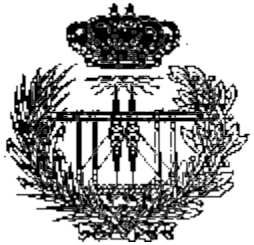


Optimización y Paralelización del Software para el Cálculo de Campos Electromagnéticos MEATSS

José Ginés Picón López

Director:

Vicente Boria Esbert

Subdirectores:

Alejandro Álvarez Melcón

Fernando Quesada Pereira

Domingo Giménez Cánovas

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Índice

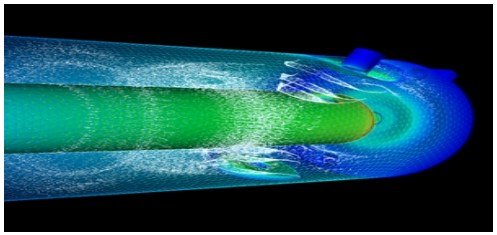
- Introducción
- Herramientas Computacionales
- Utilización de la Librería MKL
- Paralelismo con OpenMP
- Paralelismo Híbrido OpenMP+MKL
- Conclusiones y Trabajo Futuro

Índice

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Efectos de Corona y Multipactor

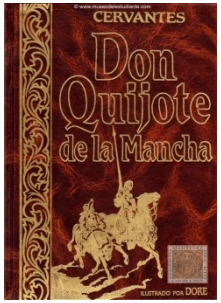
- Fenómenos físicos inherentes al comportamiento de los electrones en presencia de campos electromagnéticos intensos producidos por niveles de potencia muy elevados.
- Daños en los dispositivos electrónicos.
- Altos niveles de potencia necesarios en comunicaciones espaciales.
- Necesidad de simulaciones.



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

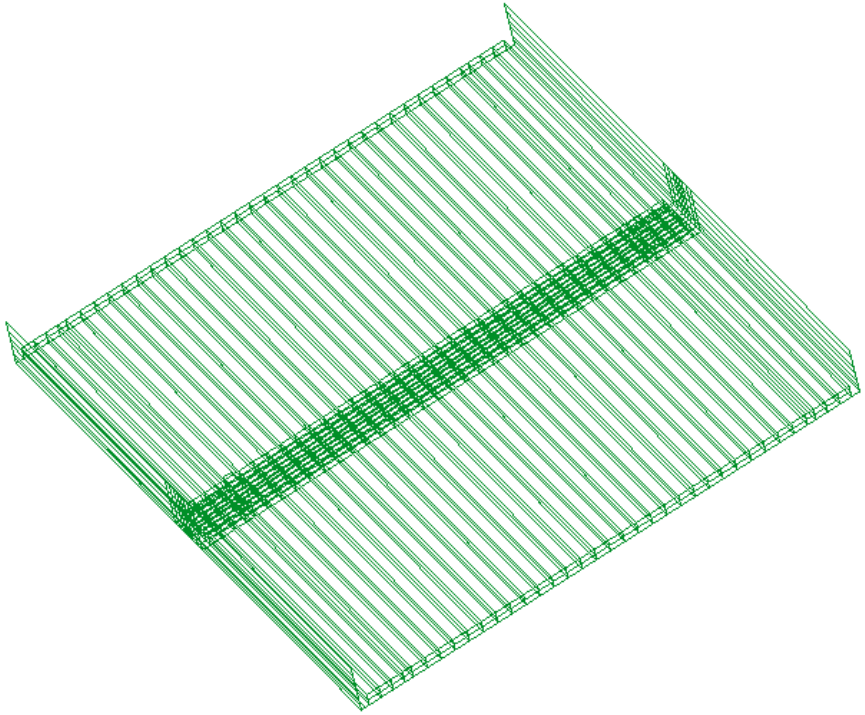
Software de Partida: MEATSS

- Software basado en la ecuación integral.
- Desarrollado por el grupo GEAT de UPCT.
- Código en C++ y Fortran-90.
- Grandes sistemas lineales de ecuaciones complejas densas.
- Análisis de los parámetros de dispersión.



