - Planificación temporal de tareas, en función de las dependencias
  - Mapeado (planificación especial) de tareas a recursos, consciente de la localidad





Programación Paralela

- SuperMatrix (UT@Austin and UJI)
  - Bloques leídos/escritos definidos implícitamente por la operación
  - Sólo válido para operaciones de algebra lineal codificadas en `libflame`

- OMPSs (BSC)
  - Lenguaje tipo OpenMP

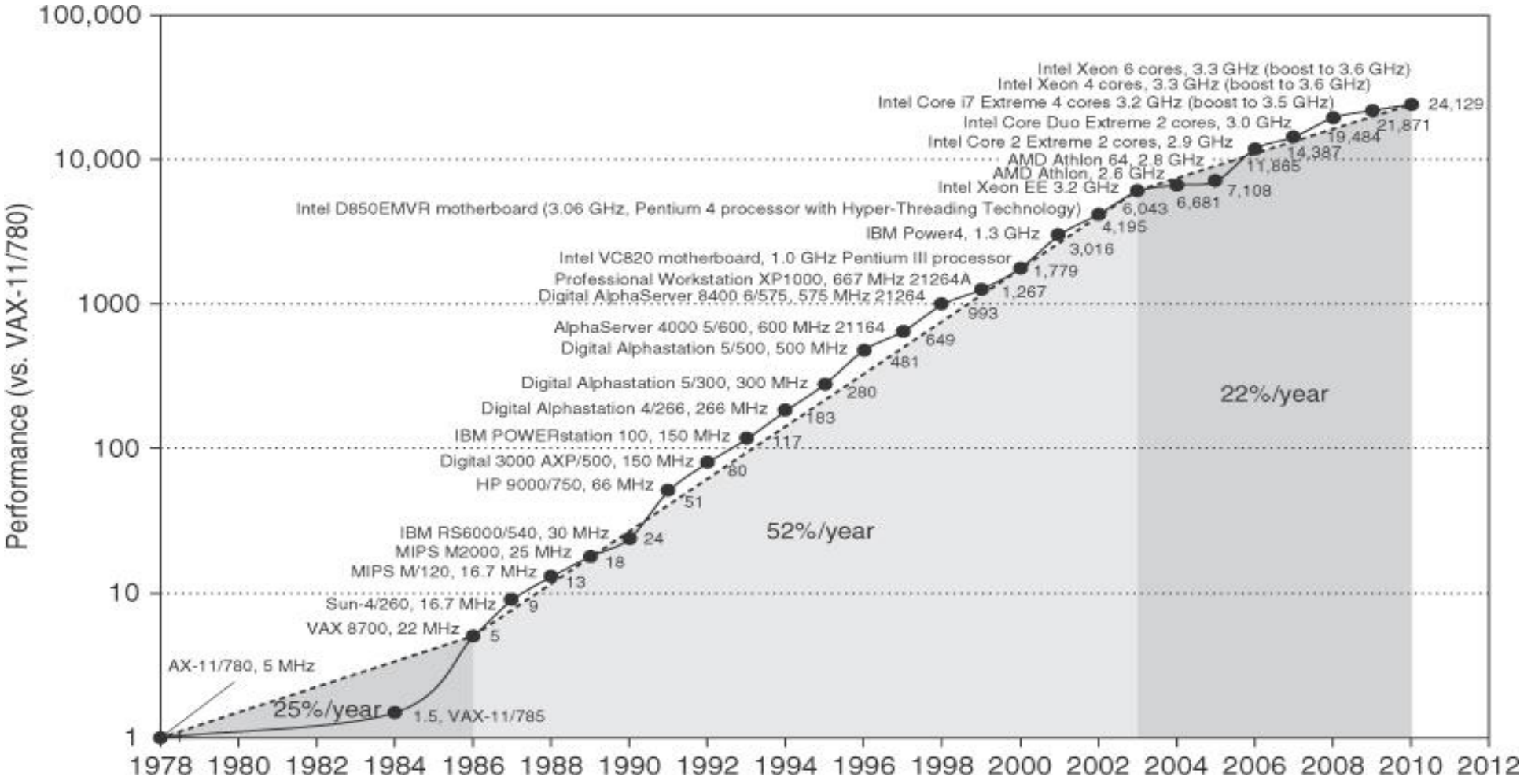
```
#pragma css task inout(A[b*b])
void Chol(double *A);
```

  - Aplicable a códigos con paralelismo de tareas en diferentes plataformas: multinúcleo, multi-GPU, cluster, Grid,…

Índice

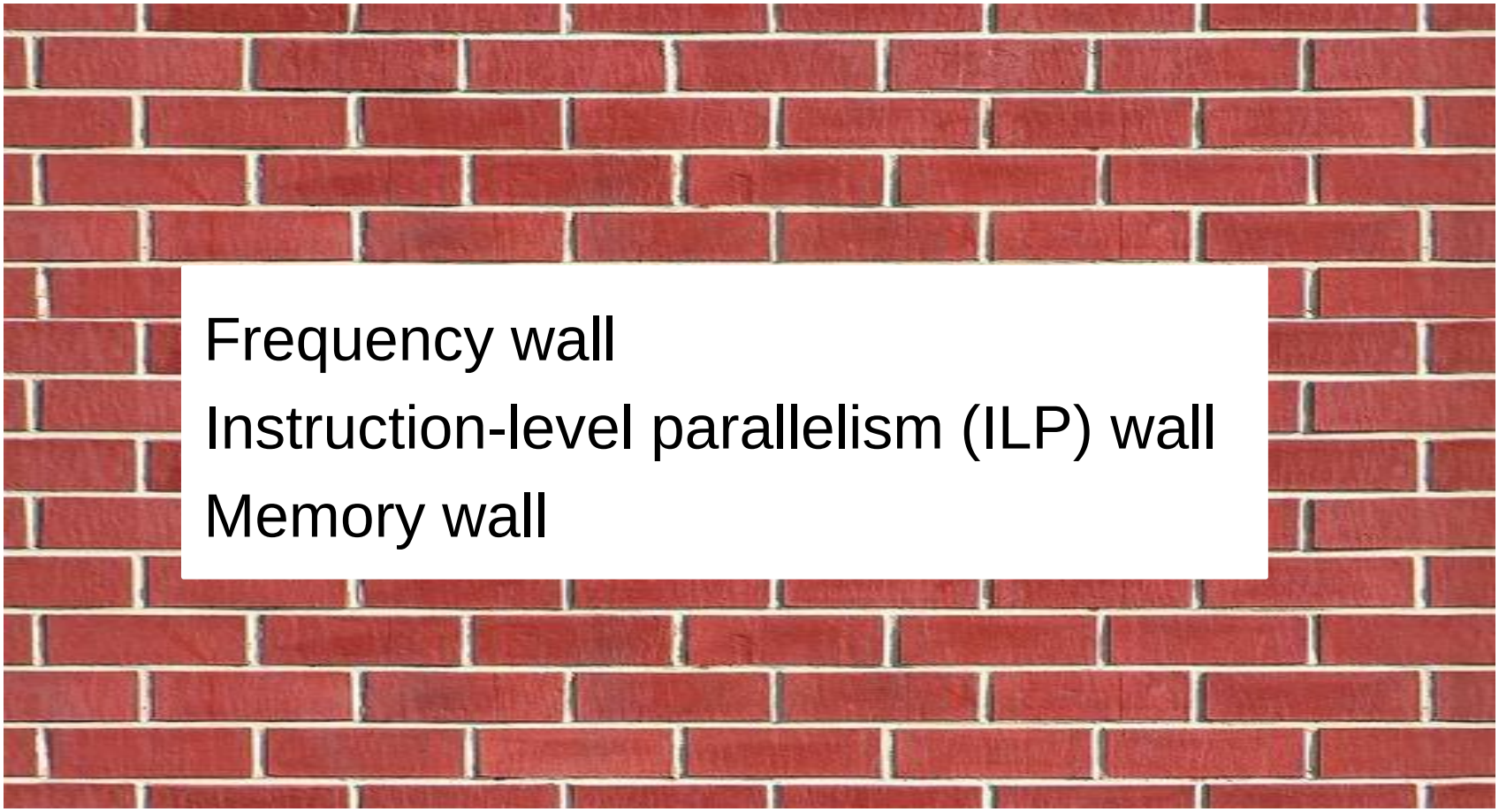
- Estado actual de la arquitectura de computadores
- Aplicaciones en HPCA@UJI
- Líneas de investigación en HPCA@UJI
  - Programación paralela
  - Ahorro de energía

