Grado en Ingeniería Informática

Trabajo Fin de Grado

Optimización de Rutinas de Álgebra Lineal en Sistemas Heterogéneos Multicore+MultiMIC

Jesús Cámara Moreno

Murcia, 14 de Septiembre de 2016

Índice de Contenidos

Índice de Contenidos

1. Introducción

- Motivación: evaluar las prestaciones obtenidas por la rutina de multiplicación de matrices en un sistema heterogéneo con varios coprocesadores Intel Xeon Phi.

- Multitud de problemas científicos y de ingeniería son resueltos utilizando rutinas matriciales de álgebra lineal cuyo núcleo computacional básico es la rutina de multiplicación de matrices

- Optimizar esta rutina es clave si se quiere explotar de forma eficiente el paralelismo ofrecido por la arquitectura del sistema y obtener el máximo rendimiento.

- El proceso de optimización depende del sistema computacional → adaptar el diseño de la rutina y las técnicas de optimización

1. Introducción

- La capacidad computacional requerida para resolver de forma rápida y eficiente problemas de gran dimensión con alto coste computacional → desarrollo de arquitecturas paralelas que integran elevado número de cores en un solo chip.

- La inclusión de arquitecturas *manycore* en sistemas *multicore* de memoria compartida → Sistemas Heterogéneos.

- Objetivo: realizar la mejor distribución de la carga de trabajo entre los componentes computacionales del sistema para cada tamaño de problema, utilizando la mejor versión del algoritmo y el valor óptimo de sus parámetros algorítmicos.

Índice de Contenidos

2. Entorno de Trabajo

- Venus: nodo de un clúster heterogéneo. Dispone de 64 GB de memoria y está compuesto por dos procesadores hexa-core Intel Xeon E5-2620 a 2.4 GHz conectados por medio del bus PCIe a dos coprocesadores Intel Xeon Phi 3120 con 6 GB de memoria y un total de 57 cores a 1.1GHz.

  Cada core del coprocesador:
  - 4 threads hardware.
  - Unidad escalar con doble emisión de instrucciones en orden.
  - Soporta modelo de procesamiento paralelo SIMD mediante el uso de una VPU de 512 bits de ancho → 16 SPFP / 8DPFP por ciclo.

  Aspectos a explotar del sistema: escalabilidad, vectorización y uso de memoria.


Machine (64GB total)
NUMANode P#0 (32GB)
Package P#0
L3 (15MB)
L2 (256KB)
L2 (256KB)
L2 (256KB)
L2 (256KB)
L2 (256KB)
L2 (256KB)
L1d (32KB)
L1d (32KB)
L1d (32KB)
L1d (32KB)
L1d (32KB)
L1d (32KB)
L1i (32KB)
L1i (32KB)
L1i (32KB)
L1i (32KB)
L1i (32KB)
L1i (32KB)
Core P#0
PU P#0
Core P#1
PU P#1
Core P#2
PU P#2
Core P#3
PU P#3
Core P#4
PU P#4
Core P#5
PU P#5
8.0
8.0
PCI 8086:225d
mic0
57 cores
5952 MB
NUMANode P#1 (32GB)
Package P#1
L3 (15MB)
L2 (256KB)
L2 (256KB)
L2 (256KB)
L2 (256KB)
L2 (256KB)
L2 (256KB)
L1d (32KB)
L1d (32KB)
L1d (32KB)
L1d (32KB)
L1d (32KB)
L1d (32KB)
L1i (32KB)
L1i (32KB)
L1i (32KB)
L1i (32KB)
L1i (32KB)
L1i (32KB)
Core P#0
PU P#6
Core P#1
PU P#7
Core P#2
PU P#8
Core P#3
PU P#9
Core P#4
PU P#10
Core P#5
PU P#11
8.0
8.0
PCI 8086:225d
mic1
57 cores
5952 MB

Índice de Contenidos

3. Técnicas de Optimización

- Aspectos a optimizar: *vectorización* y *uso de memoria.*

- Vectorización:
  - Opciones de compilación: `-openmp-simd`
  - Directivas: `#pragma ivdep`, `#pragma vector aligned`, `#pragma vector always`, `#pragma simd`

- Uso de Memoria:
  - Alineamiento de Datos: `_mm_alloc()`, `_assume_aligned()`
  - Opciones de Compilación: `-opt-dynamic-align`, `-opt-prefetch`
  - Directivas: `#pragma prefetch`

- Otras opciones de compilación: `-ip`, `-xhost`, `-parallel`, `-openmp`, `-opt-threads-per-core`.