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Software de Partida: MEATSS



Modelo 3D utilizado en las simulaciones

Objetivos

- Reducir tiempos de ejecución.
- Aumentar complejidad.
- Adaptación a sistemas de altas prestaciones.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Índice

- Introducción
- **Herramientas Computacionales**
- Utilización de la Librería MKL
- Paralelismo con OpenMP
- Paralelismo Híbrido OpenMP+MKL
- Conclusiones y Trabajo Futuro

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Herramientas Computacionales

- Herramientas Software:
  - Profiler: gprof
  - Debugger: idb
  - Compiladores Intel: icpc, ifort
  - Librería MKL
  - OpenMP
- Herramientas Hardware:
  - Ben Arabí
  - Saturno

gprof

- Información temporal post-mortem.
- Pasos:
  - Compilar y linkar con los flags –p y –g
  - Ejecutar el software y obtener *gmon.out*
  - Obtener los profiles con el comando *gprof*
  - *gprof opciones [ejecutable] gmon.out*
- Flat y Graph Profile

Flat Profile

Flat profile:

Each sample counts as 0.01 seconds.

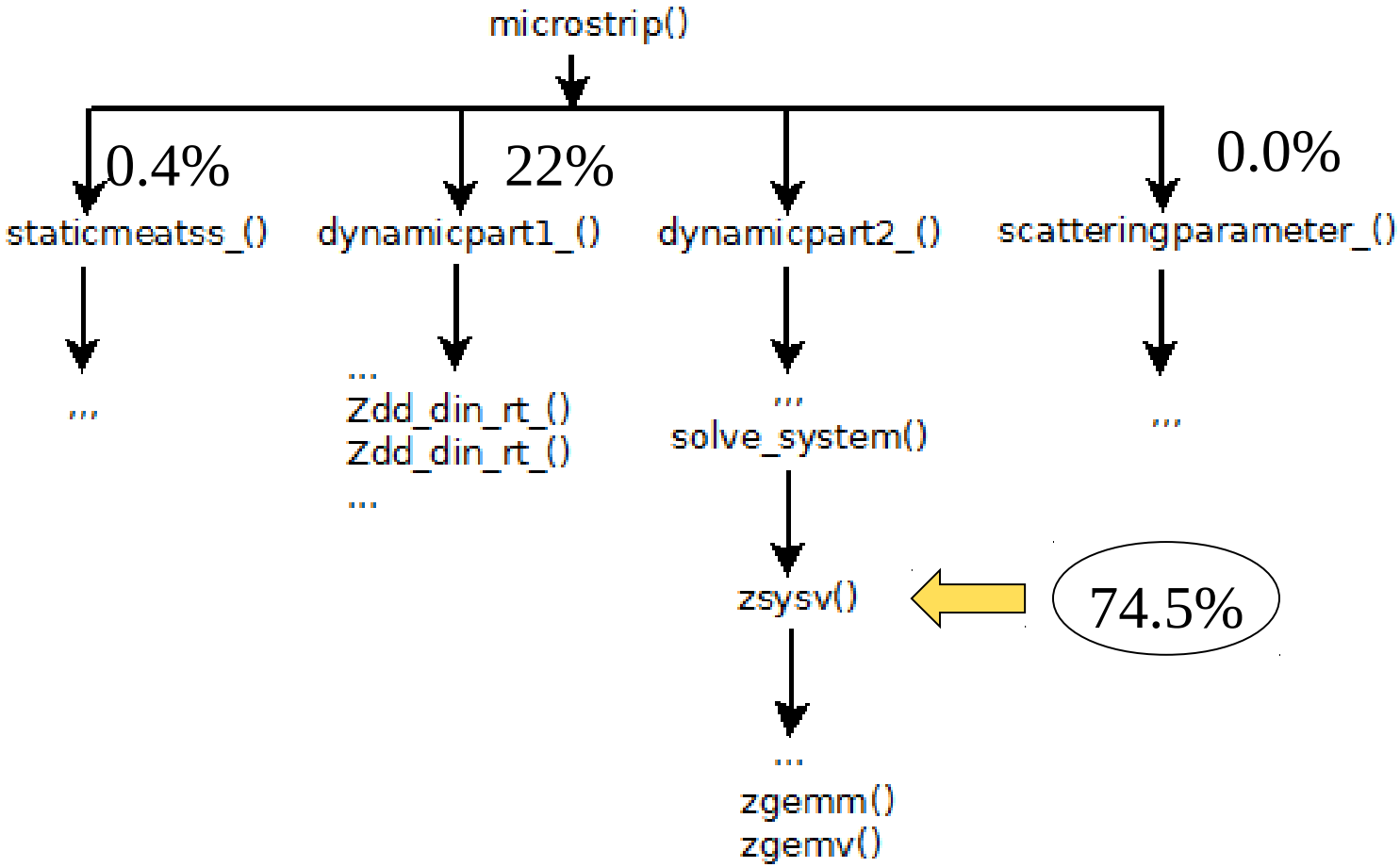
% time	cumulative seconds	self seconds	calls	self Ks/call	total Ks/call	name
66.24	6380.83	6380.83	672384	0.00	0.00	zgemm_
10.50	7392.02	1011.19	128520612	0.00	0.00	zdd_din_rt_
7.40	8104.89	712.87	45524709	0.00	0.00	zgemv_
4.53	8541.72	436.83	159306048	0.00	0.00	zrd_din_rt_
3.15	8844.81	303.09	1584114134	0.00	0.00	quad3drec_