Ahorro de Energía



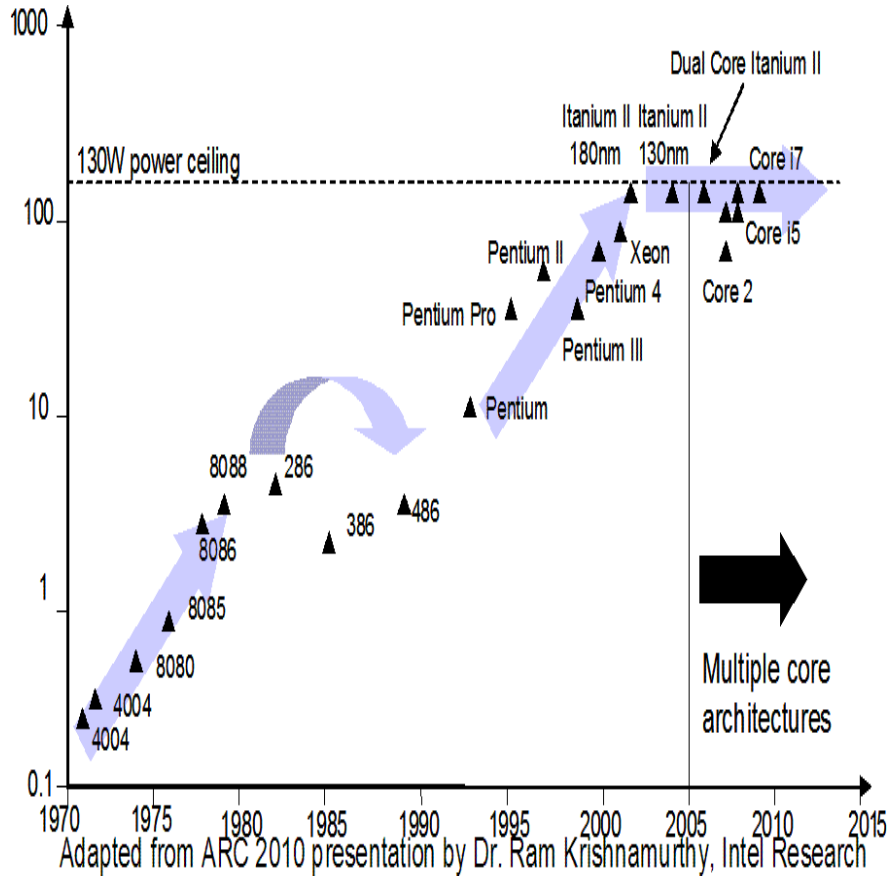
Ahorro de Energía

- "The free lunch is over" (H. Sutter, 2005)



Ahorro de Energía

- Frequency wall
  - Potencia – Energía proporcional a f^3 – f^2
    - Electricidad = dinero
  - 1ª Ley de termodinámica: La energía no se crea ni se destruye, tan solo se transforma:
    - Coste de extraer el calor
    - La temperatura reduce el tiempo de vida



Ahorro de Energía

Puesto Green/Top	Sistema	#Núcleos	MFLOPS/W	LINPACK (TFLOPS)	MW para EXAFLOPS?
1/11	TSUBAME 2.5 - Tokio Institute of Technology. Intel Xeon X5670 (6C) & NVIDIA K20x	74.358	4.503,17	2.843,0	222,07
7/1	Tianhe-2 - National Supercomputer Center Guangzhou. Intel Xeon E5 (12C) & Intel Xeon Phi	3.120.000	1.901,54	33.862,7	527,52

NVIDIA GTX 480 (250 W) (=1/4 secador de baja potencia)
0.5 millones de GTXs ≈ 222,07 MW!
o 250.000 secadores de pelo

Ahorro de Energía

Puesto Green/Top	Sistema	#Núcleos	MFLOPS/W	LINPACK (TFLOPS)	MW para EXAFLOPS?
1/11	TSUBAME 2.5 - Tokio Institute of Technology. Intel Xeon X5670 (6C) & NVIDIA K20x	74.358	4.503,17	2.843,0	222,07
7/1	Tianhe-2 - National Supercomputer Center Guangzhou. Intel Xeon E5 (12C) & Intel Xeon Phi	3.120.000	1.901,54	33.862,7	527,52

Reactor nuclear más potente en construcción en Francia: Flamanville (EDF, 2017.
6,660 millones de euros):

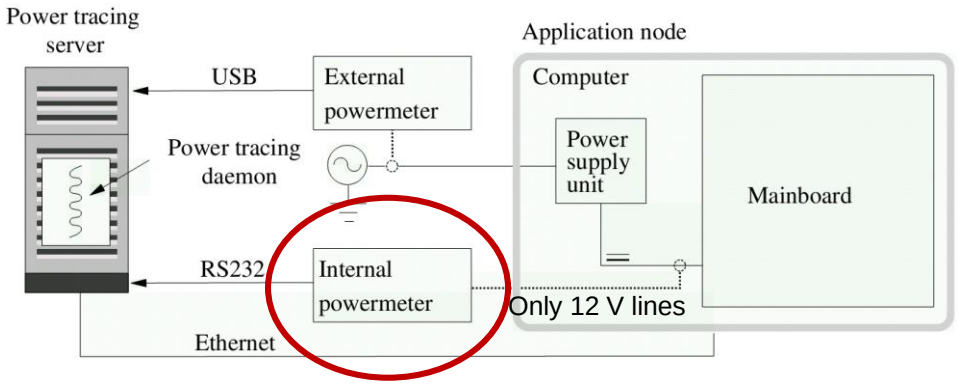
1,630 MWe

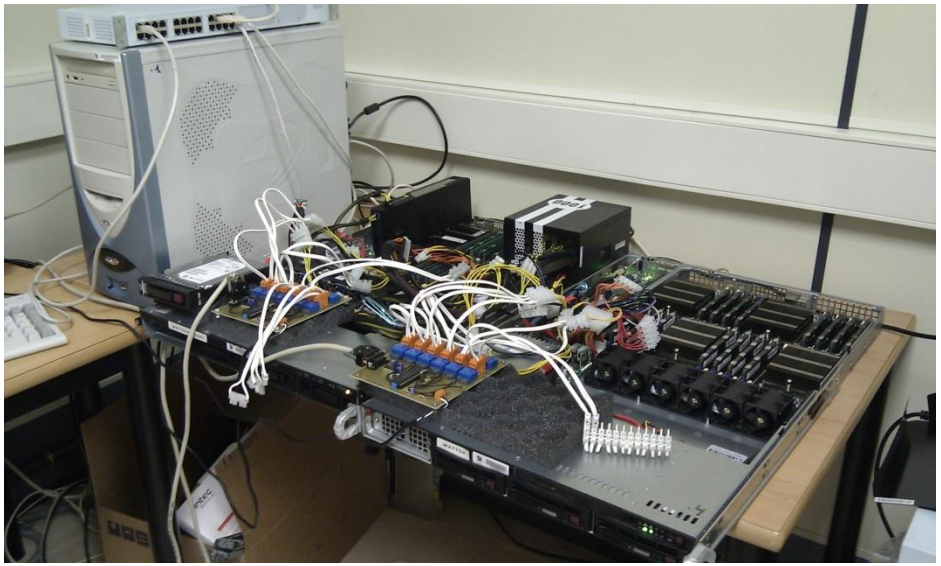
Ahorro de Energía

- Oportunidades de ahorrar energía en aplicaciones con paralelismo de datos
  - 2 procesadores AMD Opteron 6128 @ 2.0 GHz (16 núcleos)
  - Experiencia: mayor flexibilidad (DVFS nivel de núcleo)