Índice de Contenidos

4. Implementación de la Rutina

- Lenguaje C + OpenMP + Directivas para Vectorización.
- Diseño parametrizado (n, tb, $nThr$) → Portabilidad.
- Tipos de Implementación: con y sin bloques.

```
void doMatMul(...,n,tb)
{
    for(...) {
        for(...) {
            //Directivas Vectorización
            for(...) {
                //Producto de Elementos
            }
        }
    }
}
```

```
void doMatMulBloq(...,n,tb,nThr)
{
    omp_set_num_threads(nThr);
    #pragma omp parallel for
    for(...) {
        for(...) {
            for(...) {
                doMatMul(...);
            }
        }
    }
}
```

Esquema Algorítmico por Bloques

Índice de Contenidos

5. Evaluación de Prestaciones

- Objetivo: estudiar el comportamiento de la rutina con distintas optimizaciones y tamaños de problema cuando se modifica el valor de sus parámetros algorítmicos y se distribuye la carga de trabajo entre los componentes computacionales del sistema.

- Host Multicore y Coprocesador MIC:
  - Optimización del Compilador (con autovectorización) + OpenMP.
  - Opciones de Compilación + Directivas Vectorización + OpenMP.
  - Influencia del Tamaño de Bloque y de la Afinidad (asignación de threads a cores)

- Híbrido (*host + coprocesadores*):

  Multicore + monoMIC ; Multicore + multiMIC

5. Evaluación de Prestaciones

- Evaluación en Multicore Intel Xeon

	Auto-Vectorización (-O3)			Vectorización + Flags		
N	**Threads**	**Tiempo**	**GFlops**	**Threads**	**Tiempo**	**GFlops**
1152	12	0.0434	70.40	12	0.0335	91.36
1920	12	0.6530	21.68	12	0.6024	23.50
2688	12	1.8370	21.15	12	1.7192	22.59
3456	12	3.9438	20.93	12	3.6918	22.36
4224	12	7,2293	20.85	12	6.7665	22.28

Tabla 1. Prestaciones de la rutina en el *host* sin uso de bloques.

	Auto-Vectorización (-O3)				Vectorización + Flags			
N	**Threads**	**N_{bloque}**	**Tiempo**	**GFlops**	**Threads**	**N_{bloque}**	**Tiempo**	**GFlops**
1152	12	96	0.0460	66.40	12	96	0.0260	117.84
1920	12	32	0.2178	64.97	12	80	0.1199	118.02
2688	12	112	0.5442	71.38	12	112	0.3131	124.04
3456	12	72	1.2303	67.10	12	96	0.6846	120.59
4224	12	88	2.1675	69.54	12	128	1.3345	112.95

Tabla 2. Prestaciones de la rutina en el *host* con uso de bloques.

5. Evaluación de Prestaciones

- Evaluación en Coprocesador Intel Xeon Phi *(-mmic)*

	Auto-Vectorización (-O3)			**Vectorización + Flags**		
N	**Threads**	**Tiempo**	**GFlops**	**Threads**	**Tiempo**	**GFlops**
1152	228	0.0805	37.98	168	0.0359	85.20
1920	228	0.2245	63.03	224	0.1768	80.08
2688	200	0.5502	70.60	204	0.5240	74.13
3456	180	1.2045	68.54	172	1.1618	71.06
4224	152	2.5450	59.23	176	2.2334	67.49

Tabla 3. Prestaciones de la rutina en MIC sin uso de bloques.