Call Graph Profile

```
  index % time    self  children    called     name
                                                <spontaneous>
[1]       96.9    0.00 9330.85                 main [1]
                  0.00 9330.85       1/1           microstrip(int, char* const*) [2]
-----------------------------------------------
                  0.00 9330.85       1/1           main [1]
[2]       96.9    0.00 9330.85         1         microstrip(int, char* const*) [2]
                  0.00 7173.44     204/204         dynamicpart2_ [3]
                 68.15 2054.11     204/204         dynamicpart1_ [9]
                  0.00   35.13       1/1           staticmeatss_ [23]
                  0.00    0.01     204/204         scatteringparameter_ [84]
                  0.00    0.00       1/1           Read_file(char const*, double&, ... int&)
-----------------------------------------------
                  0.00 7173.44     204/204         microstrip(int, char* const*) [2]
[3]       74.5    0.00 7173.44     204           dynamicpart2_ [3]
                  0.00 7173.39     204/204         solve_system_ [5]
                  0.01    0.04     204/204         excitation_evaluation_ [71]
-----------------------------------------------
                  0.00 7173.39     408/408         solve_system_ [5]
[4]       74.5    0.00 7173.39     408           zsysv_ [4]
                  0.01 7135.44     204/204         zsytrf_ [6]
                  0.03   37.91     204/204         zsytrs_ [21]
                  0.00    0.00     816/277234284     lsame_ [50]
                  0.00    0.00     408/612         ilaenv_ [109]
```

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Árbol de llamadas

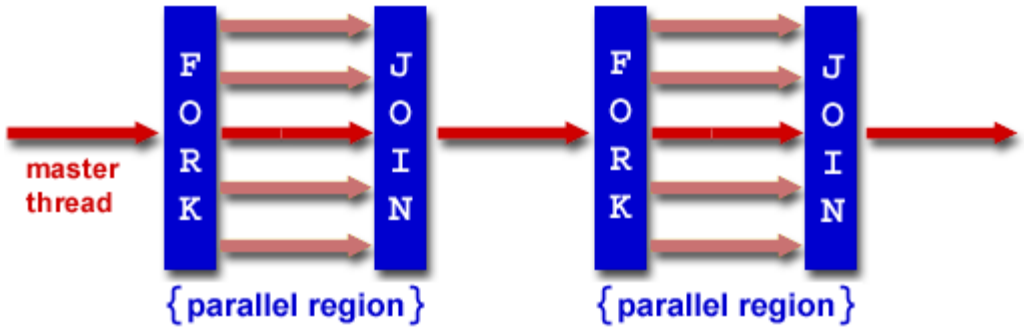


Intel Math Kernel Library (MKL)

- Altamente Optimizada y Multithread.
- Modelo de capas:
  - Interface layer
  - Threading layer
  - Computational layer
  - Run time libraries
- MKL_NUM_THREADS.
- Linkado versión secuencial en superdome:
  - lmkl_intel_lp64, lmkl_sequential, lmkl_core, lpthread