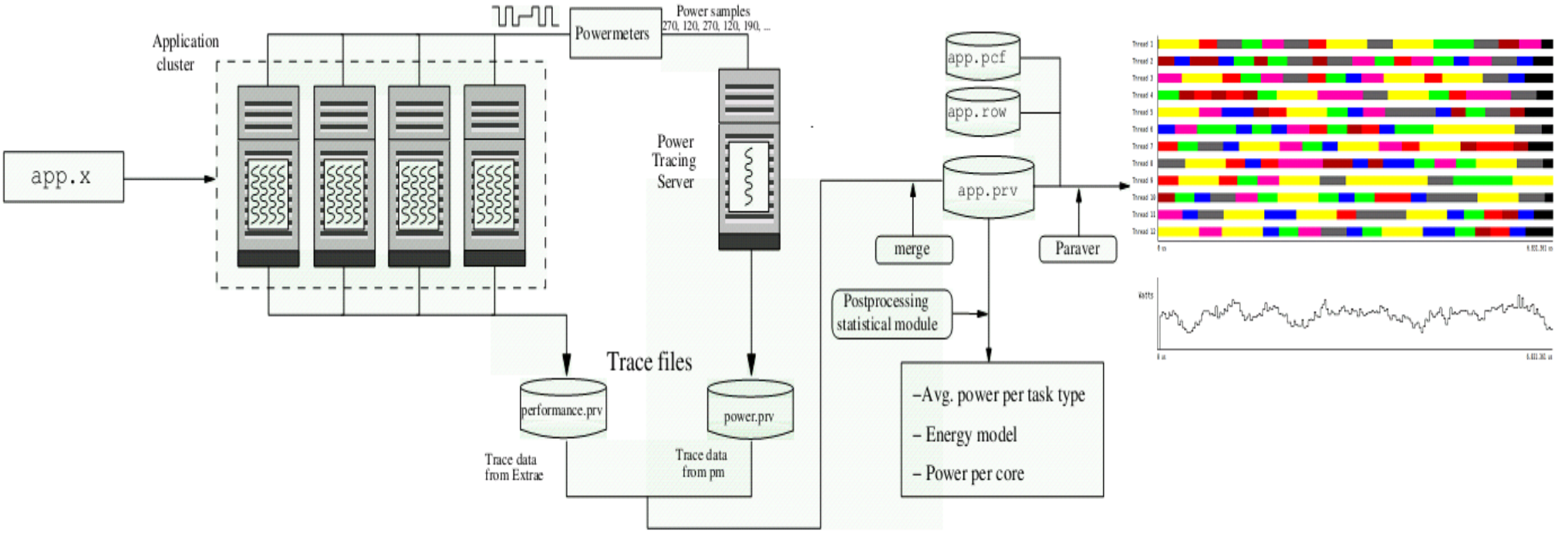
Ahorro de Energía

- Wattmeter DC con frecuencia de muestreo $= 25$ Hz

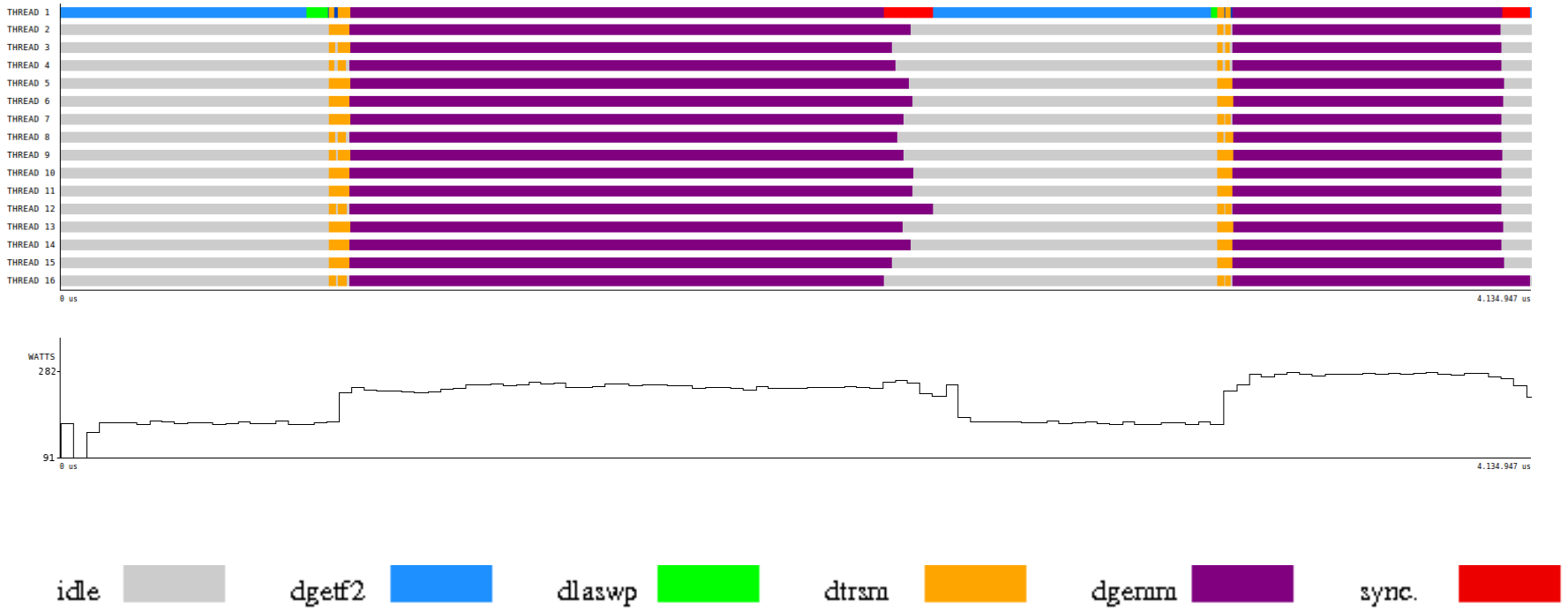




Ahorro de Energía

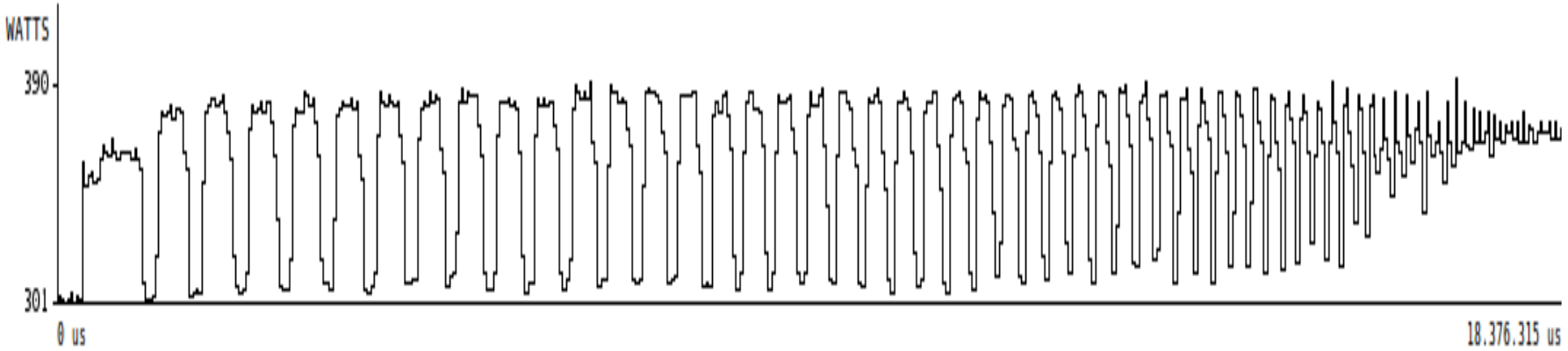


Ahorro de Energía



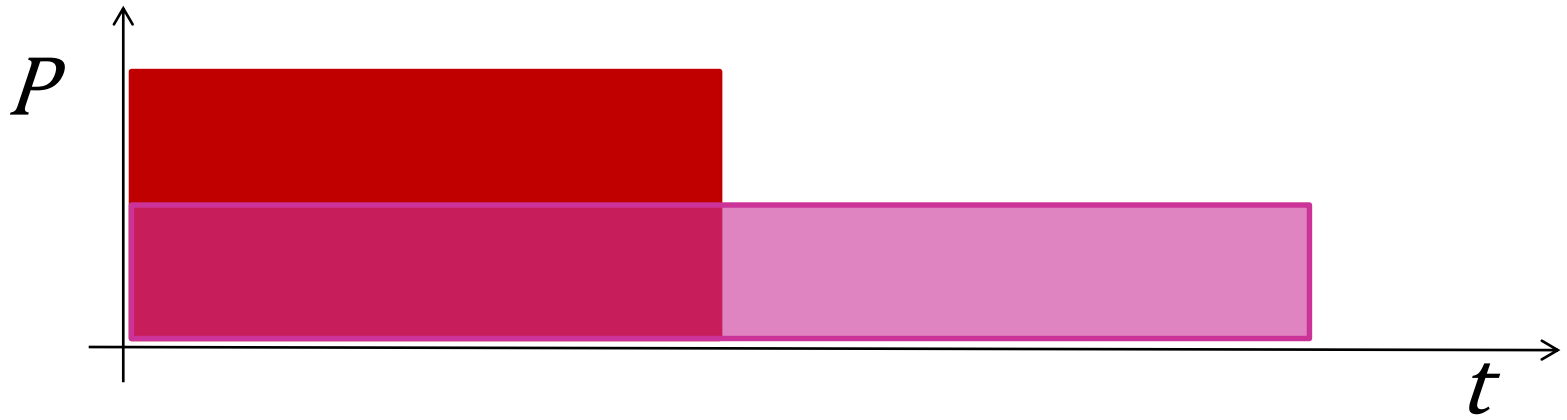
Ahorro de Energía

$$E = \int_0^T P \, dt$$



Ahorro de Energía

- ¿Qué es mejor, A o B?



$$E_A = P_A t_A$$
$$E_B = P_B t_B$$

Ahorro de Energía

$$P = P^{(S)Y(stem)} + P^{C(PU)} = P^{Y} + P^{S(tatic)} + P^{D(ynamic)}$$

P^C es la potencia de la CPU (socket): $P^S + P^D$

P^Y es la potencia del resto de componentes (e.g., RAM)

Consideraciones:

