Procesadores Comerciales

Práctica 6: Medida de Prestaciones

García Esteban, Sergio

25-Mayo-2020

MFLOPS

Calcula los MFLOPS para cada tamaño y nivel de optimización (velocidad de las operaciones suma y producto de coma flotante efectuados por el programa). La cifra cambia según el tamaño de la matriz, ¿por qué?

Arquitectura	Optimización	Dimensión matriz	Tiempo (microsegundos)	MFLOPS
SPARC	-xO1	16	15,833	32.337
		64	285,714	28.672
		256	4533,333	28.912
		1024	99533,328	21.069
		4096	1591000	21.090
	-xO2	16	5,667	90.347
		64	128	64
		256	2033,333	64.461
		1024	59333,332	35.345
		4096	955333,312	35.123
X86	-O0	16	0,664	771,084
		64	10,9	751.559
		256	168,6	777.413
		1024	2780	754.371
		4096	43000	780.335
	-O3 -fno-tree-vectorize	16	0,158	3240,506
		64	2,48	3303.225
		256	35,4	3702.598
		1024	808	2595.485
		4096	14600	2298,248
	-O3	16	0,092	5565,217
		64	1,216	6736.842
		256	23,6	5553.898
		1024	700	2995.931
		4096	14600	2298.248
	-O3 -march=native	16	0,064	8000
		64	0,8	10240
		256	20,8	6301.538
		1024	632	3318.278
		4096	14720	2279.513

En SPARC los MFLOPS disminuyen cuando el tamaño de la matriz aumenta, en x86 sin optimizar no cambia cuando cambia el tamaño de la matriz y en x86 optimizado obtiene mejores resultados en matrices medianas.

¿Por qué la cifra que habéis obtenido al ejecutar gauss es muy inferior a los MFLOPS de pico anunciados por la propaganda? ¿Qué tipo de programas rondaría los MFLOPS de pico?

En nuestras mediciones contamos instrucciones que no son de coma flotante, los MFLOPS de propaganda se alcanzan en un programa conformado por únicamente instrucciones de coma flotante.

MIPS

Para cada nivel de optimización y arquitectura, ¿cuántas instrucciones correspondientes al cuerpo del bucle L/U (A[i][j] = ...) se ejecutan en cada iteración?

Arquitectura	Optimización	Instrucciones cuerpo bucle
SPARC	-xO1	43
	-xO2	11
x86	-O0	32
	-O3 -fno-tree-vectorize	8
	-O3	8
	-O3 -march=native	6

Calcular las velocidades en MIPS suponiendo que los tiempos medidos en el apartado anterior se deben exclusivamente a la ejecución de las instrucciones correspondientes al cuerpo del bucle L/U.

Arquitectura	Optimización	Dimensión matriz	Número iteraciones	MIPS
SPARC	-xO1	16	DIM*DIM	695,256742
		64		616,448616
		256		621,628281
		1024		453,001712
		4096		453,43827
	-xO2	16	DIM*DIM	496,911946
		64		352
		256		354,539075
		1024		194,398926
		4096		193,177997
x86	-O0	16	DIM*DIM	12337,3494
		64		12024,9541
		256		12438,624
		1024		12069,9396
		4096		12485,37
	-O3 -fno-tree-vectorize	16	DIM*DIM	12962,0253
		64		13212,9032
		256		14727,191
		1024		10381,9406
		4096		9192,99507

	-O3	16	(DIM*DIM)/2	11130,4348
		64		13473,6842
		256		11107,7966
		1024		5991,86286
		4096		4596,49753
	-O3 -march=native	16	(DIM*DIM)/4	6000
		64		7680
		256		4726,15385
		1024		2488,70886
		4096		1709,63478

¿Por qué la cifra que habéis obtenido al ejecutar gauss es muy inferior a los MIPS de pico anunciados por la propaganda? ¿Qué tipo de programas conseguiría los MIPS de pico?

Porque en nuestro programa se ejecutan instrucciones costosas como accesos a memoria, los MIPS pico se alcanzan en programas con instrucciones sencillas y sin dependencias.

CPI

Conociendo el tiempo de ciclo del procesador, calculad el CPI de este programa.

Arquitectura	Optimización	Dimensión matriz	CPI
SPARC	-xO1	16	1,67564
		64	1,889857
		256	1,87411
		1024	2,571734
		4096	2,569258
	-xO2	16	2,34448
		64	3,309659
		256	3,285957
		1024	5,992831
		4096	6,030708
X86	-O0	16	0,26748
		64	0,274429
		256	0,265303
		1024	0,273407
		4096	0,264309
	-O3 -fno-tree-vectorize	16	0,25459
		64	0,249756
		256	0,224075
		1024	0,31786
		4096	0,358969
	-O3	16	0,296484
		64	0,244922
		256	0,297089
		1024	0,550747
		4096	0,717938
	-O3 -march=native	16	0,55
		64	0,429688

		256	0,698242
		1024	1,325989
		4096	1,930237

¿Para qué tamaño y nivel de optimización se obtiene el mayor CPI? ¿Y el menor? ¿Cómo se explican las diferencias, si existen? ¿Menor CPI implica siempre menor tiempo de ejecución?

El mayor CPI es 6,03 en la máquina SPARC ejecutando con optimización -xO2 la matriz más grande (4096x4096).

El menor CPI es 0,22 en la máquina x86 ejecutando con optimización -O3 escalar la matriz mediana (256x256).

El valor del CPI depende del coste en ciclos de las instrucciones ejecutadas.

Menor CPI implica que las instrucciones ejecutadas de media son menos costosas, en la ejecución del mismo programa si que implica menor tiempo de ejecución pero en diferente programa no tiene por qué implicarlo.

Ancho de banda

Rellenad una tabla en la que se especifiquen, para las versiones optimizadas de las dos arquitecturas, los bytes de cada tipo leídos/escritos en una iteración del bucle interno.

Máquina	Instrucciones	Datos	Banco de registros
Hendrix	44	24/8	76/52
Lab000	42	16/8	68/36

Rellenad ahora otra tabla en la que, para cada tamaño de matriz y para cada arquitectura, se especifiquen los tiempos de ejecución y los MBps de cada tipo.

Máquina	Dimensión matriz	Tiempo (us)	BW instrucciones	BW datos lectura	BW datos escritura	BW registros lectura	BW registros escritura
Hendrix	16	5,667	1987,647	1084,171	361,39	3433,209	2349,038
	64	128	1408	768	256	2432	1664
	256	2033,333	1418,156	773,539	257,846	2449,542	1676,002
	1024	59333,332	777,595	424,143	141,381	1343,119	918,976
	4096	955333,312	772,711	421,479	140,493	1334,684	913,205
Lab000	16	0,158	68050,632	25924,05	12962,025	110177,215	58329,113
	64	2,48	69367,741	26425,806	13212,903	112309,677	59458,064
	256	35,6	77317,752	29454,382	14727,191	125181,124	66272,359
	1024	808	54505,188	20763,881	10381,94	88246,495	46718,732
	4096	14600	48263,224	18385,99	9192,995	78140,458	41368,477