OpenMP

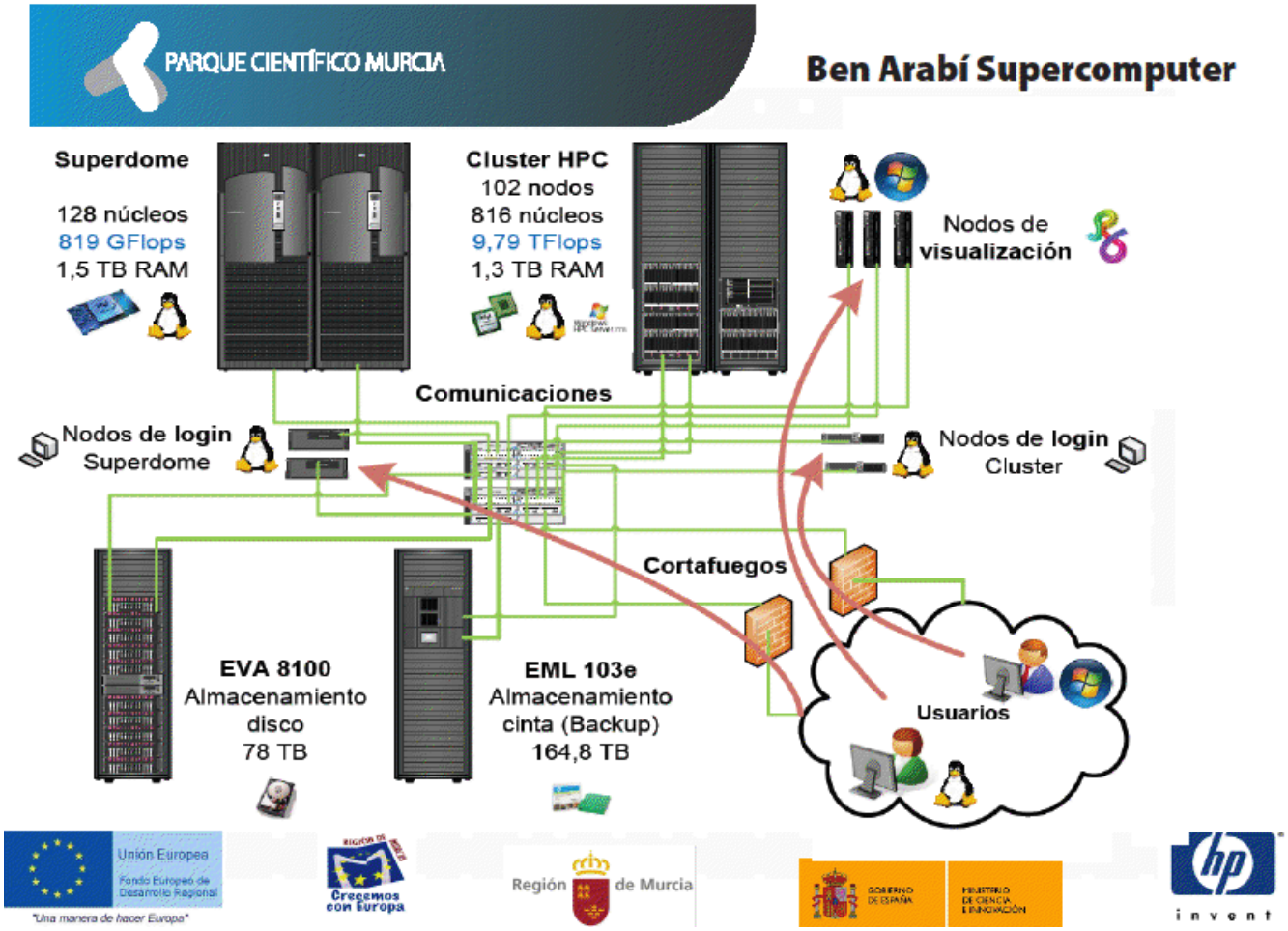
- Paralelismo memoria compartida.
- Modelo fork-join:



- OMP_NUM_THREADS.
- Flag –openmp.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Supercomputador Ben Arabí



Load Sharing Facility (LSF)

```
#!/bin/bash

#BSUB -J nombre_jobs # Nombre del trabajo
#BSUB -o salida.%J.out # Fichero de salida
#BSUB -e error.%J.err # Fichero de error
#BSUB -q arabi_8x24h # Nombre de la cola
#BSUB -n 8 # Nº de cores reservados

source /etc/profile.d/modules.sh
module load intel # Carga de módulos

export MKL_NUM_THREADS=8  # Número de threads

time ./meatss_scattering
```

Índice

Estructura del Código

Algorithm 1 Bucle en frecuencia de MEATSS

staticpart
for $i = 0 \rightarrow num\_freq$ **do**
    $freq = init\_freq + i * step$
    dynamicpart1($freq$)
    dynamicpart2
    scatteringparameters
end for