- P^Y y P^S son constantes (aunque P^S crece con la temperatura)
- Sistema “caliente”
- Rutinas con paralelismo de datos
- Plataforma Intel

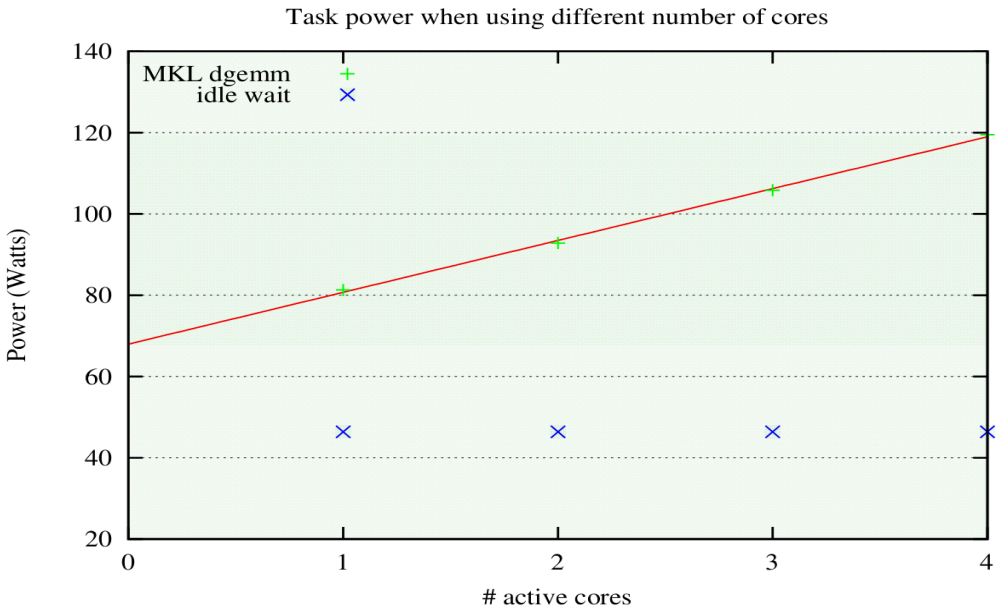
Ahorro de Energía

- Potencia de sistema: $P = P^Y + P^S + P^D$

Estimada como potencia idle

Debida a componentes off-chip:

e.g, RAM (solo placa base)



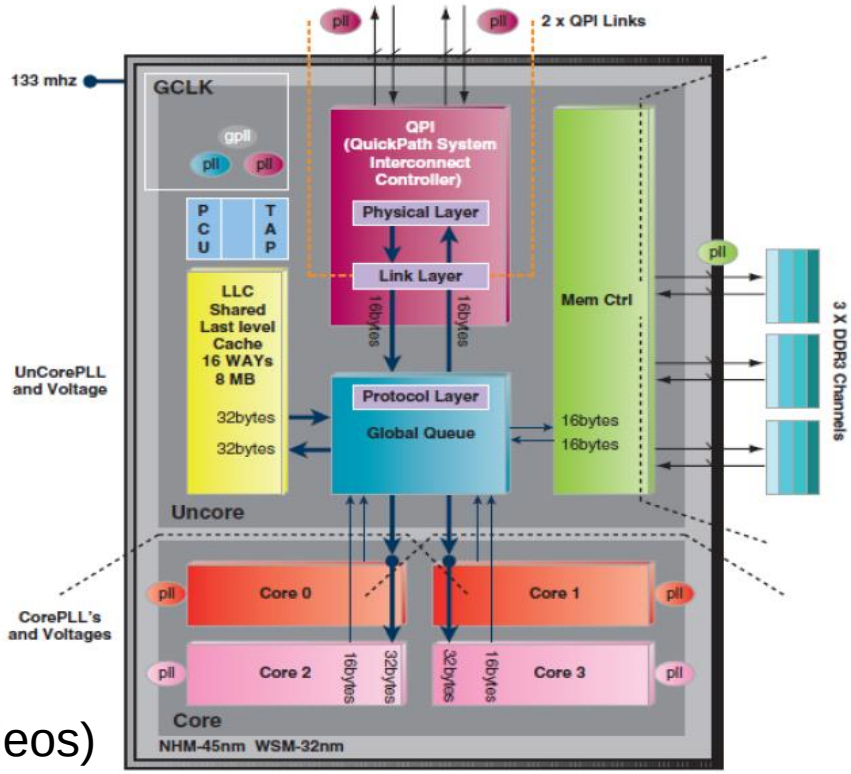
$$P^Y \approx P^I = 46.37 \text{ W}$$

Ahorro de Energía

- Potencia estática: $P = P^Y + P^S + P^D$

También conocida como potencia *uncore* (Intel):

- LLC
- Controlador de memoria
- Controlador de red
- Lógica de control de potencia
- etc.