	Vectorización + Flags			
N	**Threads**	**N_{bloque}**	**Tiempo**	**GFlops**
1152	48	32	0.1688	18.11
1920	64	32	0.5659	25.02
2688	84	32	1.1527	33.70

Tabla 4. Prestaciones de la rutina en MIC con uso de bloques.

5. Evaluación de Prestaciones

- Procesador Intel Xeon vs Coprocesador Intel Xeon Phi

	Intel Xeon			Intel Xeon Phi		
N	**Threads**	**Tiempo**	**GFlops**	**Threads**	**Tiempo**	**GFlops**
1152	12	0.0335	91.36	168	0.0359	85.20
1920	12	0.6024	23.50	224	0.1768	80.08
2688	12	1.7192	22.59	204	0.5240	74.13
3456	12	3.6918	22.36	172	1.1618	71.06
4224	12	6.7665	22.28	176	2.2334	67.49

Tabla 5. Prestaciones obtenidas sin uso de bloques.

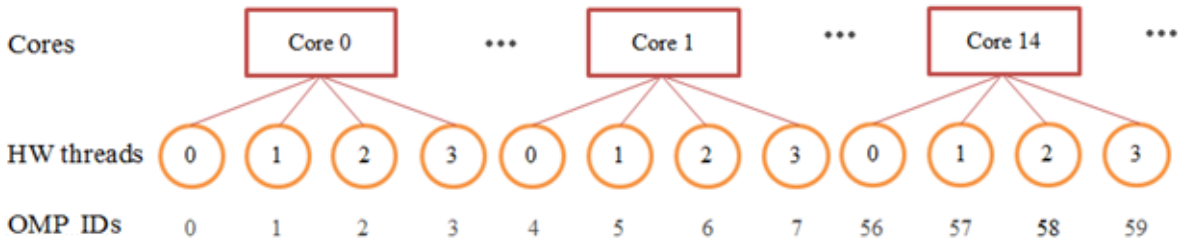
	Intel Xeon				Intel Xeon Phi			
N	**Threads**	**N_{bloque}**	**Tiempo**	**GFlops**	**Threads**	**N_{bloque}**	**Tiempo**	**GFlops**
1152	12	96	0.0260	117.84	48	32	0.1688	18.11
1920	12	80	0.1199	118.02	64	32	0.5659	25.02
2688	12	112	0.3131	124.04	84	32	1.1527	33.70

Tabla 6. Prestaciones obtenidas con uso de bloques.