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Estudio Experimental

Tabla 3.1: Tiempos (en segundos) de `staticpart`, `dynamicpart1` y `dynamicpart2` y coste relativo de cada parte, en Arabí para mallados de distinta complejidad

	sta-par		dyn-par1		dyn-par2	
	tiempo	%	tiempo	%	tiempo	%
simple	7.6	89.6	0.65	7.7	0.23	2.7
media	106.49	76.6	12.5	8.9	21.77	15.5
compleja	70.31	40.4	31.1	17.9	72.72	41.8

Tabla 3.2: Tiempos (en segundos) de `staticpart`, `dynamicpart1` y `dynamicpart2` y coste relativo de cada parte, en Ben para mallados de distinta complejidad

	sta-par		dyn-par1		dyn-par2	
	tiempo	%	tiempo	%	tiempo	%
simple	10.82	77.8	2.37	17.1	0.71	5.1
media	145.53	56.6	44.78	17.4	67.13	26.1
compleja	149.54	31.9	95.32	20.3	223.6	47.3

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Estudio Experimental: MKL Secuencial

Tabla 3.3: Tiempos (en segundos) de `dynamicpart2` con código insertado y con llamada a `zsysv` de MKL, y speed-up conseguido con el uso de MKL, en Arabí para mallados de distinta complejidad

	insertado	MKL	speed-up
simple	0.23	0.07	3.28
media	21.77	5.90	3.69
compleja	72.72	19.51	3.73

Tabla 3.4: Tiempos (en segundos) de `dynamicpart2` con código insertado y con llamada a `zsysv` de MKL, y speed-up conseguido con el uso de MKL, en Ben para mallados de distinta complejidad

	insertado	MKL	speed-up
simple	0.71	0.12	5.92
media	67.13	10.43	6.44
compleja	223.60	34.17	6.54

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Estudio Experimental: MKL Paralelo

Tabla 3.5: Tiempos de ejecución (en segundos) de `dynamicpart2` con MKL, en Arabí variando el número de *cores*

Mallado	Sec	2 c	4 c	8 c
Simple	0.07	0.07	**0.05**	0.09
Media	5.90	3.61	2.47	**2.00**
Compleja	19.51	11.96	7.02	**5.64**

Tabla 3.6: Tiempos de ejecución (en segundos) de `dynamicpart2` con MKL, en Ben variando el número de *cores*

Mallado	Sec	2 c	4 c	8 c	16 c	24 c	32 c
Simple	0.12	0.10	0.08	**0.07**	**0.07**	0.08	0.08
Media	10.43	6.68	4.39	3.23	2.70	**2.66**	2.69
Compleja	34.17	20.75	12.55	8.76	7.18	6.89	**6.58**

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Estudio Experimental: Rutina no simétrica