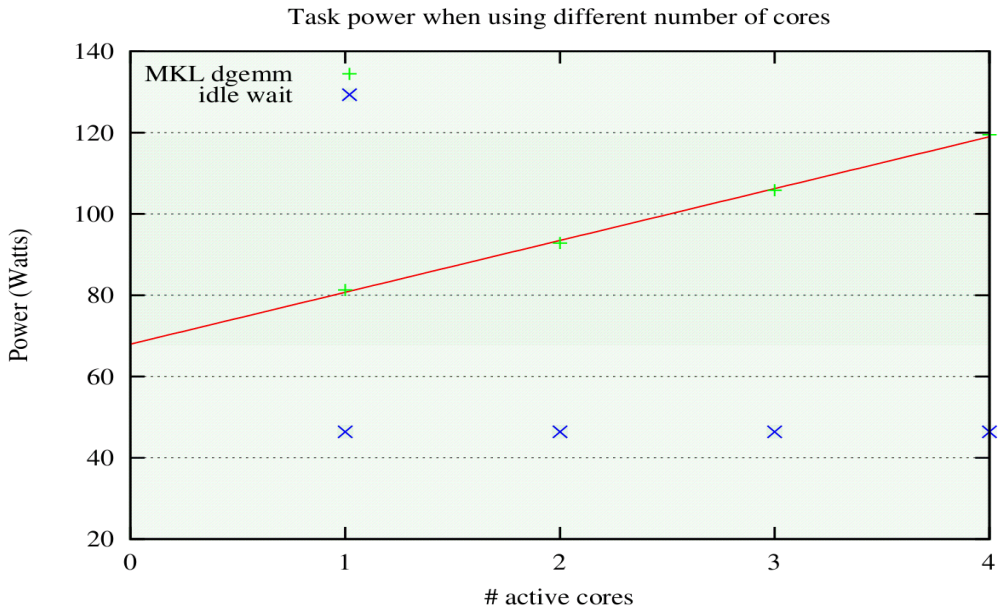
Intel Xeon 5500 (4 núcleos)

The Uncore: A Modular Approach to Feeding the High-performance Cores.
D. L. Hill et al. Intel Technology Journal, Vol. 14(3), 2010

Ahorro de Energía

- Potencia estática:

$$P = P^Y + P^S + P^D$$



$$P_{dgemm}(c) = 67.97 + 12.75\,c$$

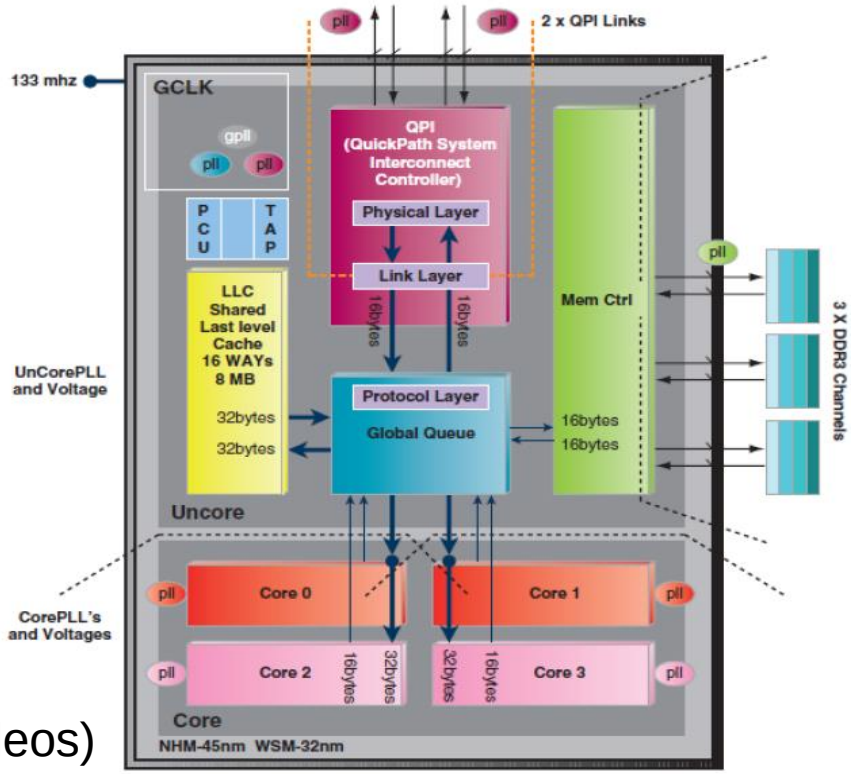
$$P^S = 67.97 - 46.37 = 21.6\ \text{W}$$

Ahorro de Energía

- Potencia dinámica: $P = P^Y + P^S + P^D$

También conocida como potencia core (Intel):

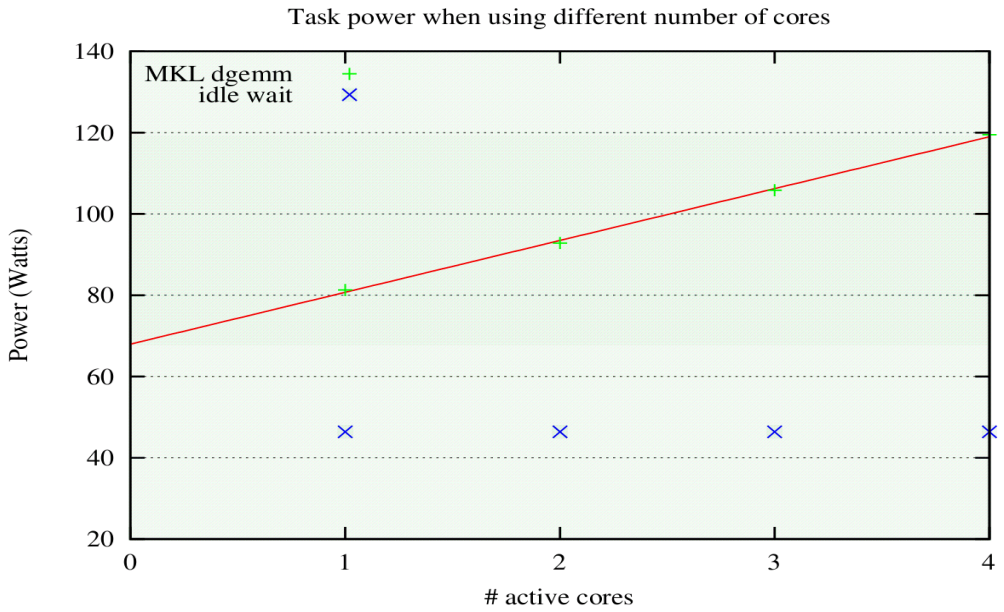
- Unidades de ejecución
- Caché L1 y L2
- Lógica de predicción de saltos
- etc.



Intel Xeon 5500 (4 núcleos)

Ahorro de Energía

- Potencia dinámica : $P = P^Y + P^S + P^D$



$$P_{dgemm}(c) = 67.97 + 12.75\,c$$

Ahorro de Energía

- ACPI (*Advanced Configuration and Power Interface*): Interfaces estándar definidas por la industria que permite la gestión de potencia/temperatura

- Versión 5.0 (diciembre 2011)
- En el procesador: Power states (C-states) y performance states (P-states)