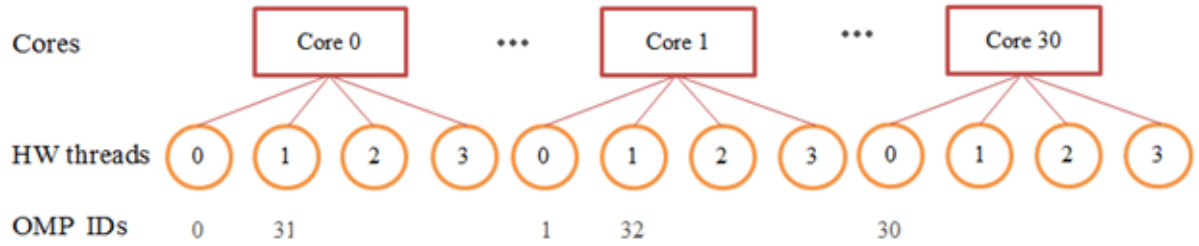
5. Evaluación de Prestaciones

Esquemas de Afinidad

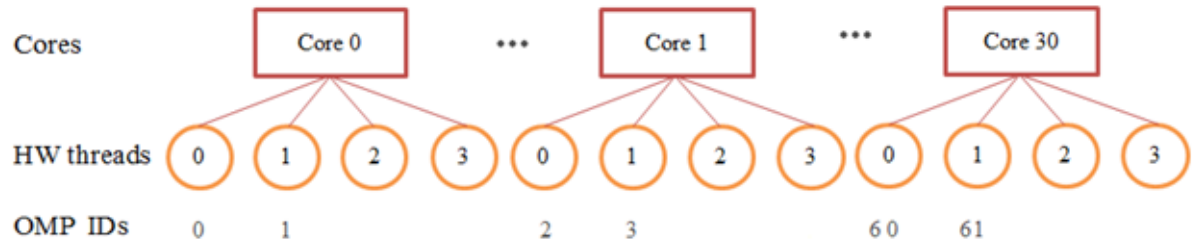
AFFINITY: compact



AFFINITY: scatter



AFFINITY: balanced



5. Evaluación de Prestaciones

- Influencia de la Afinidad en Procesador Intel Xeon

	Afinidad = *compact*				Afinidad = *scatter*			
N	**Threads**	**Block_Size**	**Tiempo**	**GFlops**	**Threads**	**Block_Size**	**Tiempo**	**GFlops**
1152	12	96	0.0262	116.86	12	96	0.0260	**117.84**
1920	12	80	0.1181	**119.88**	12	80	0.1199	118.02
2688	12	112	0.3175	122.34	12	112	0.3131	**124.04**
3456	12	96	0.7052	117.07	12	96	0.6846	**120.59**
4224	11	128	1.3338	**113.01**	12	128	1.3345	112.95

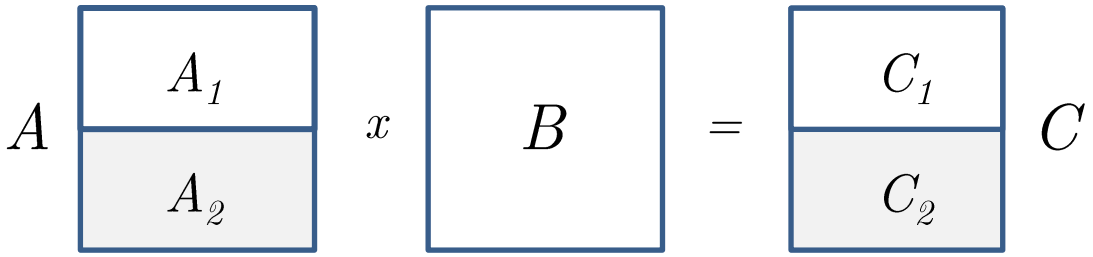
- Influencia de la Afinidad en Coprocesador Intel Xeon Phi

	Afinidad = *compact*			Afinidad = *scatter*			Afinidad = *balanced*		
N	**Threads**	**Tiempo**	**GFlops**	**Threads**	**Tiempo**	**GFlops**	**Threads**	**Tiempo**	**GFlops**
1152	228	0.0384	79.70	168	0.0370	**82.67**	228	0.0429	71.22
1920	224	0.1743	**81.21**	208	0.1754	80.71	224	0.1888	74.96
2688	224	0.5288	73.45	200	0.5253	**73.94**	168	0.5570	69.73
3456	228	1.2103	68.21	172	1.1774	**70.12**	168	1.2312	67.06
4224	224	2.2543	66.86	180	2.2067	**68.31**	164	2.3057	65.37

5. Evaluación de Prestaciones

- Evaluación en Sistema Heterogéneo (*Esquema Algorítmico*)

```
#pragma omp parallel
{
   #pragma omp sections
   {
      #pragma section
      {
         #pragma offload target(mic:micId) in(A...) in(B...) inout(C...)
         {
            doMatMul(...,n,inicio,fin,thrMIC);        //A ejecutar en cada coprocesador
         }
      }
      #pragma section
      {
         doMatMulBloq(...,n,tb,inicio,fin,thrHost);   //A ejecutar en el host utilizando
      }                                               //paralelismo anidado OpenMP
   }
}
```