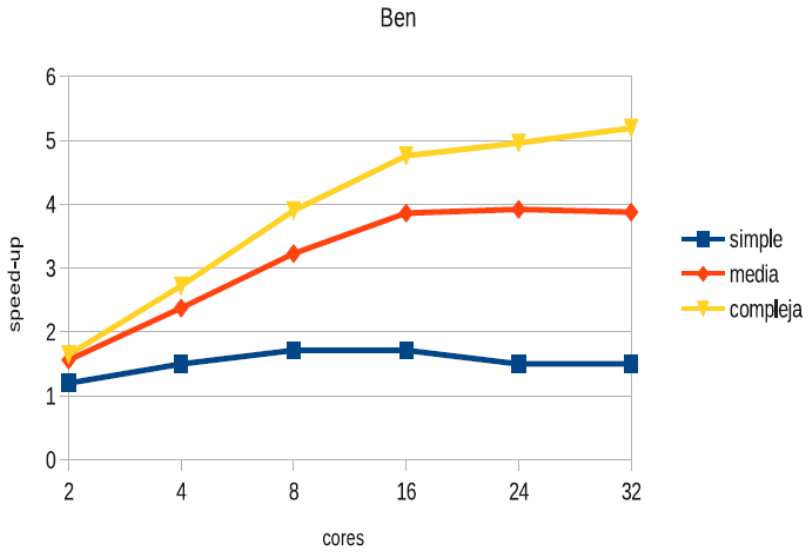


Figura 3.2: *Speed-up* de la rutina `zsysv` de MKL en Ben

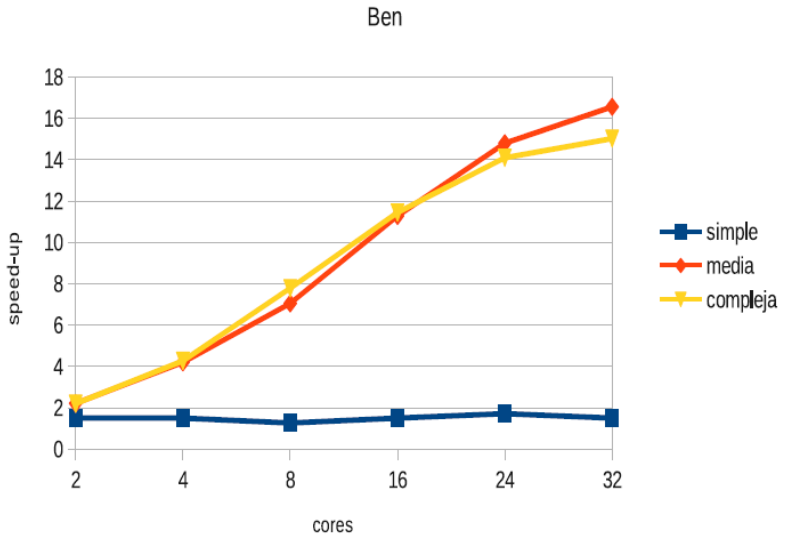


Figura 3.4: *Speed-up* de la rutina `zgesv` de MKL en Ben

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Estudio Experimental: Comparativa

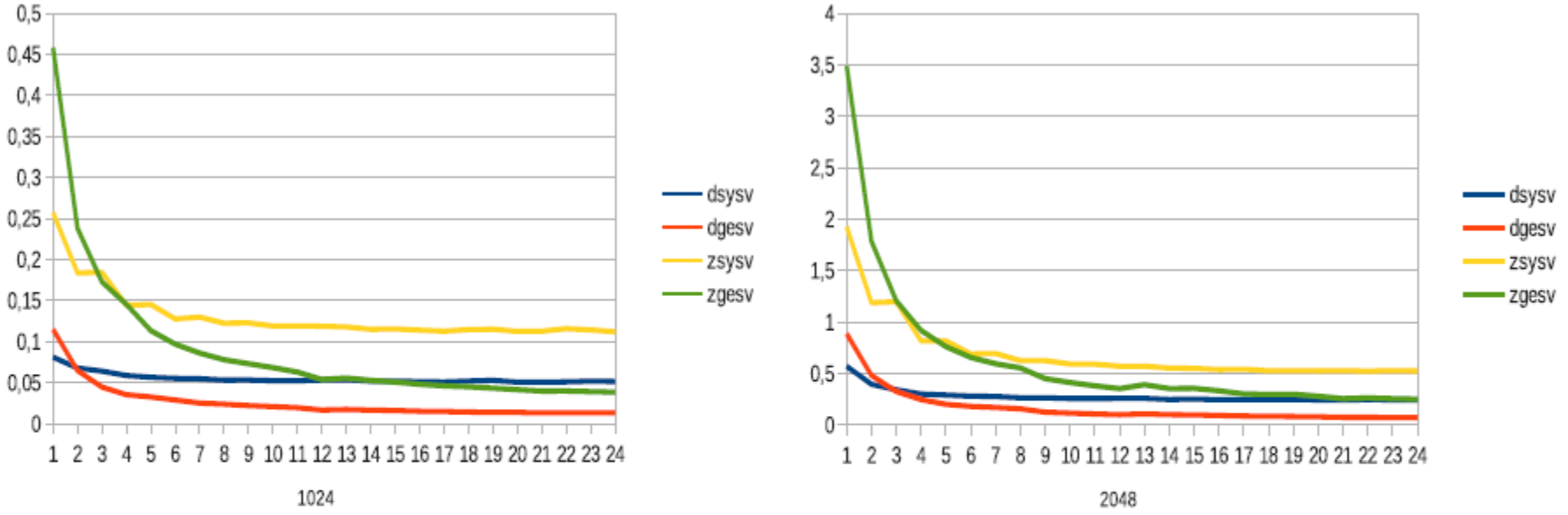


Figura 3.5: Comparación del tiempo de ejecución de rutinas de resolución de sistemas de ecuaciones lineales, variando el número de *cores*, para matrices de tamaño 1024 (izquierda) y 2048 (derecha), en Saturno