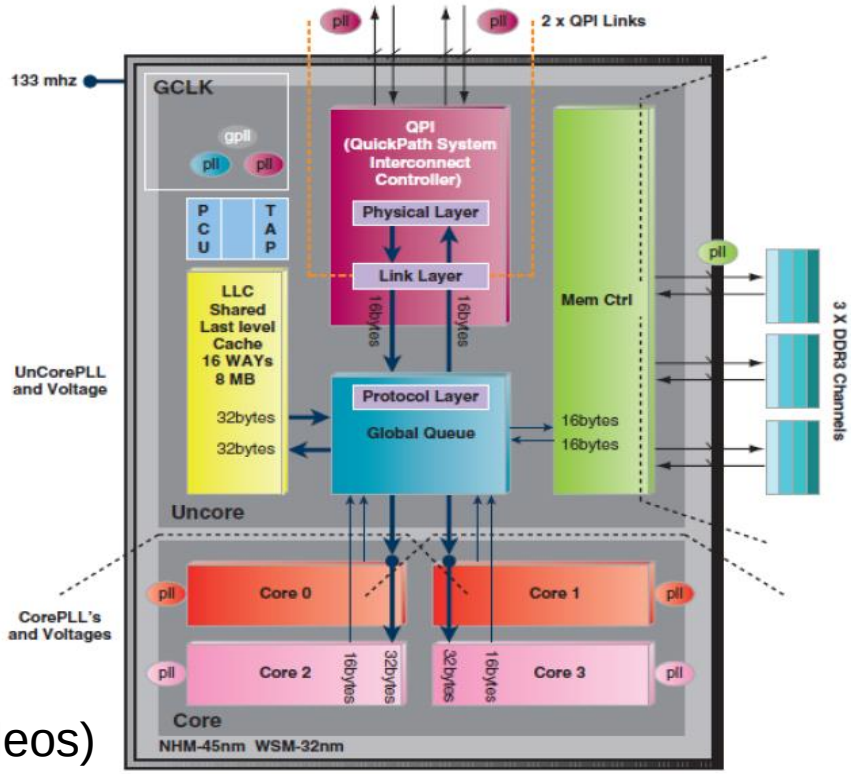
Ahorro de Energía

- Power states (C-states):
  - C0: ejecución normal (también un P-state)
  - Cx, x>0: sin instrucciones en ejecución. A medida que x crece, más ahorro pero más latencia para volver a C0
    - Parar la señal de reloj
    - Vaciar y parar la caché (L1 y L2 volcadas en LLC)
    - Detener núcleos

Ahorro de Energía

- Package power states (PC-states):
  - PC0, PC1, PC2,…

Subsistema uncore permanece activo y consume potencia mientras haya un núcleo active en la CPU



Intel Xeon 5500 (4 núcleos)

Ahorro de Energía

- Procesador Intel Core i7:
  - **Core C0 State**
    - Estado normal del núcleo cuando está ejecutando código
  - **Core C1/C1E State**
    - El núcleo se detiene; el protocolo de control de coherencia de cache se detiene
  - **Core C3 State**
    - El núcleo vacía la caché de instrucciones y datos L1, y la cache L2 sobre la caché compartida L3, pero mantiene los estados. Todos los relojes del núcleo sse detienen
  - **Core C6 State**
    - Antes de entrar en este estado, el núcleo salva los estados a una SRAM dedicada en el chip. Una vez completado, el núcleo reduce el voltaje a 0V

Ahorro de Energía

- Performance states (P-states):
  - P0: Potencia y rendimiento más altos
  - Pi, i>0: Mayor ahorro pero menor rendimiento con i

P-state P_i	VCC_i	f_i
P_0	1.23	2.00
P_1	1.17	1.50
P_2	1.12	1.20
P_3	1.09	1.00
P_4	1.06	0.80

AMD

  - $P = a\,V^2\,f$, donde a depende de la tecnología (pero $E = \int_0^T P\,dt = a\,V^2$)

⟶ DVFS!

Ahorro de Energía

- Aprovechar DVFS: cpufreq

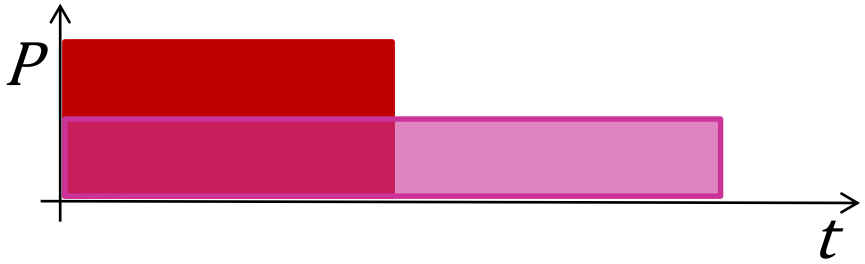
```
quintana@watts2:~$ cpufreq-info
analyzing CPU 15:
  driver: powernow-k8
  CPUs which run at the same hardware frequency: 15
  CPUs which need to have their frequency coordinated by software: 15
  maximum transition latency: 10.0 us.
  hardware limits: 800 MHz - 2.00 GHz
  available frequency steps: 2.00 GHz, 1.50 GHz, 1.20 GHz, 1000 MHz, 800 MHz
  available cpufreq governors: ondemand, userspace, performance
  current policy: frequency should be within 800 MHz and 2.00 GHz.
                  The governor "ondemand" may decide which speed to use
                  within this range.
  current CPU frequency is 800 MHz (asserted by call to hardware).
```

Ahorro de Energía

- Aprovechar DVFS (transparente): gobernadores Linux
  - **`Performance`**: Mayor frecuencia/rendimiento
  - **`Powersave`**: Menor frecuencia/rendimiento
  - **`Userspace`**: Decisión del usuario
  - **`Ondemand`**: Crecimiento rápido, disminución escalonada
  - **`Conservative`**: Crecimiento escalonado, disminución rápida

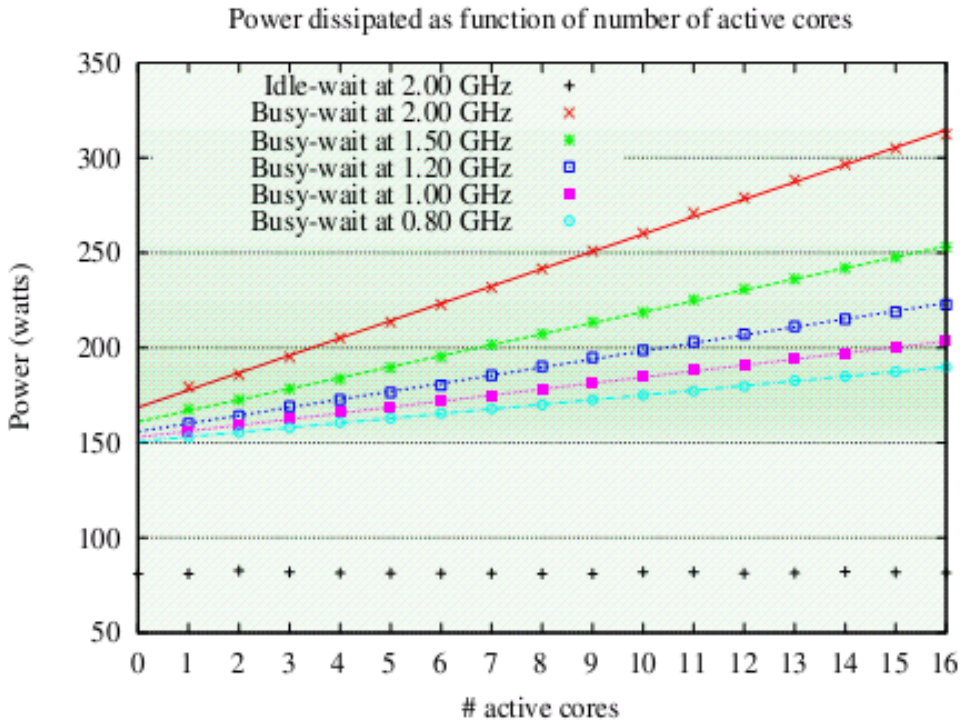
Ahorro de Energía

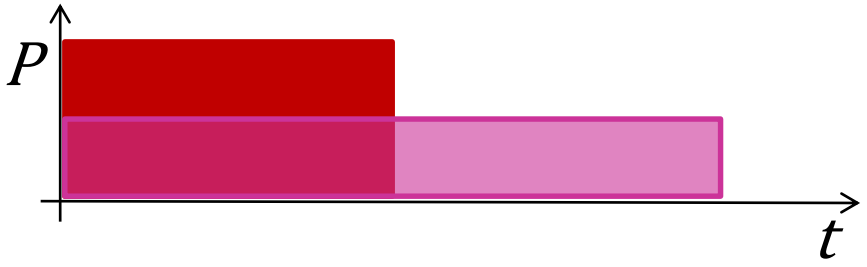
- ¿Qué es mejor, A o B?



