

Módulo 9 Auto-Vectorização

Copie o ficheiro `/share/acomp/LOOP-P09.zip` para a sua directoria, faça o unzip e gere os executáveis com nível de optimização O2 e 2 opções diferentes: `novect-kerne1` não inclui vectorização, enquanto `vec-kerne1` inclui a opção de auto-vectorização pelo compilador. Para esse efeito, basta escrever `./mymake.sh`. Este comando gerará também os ficheiros `vec-kerne1.s` e `novect-kerne1.s` contendo o *assembly* correspondente aos diferentes níveis de optimização.

Ao longo deste módulo iremos concentrar a nossa atenção nas funções `loop1()` a `loop6()`, `loop_AoS()` e `loop_SoA()` que se encontram em `vec-kerne1.c`.

Exercício 1 - Para cada uma das funções `loop1()` a `loop6()` e `loop_AoS()` examine o respectivo código e indique justificando, na coluna “Obs.” da Tabela 1, se pensa tratar-se, ou não, de código vectorizável pelo compilador. As linhas correspondentes a `loop3-V2()` e `loop_SoA()` serão preenchidas em exercícios posteriores.

Para as funções `loop1()` a `loop6()` e `loop_AoS()` preencha as restantes colunas da tabela para ambas as versões (`vec` e `novect`). Note que pode verificar a vectorização consultando os ficheiros `.s`. Para executar cada versão da função `loop()` use o comando `qsub` conforme indicado abaixo para a versão `loop1()`. Este comando executará ambos os executáveis, isto é, `novect-kerne1` e `vec-kerne1`. O argumento usado para seleccionar a versão pode ser consultado na coluna V da Tabela 1:

```
qsub -F “1” loop.sh
```

Exercício 2 - Analize a função `loop3()`. Porque é que não vectoriza? Consegue sugerir uma alteração ao código desta função de forma que vectorize? Se sim, altere-a e preencha a linha `loop3-V2()` da tabela.

Exercício 3 - Escreva a função `loop_SoA()`, para que realize os mesmos cálculos que `loop_AoS()` mas usando uma estrutura de *arrays*.

Note que:

- o tipo de dados `My_SoA` já está definido em `vec-kerne1.h`:
`typedef struct {float *a, *c} MY_SoA;`
- a variável `SoA` é declarada em `main.h` e os respectivos vectores (`a` e `c`) devidamente inicializados.

Exercício Complementar

Considere a função `loop5()`, que não vectoriza devido a uma dependência RAW. Consegue reorganizar as computações para que esta possa vectorizar parcialmente?

Sugestão: Pode separar o ciclo em dois ciclos, um deles sem dependências! Um dos ciclos vectorizará, o outro não!

func	v	opt	T _{exec} (ms)	#I (M)	CPI	#VEC_SP (M)	Obs.
loop1()	1	no					
		vec					
loop2()	2	no					
		vec					
loop3()	3	no					
		vec					
loop3-v2()	3	no					
		vec					
loop4()	4	no					
		vec					
loop5()	5	no					
		vec					
loop6()	6	no					
		vec					
loop_AoS()	7	no					
		vec					
loop_SoA()	8	no					
		vec					

Tabela 1