Índice

- Introducción
- Herramientas Computacionales
- Utilización de la Librería MKL
- **Paralelismo con OpenMP**
- Paralelismo Híbrido OpenMP+MKL
- Conclusiones y Trabajo Futuro

Reestructuración del Código

- Para cada subrutina. Si una variable global es escrita se crea una copia local que reemplazará a la variable global.
- Si se lee alguna variable global modificada por otra subrutina antecesora, deben modificarse todas las subrutinas desde esta antecesora hasta la que lee la variable. Se realizará una copia local de la variable en la subrutina antecesora y se pasará como parámetro.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

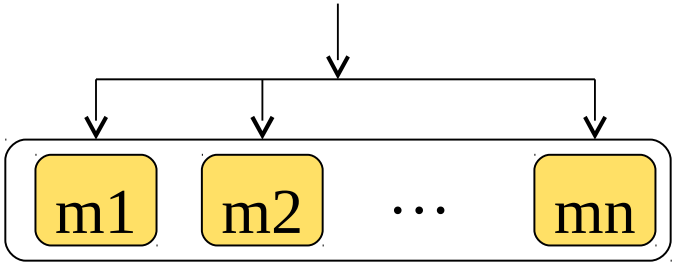
Reestructuración del Bucle en Frecuencias

Algorithm 2 Reestructuración del bucle en frecuencia de MEATSS en 3 bucles

for $i = 0 \rightarrow num\_freq$ **do**
    fillmatrix(i,init_freq,step)
end for

for $i = 0 \rightarrow num\_freq$ **do**
    solvesystem(i)
end for

for $i = 0 \rightarrow num\_freq$ **do**
    circuitalparameters(i,init_freq,step)
end for



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Bucle de Resolución de Sistemas

Algorithm 4 Bucle de resolución de sistemas con OpenMP

```
# pragma omp parallel for private(i)
for i = 0 → num_freq do
    solvesystem(i,init_freq,step)
end for
```

- OMP_NESTED=FALSE
- MKL_DINAMIC=FALSE
- MKL_NUM_THREADS=1
- OMP_NUM_THREADS=XX

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Estudio Experimental

Tabla 4.1: Tiempos de ejecución (en segundos) del bucle de resolución de sistemas con llamadas a `zsysv` de MKL, en Arabí para 8 frecuencias con OpenMP

Mallado	2 c	4 c	8 c
Simple	1.43	0.99	**0.88**
Media	-	-	**10.78**
Compleja	-	-	**33.47**

Tabla 4.2: Tiempos de ejecución (en segundos) del bucle de resolución de sistemas con llamadas a `zsysv` de MKL, en Ben para 128 frecuencias con OpenMP

Mallado	16 c	32 c	48 c	56 c	64 c	96 c	128 c
Simple	3.76	2.90	**1.93**	-	3.79	-	-
Media	-	-	59.41	61.90	**45.25**	81.6	91.55
Compleja	-	-	-	-	**123.54**	171.25	-

Ejemplo Numérico en Superdome para 128 Frecuencias

Tabla 4.4: Tiempos en segundos para el cálculo de 8 frecuencias en el caso de utilizar MKL secuencial, OpenMP con el número óptimo de *threads* y *speed-up*, en Arabí para mallados de distinta complejidad

Mallado	MKL sec	OpenMP	speed-up
Simple	0.56	0.88	0.63
Media	47.2	10.78	4.38
Compleja	156.08	33.47	4.66

Tabla 4.5: Tiempos en segundos para el cálculo de 128 frecuencias en el caso de utilizar MKL secuencial, OpenMP con el número óptimo de *threads* y *speed-up*, en Ben para mallados de distinta complejidad

Mallado	MKL sec	OpenMP	speed-up
Simple	15.36	1.93	7.95
Media	1335.04	45.25	29.5
Compleja	4373.76	123.54	35.4

- Código original: 229.6 s x 128 = 29620 s.
- OpenMP: 123 s (232x).