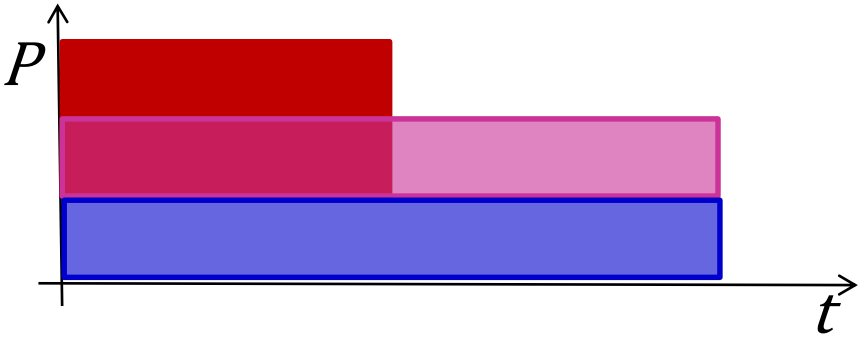
Ahorro de Energía

- ¿Qué es mejor, A o B?



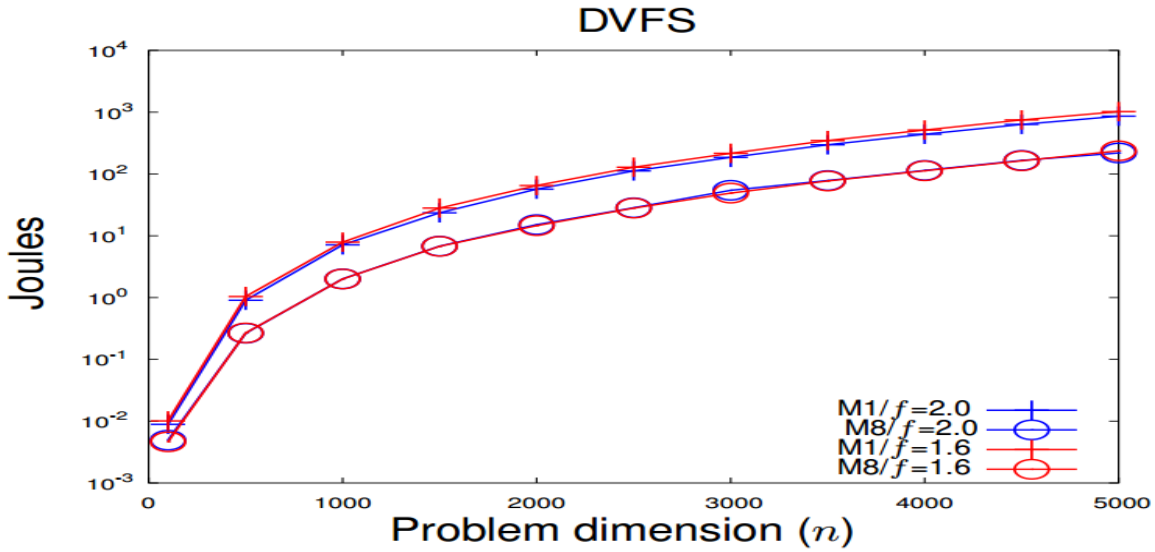


Pero considerar también
$P^Y + P^S \simeq$ 50% de potencia



Ahorro de Energía

- ¿DVFS o no? Consenso general:
  - No para aplicaciones intensivas en cálculo: reducir la frecuencia incremental el tiempo linealmente



  - Sí para aplicaciones intensivas en acceso a memoria, pues los núcleos están ociosos la mayor parte del tiempo

Ahorro de Energía

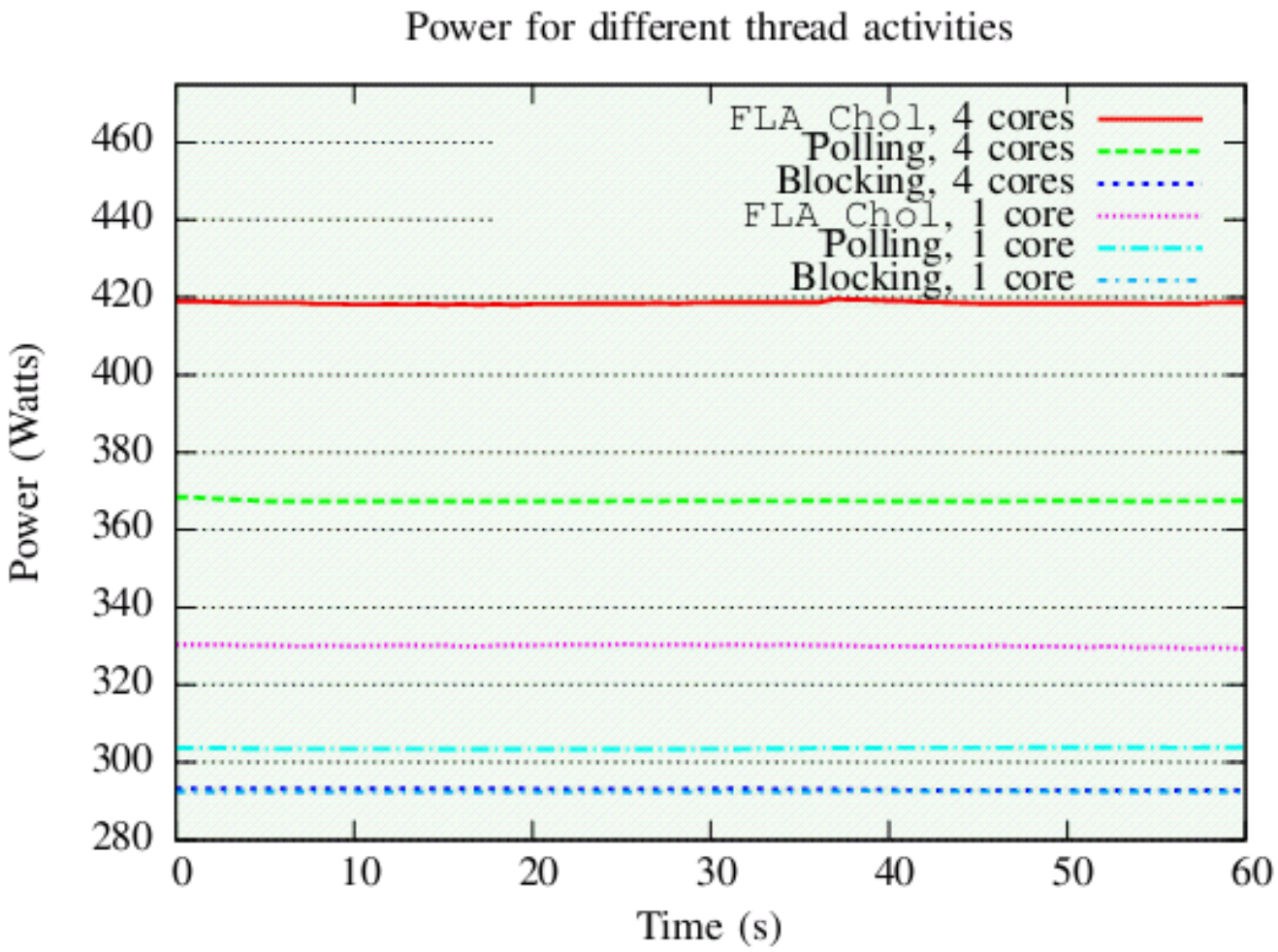
- …pero, en muchas plataformas, reducir la frecuencia mediante DVFS ¡también reduce el ancho de banda a memoria!