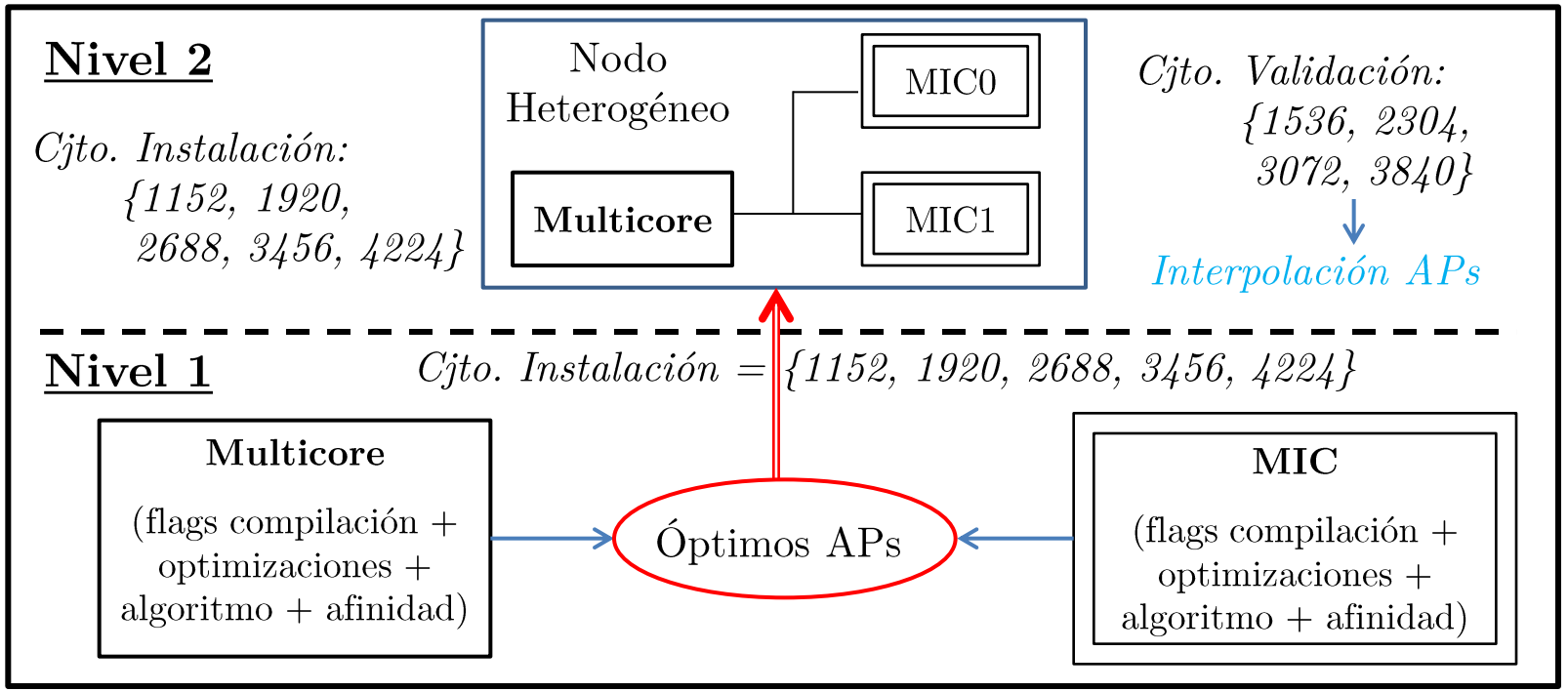


distribución de la carga

5. Evaluación de Prestaciones

- Metodología de Instalación Jerárquica

Objetivo: automatizar selección de los parámetros algorítmicos de la rutina a ejecutar en el sistema.



5. Evaluación de Prestaciones

- Evaluación en Sistema Heterogéneo (*multicore + monoMIC*)

	Host + MonoMIC						
N	**Thr_{host}**	**Thr_{MIC}**	**N_{host}**	**T_{bloque}**	**N_{mic}**	**Tiempo**	**GFlops**
1152	12	0	1152	96	0	0.0250	122.10
1920	12	0	1920	80	0	0.1163	121.76
2688	12	0	2688	112	0	0.3085	125.89
3456	12	0	3456	96	0	0.6740	122.49
4224	12	0	4224	128	0	1.3314	113.21

Tabla 7. Prestaciones de la Rutina en multicore+monoMIC.

