High Performance Computing
Homework assignment 2

Luca Venturi

March 23, 2017

All the HW material is posted in the Github repository at the link [1] .

Comments to solved bugs The comments to the solved `omp_bug`$\{2, \dots, 6\}$`.c` are reported at the beginning of the respective `omp_solved`$\{2, \dots, 6\}$`.c`.

Timings for 2D Dirichlet problem

N = 100				
		$\text{it}_{\text{max}} = 1,000$	$\text{it}_{\text{max}} = 10,000$	$\text{it}_{\text{max}} = 100,000$
J	T_{my}	0.0821	0.765	1.7879
	T_{cims}	0.0808	0.5581	1.2537
	Res	0.50797	0.00651	$\text{it}_{\text{max}} = 23391$
J-OMP	T_{my}	0.0801	0.6730	1.6616
	T_{cims}	0.0801	0.4536	1.0146
GS	T_{my}	0.1605	1.4161	1.6328
	T_{cims}	0.1101	0.8214	0.9556
	Res	0.31394	0.00005	$\text{it}_{\text{max}} = 11701$
GS-OMP	T_{my}	0.1193	0.8243	0.9807
	T_{cims}	0.0758	0.5185	0.6231
	Res	0.44382	0.00007	$\text{it}_{\text{max}} = 12051$

N = 1,000				
		$\text{it}_{\text{max}} = 1,000$	$\text{it}_{\text{max}} = 10,000$	$\text{it}_{\text{max}} = 100,000$
J	T_{my}	7.3858	73.676	734.61
	T_{cims}	5.2983	52.492	529
	Res	0.95075	0.84149	0.49692
J-OMP	T_{my}	7.0621	72.813	700.31
	T_{cims}	4.4626	51.195	449.86
GS	T_{my}	14.388	143.95	1439.4
	T_{cims}	8.1377	81.022	814.13
	Res	0.92995	0.77542	0.30304
GS-OMP	T_{my}	7.5757	75.447	759.16
	T_{cims}	4.8405	47.78	549.86
	Res	1.31516	1.09662	0.42856

The above tables show the timings and the residuals obtained using the various method: Jacobi (J), parallel Jacobi (J-OMP), Gauss-Seidel (GS), parallel Red-Black Gauss-Seidel (GS-OMP). We

[1] `https://github.com/luca-venturi/hpc17_cims_nyu/tree/master/hw2`

ran the programs for $N = 100,\ 1,000$ and for different numbers of maximum iterations $\text{it}_{\text{max}} = 1,000,\ 10,000,\ 100,000$. When the residual hits the tolerance $\text{tol} = 10^{-5}$ before it_{max}, the maximum iteration used is reported. Note that the residuals for the parallel Jacobi method are not reported since they are the same of the Jacobi method. The timings reported are those obtained with my machine (T_{my}) and with a CIMS machine (T_{cims}). My machine (PC) has a processor `Intel-Core i3-4010U CPU @ 1.70 GHz` with 2 cores and a 4 GB total RAM. OS is Linux Ubuntu 16.04 (64-bit). The number of threads used in the parallel programs was 4 for both the machines.
By the above results we can deduce the following facts:

- With the respect to J, J-OMP algorithm shows to be faster for big values of N and it_{max}.
- With the respect to GS, GS-OMP algorithm shows to be faster for big values of N and it_{max}. Nevertheless, GS-OMP algorithm converges slower than GS (actually, for small values of it_{max}, the residual increases before starting decreasing). If, instead of it_{max}, we use tol as stopping criteria, the GS-OMP algorithm shows to perform better (i.e. it's faster) than the GS (at least for big enough values of N).

The performances of the various algorithms reflect what was expected: for both J and GS, the parallel version is faster, even if the time difference is significant only for big values of N and it_{max}. Moreover we can see that the parallel Red-Black GS shows a slower convergence of the residual with respect to the standard GS, while the former is faster than the latter for a given number of iterations. This is due to the fact that Red-Black GS corresponds to a GS after a re-ordering of the variables (which translates as a preconditioning of the matrix in the linear system we are considering). Also, parallel GS can implemented using more than 2 colors: in this case, it is possible to show that the fewer the number the slower the convergence is; nevertheless the fewer the number the more parallel the algorithm is. Hence, for parallel GS there is trade-off between parallelism and efficiency.

Note on the implementation of 2D parallel Red-Black Gauss-Seidel algorithm My implementation of the 2D parallel Red-Black Gauss-Seidel algorithm seems quite complicate; the main difficulty was to assign the right labeling to every point of the grid (depending on black or red coloring) in the updating of GS method and in the evaluation of the residual, avoiding the updating of ghost points. An additional complication was due to the fact that this labeling is dependent on N even or odd. In particular this implies that my algorithm is not easily adaptable to the case of number of colors greater than 2.