

高速計算プログラミングⅡ 第三回

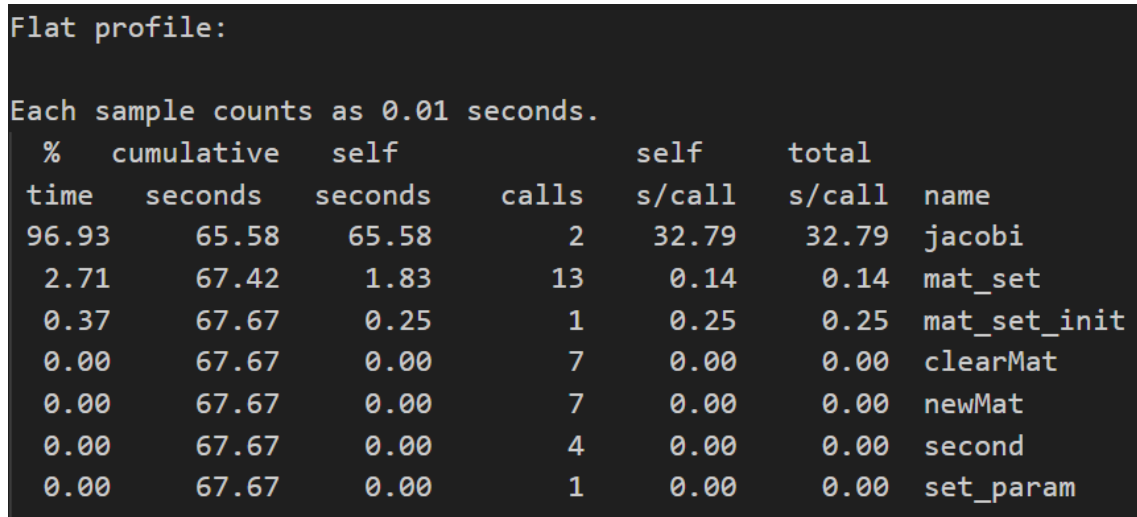
M223337 田川幸汰

Q.

姫野ベンチの C 言語 OpenMP 並列版 (C + OMP, dynamic allocate version) をダウンロードし, gprof, Linux perf, VTune のいずれかを姫野ベンチに適用して結果を示せ. 実行時のグリッドサイズは L とすること.

A.

今回は, gprof を姫野ベンチに適用した. 関数ごとの情報である flat profile 部分の結果について図 1 に示す.



```
Flat profile:
Each sample counts as 0.01 seconds.
%   cumulative   self           calls   self   total    name
time  seconds  seconds             s/call  s/call  s/call
96.93    65.58    65.58              2    32.79    32.79  jacobi
 2.71    67.42     1.83             13     0.14     0.14  mat_set
 0.37    67.67     0.25              1     0.25     0.25  mat_set_init
 0.00    67.67     0.00              7     0.00     0.00  clearMat
 0.00    67.67     0.00              7     0.00     0.00  newMat
 0.00    67.67     0.00              4     0.00     0.00  second
 0.00    67.67     0.00              1     0.00     0.00  set_param
```

%	cumulative	self		self	total	
time	seconds	seconds	calls	s/call	s/call	name
96.93	65.58	65.58	2	32.79	32.79	jacobi
2.71	67.42	1.83	13	0.14	0.14	mat_set
0.37	67.67	0.25	1	0.25	0.25	mat_set_init
0.00	67.67	0.00	7	0.00	0.00	clearMat
0.00	67.67	0.00	7	0.00	0.00	newMat
0.00	67.67	0.00	4	0.00	0.00	second
0.00	67.67	0.00	1	0.00	0.00	set_param

図 1 gprof: flat profile 部分

姫野ベンチの主な処理であるヤコビ反復法の実行時間は 65.58 秒、1 回 call 当たりの実行時間は 32.79 秒となっている。また mat_set 関数は 13 回呼び出しされていて実行時間は 1.83 秒となっている。初期化などその他の関数の呼び出し時間はとても小さくなっている。

各関数とその呼び出し先関数 (callee)、呼び出し元関数 (caller) の 情報である call graph 部分の結果について図 2 に示す。また、図で表して可視化した様子を図 3 に示す。

Call graph (explanation follows)						
granularity: each sample hit covers 2 byte(s) for 0.01% of 67.67 seconds						
index	% time	self	children	called	name	
					<spontaneous>	
[1]	100.0	0.00	67.67		main [1]	
		65.58	0.00	2/2	jacobi [2]	
		1.83	0.00	13/13	mat_set [3]	
		0.25	0.00	1/1	mat_set_init [4]	
		0.00	0.00	7/7	newMat [6]	
		0.00	0.00	7/7	clearMat [5]	
		0.00	0.00	4/4	second [7]	
		0.00	0.00	1/1	set_param [8]	

[2]	96.9	65.58	0.00	2	jacobi [2]	

[3]	2.7	1.83	0.00	13	mat_set [3]	

[4]	0.4	0.25	0.00	1	mat_set_init [4]	

[5]	0.0	0.00	0.00	7	clearMat [5]	

[6]	0.0	0.00	0.00	7	newMat [6]	

[7]	0.0	0.00	0.00	4	second [7]	

[8]	0.0	0.00	0.00	1	set_param [8]	

图 2 gprof: call graph 部分

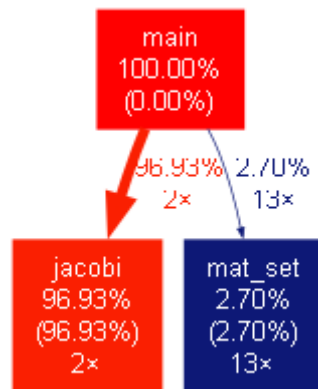


図 3 call graph の可視化

main 関数で、行列のセット、ヤコビ反復法の処理が呼び出されていることがわかる。そして実行時間の約 97% でヤコビ反復法の処理が行われていることがわかる。