	Host + MonoMIC						
N	**Thr_Host**	**Thr_MIC**	**N_{host}**	**T_{bloque}**	**N_{mic}**	**Tiempo**	**GFlops**
1536	12 (12)	0 (0)	1536 (1536)	88 (64)	0 (0)	0.1474 (0.0695)	49.18 (104.33)
2304	12 (12)	0 (0)	2304 (2304)	96 (96)	0 (0)	0.2096 (0.1970)	116.69 (124.17)
3072	12 (12)	0 (0)	3072 (3072)	104 (32)	0 (0)	0.8385 (0.6038)	69.15 (96.03)
3840	12 (12)	0 (0)	3840 (3840)	112 (80)	0 (0)	1.2482 (0.9176)	90.73 (123.42)

Tabla 8. Validación de las Prestaciones Obtenidas.

5. Evaluación de Prestaciones

- Evaluación en Sistema Heterogéneo (*multicore + multiMIC*)

	Host + MultiMIC						
N	**Thr_{host}**	**Thr_{MIC}**	**N_{host}**	**T_{bloque}**	**N_{mic}**	**Tiempo**	**GFlops**
1152	12	100	768	64	192	0.0309	98.96
1920	12	216	1152	96	384	0.1039	136.29
2688	12	196	1536	64	576	0.2944	131.93
3456	12	196	2304	96	576	0.6131	134.66
4224	12	192	2688	112	768	0.9404	160.29

Tabla 9. Prestaciones de la Rutina en multicore+multiMIC.

	Host + MultiMIC						
N	**Thr_Host**	**Thr_MIC**	**N_{host}**	**T_{bloque}**	**N_{mic}**	**Tiempo**	**GFlops**
1536	12 (12)	158 (216)	960 (768)	80 (64)	288 (384)	0.3791 (0.0866)	19.12 (83.66)
2304	12 (12)	206 (216)	1344 (1536)	80 (64)	480 (384)	0.7552 (0.1937)	32.39 (126.32)
3072	12 (12)	196 (192)	1920 (1536)	80 (64)	576 (768)	0.5790 (0.5422)	100.14 (106.94)
3840	12 (12)	194 (192)	2496 (2304)	104 (96)	672 (768)	1.6455 (0.8039)	68.82 (140.88)

Tabla 10. Validación de las Prestaciones Obtenidas.

Índice de Contenidos

6. Conclusiones del TFG

- El uso de flags de compilación, técnicas de optimización y la selección adecuada del valor de los parámetros algorítmicos mejora sustancialmente el rendimiento de la rutina.

- El comportamiento de la rutina difiere en cada componente computacional (multicore y coprocesador) en función de la versión utilizada del algoritmo.

- Los resultados obtenidos pueden variar si se realizan nuevas pruebas, se aplican otras optimizaciones o se ejecuta la rutina en otros sistemas con mayor número de multicore y multiMIC.

- Las técnicas de optimización y metodología utilizadas son aplicables a otras rutinas de álgebra lineal.

Índice de Contenidos

7. Trabajo Futuro

- Mejorar técnicas de auto-optimización utilizadas y combinarlas con técnicas basadas en modelado teórico de rutinas.

- Aplicar metodología propuesta a otras rutinas de álgebra lineal y estudiar la ganancia utilizando la rutina de multiplicación de matrices optimizada.

- Utilizar la metodología para ejecutar eficientemente rutinas de librerías en sistemas computacionales complejos.

- Analizar el comportamiento en sistemas heterogéneos con mayor número de coprocesadores y mayor heterogeneidad (clusters con nodos multicore+multiGPU+multiMIC)

Gracias por su Atención