P-state P_i	VCC_i	f_i	α_i	β_i	ΔP_i^S	ΔP_i^D	$\Delta P_i^T(16)$	BW_i	ΔBW_i
P_0	1.23	2.00	168.59	9.12	–	–	–	4.43	–
P_1	1.17	1.50	161.10	5.77	-9.52	-32.14	-17.58	3.89	-12.19
P_2	1.12	1.20	155.90	4.23	-17.09	-50.25	-28.34	3.49	-21.21
P_3	1.09	1.00	152.94	3.15	-21.47	-60.73	-33.26	3.19	-27.99
P_4	1.06	0.80	150.61	2.44	-25.73	-70.30	-39.85	2.80	-36.79

Ahorro de Energía

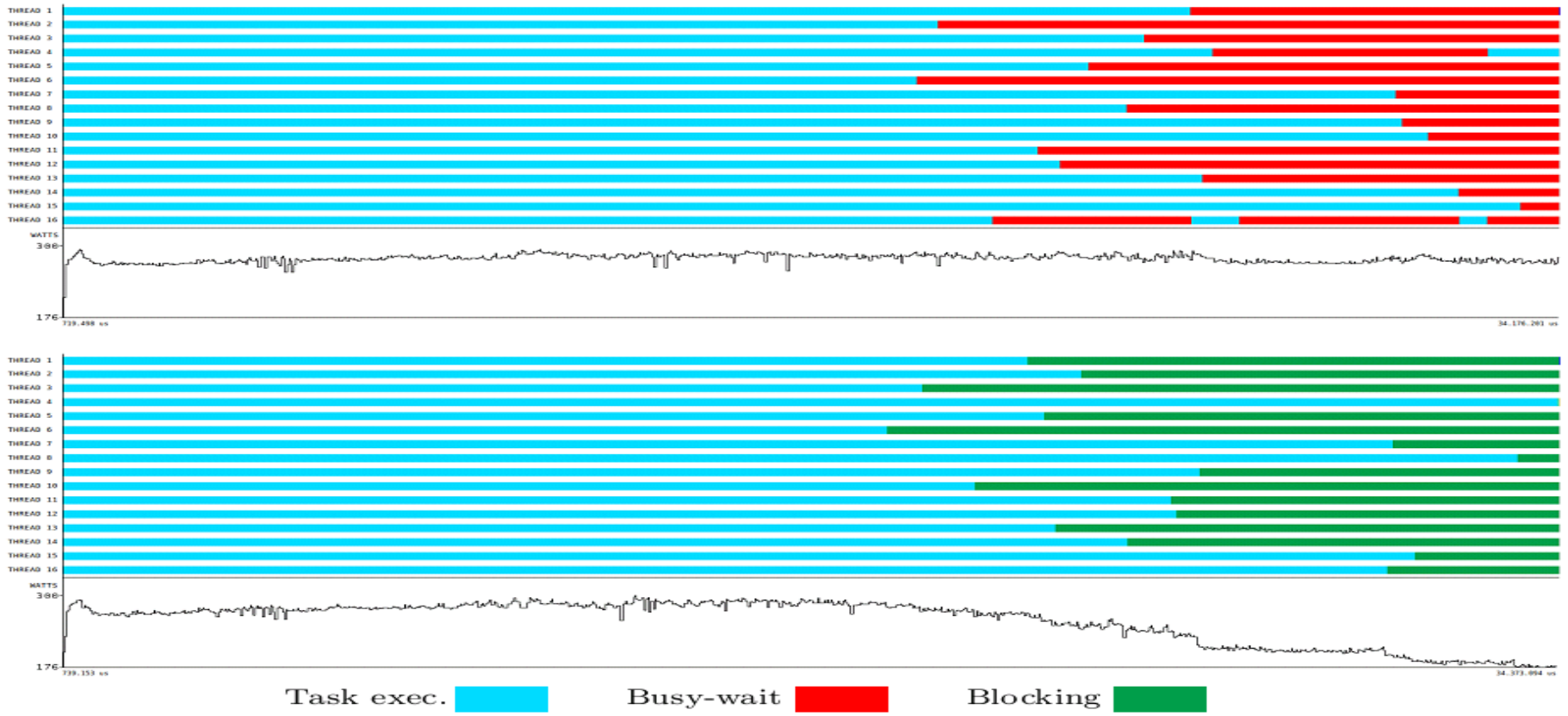
- Estrategias alternativas para aplicaciones intensivas en cálculo:
  - Idle-wait en aplicaciones multihebra
  - Idle-wait en aplicaciones híbridas CPU-GPU
  - Idle-wait durante las comunicaciones en aplicaciones MPI

Ahorro de Energía



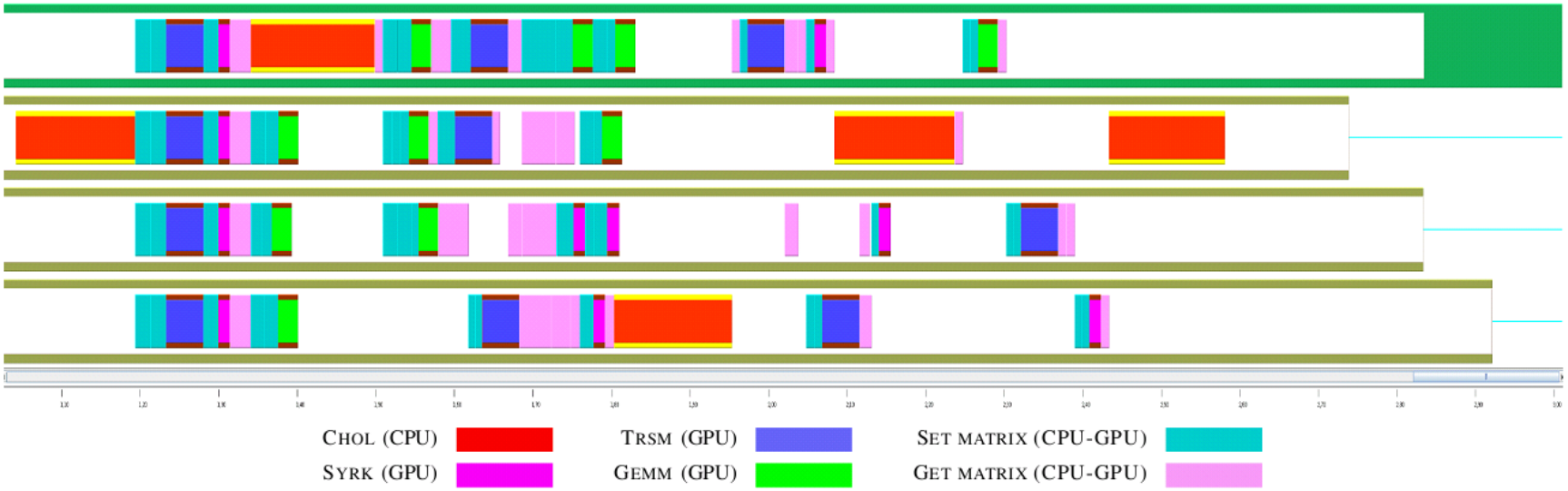
Ahorro de Energía

- Idle-wait en aplicaciones multihebra (precondicionador ILU)



Cost of energy
Saving opportunities

- Idle-wait en aplicaciones híbridas CPU-GPU (factorización de Cholesky multiGPU via runtime)



  - Intel Xeon E5540 @ 2.83 GHz (4 núcleos) y NVIDIA Tesla S2050 (4 “Fermis”)

Sumario

- Una batalla compleja por luchar por el rendimiento:
  - Más concurrencia
  - Diseños heterogéneos

- Una batalla relacionada en el ámbito del consumo
  - Do nothing, efficiently… (V. Pallipadi, A. Belay)

…¡pero no siempre hay que creer al vendedor!