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Índice

- Introducción
- Herramientas Computacionales
- Utilización de la Librería MKL
- Paralelismo con OpenMP
- **Paralelismo Híbrido OpenMP+MKL**
- Conclusiones y Trabajo Futuro

Paralelismo Híbrido

Algorithm 5 Paralelismo híbrido en MEATSS con 3 bucles

```
omp_set_nested(1)
mkl_set_dynamic(0)
omp_set_num_threads(ntomp)
mkl_set_num_threads(ntmkl)

for i = 0 → num_freq do
    fillmatrix(i,init_freq,step)
end for

# pragma omp parallel for private(i)
for i = 0 → num_freq do
    solvesystem(i)
end for

for i = 0 → num_freq do
    circuitalparameters(i,init_freq,step)
end for
```

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Estudio Experimental con zsysv

Tabla 5.1: Tiempos en segundos para el cálculo de 8 frecuencias utilizando programación híbrida OpenMP+MKL con subrutina `zsysv` en Arabí para mallados de distinta complejidad

Mallado	8-1	4-2	2-4	1-8
Simple	0.99	**0.36**	0.48	0.84
Media	**11.82**	11.91	16.13	24.14
Compleja	**33.57**	35.29	42.20	65.10

Tabla 5.2: Tiempos en segundos para el cálculo de 128 frecuencias utilizando programación híbrida OpenMP+MKL con subrutina `zsysv` en Ben para mallados de distinta complejidad

Mallado	64-1	32-2	16-4
Simple	3.08	**2.22**	2.85
Media	**48.53**	63.66	97.65
Compleja	**114.68**	152.91	241.79

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Estudio Experimental con zgesv

Tabla 5.3: Tiempos en segundos para el cálculo de 8 frecuencias utilizando programación híbrida OpenMP+MKL con subrutina `zgesv` en Arabí para mallados de distinta complejidad

Mallado	8-1	4-2	2-4	1-8
Simple	1.09	**0.76**	1.41	2.21
Media	**15.50**	20.14	25.40	34.07
Compleja	**46.25**	53.56	66.75	-

Tabla 5.4: Tiempos en segundos para el cálculo de 128 frecuencias utilizando programación híbrida OpenMP+MKL con subrutina `zgesv` en Ben para mallados de distinta complejidad

Mallado	64-1	32-2	16-4
Simple	4.94	**4.93**	6.12
Media	96.49	**81.36**	89.04
Compleja	222.01	**171.42**	193.46

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Índice

- Introducción
- Herramientas Computacionales
- Utilización de la Librería MKL
- Paralelismo con OpenMP
- Paralelismo Híbrido OpenMP+MKL
- **Conclusiones y Trabajo Futuro**

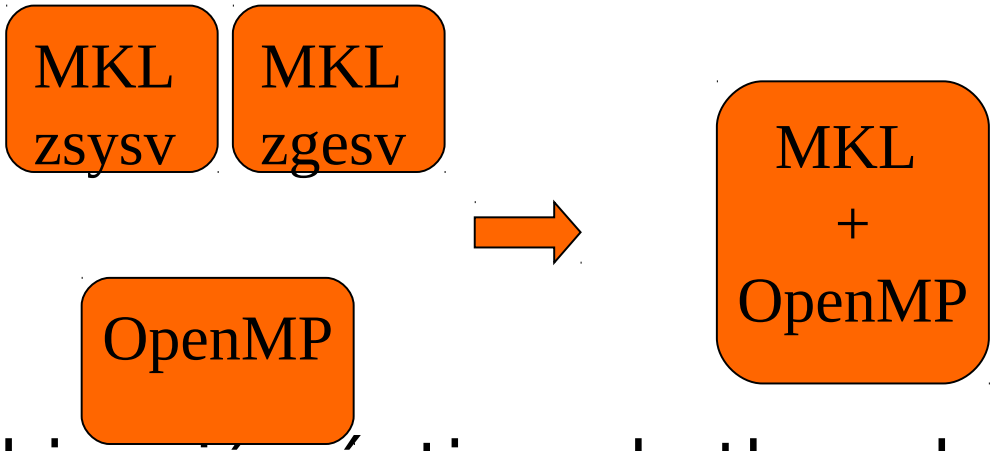
Conclusiones

- Reducción significativa del tiempo.
- Uso eficiente de grandes sistemas.
- Mejoras especialmente en mallados complejos.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Trabajo Futuro: Autooptimización

- Diferentes versiones de MEATSS:



- Combinación óptima de threads y rutinas dependiendo del problema, frecuencias, sistema…

Trabajo Futuro

- Técnicas de autooptimización.
- Optimizar y paralelizar otros módulos.
- Geometrías más complejas.
- Alternativas a MKL: ATLAS, PLASMA, ...
- Uso de la GPU mediante CUDA.
- Paralelismo por paso de mensajes MPI.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

¿Preguntas?