Prof. Dr. Christoph Pflaum
Florian Schornbaum
Christian Kuschel

Winter Term
2015/2016

Simulation and Scientific Computing
Assignment 4

Exercise 4 (*The Parabolic Heat Equation*)

Tasks

1. Compute an approximate solution $u(x, y, t)$ for $0 \leq t \leq t_{\text{end}}$ to the (parabolic) heat equation [1]:

$$\frac{\partial u}{\partial t} = \kappa \Delta u, \tag{1}$$

with the thermal diffusivity κ. The domain is defined as $[0, 1] \times [0, 1]$ and the boundary conditions for this problem are homogeneous Dirichlet boundary conditions. The initial condition for $t = 0$ is

$$u(x, y, 0) := u_0(x, y) := \sin(\pi x) \sin(\pi y). \tag{2}$$

There are different possibilities for the time discretization which are differently stable. Use a parameter α (passed by the command line) that switches between implicit $(\alpha = 1)$, Crank-Nicolson $(\alpha = 0.5)$, and explicit $(\alpha = 0)$ scheme in the following way:

$$\frac{u^{\text{new}} - u^{\text{old}}}{\tau} = \alpha \kappa \Delta u^{\text{new}} + (1 - \alpha) \kappa \Delta u^{\text{old}} \tag{3}$$

Solve the LSE arising from the implicit and Crank-Nicolson scheme with a MPI parallel Conjugate Gradient method. You can use your solver from the previous assignment.

2. Your program must be callable in the following fashion:

   mpirun -np *N* ./heat *nx ny c eps timesteps* τ κ α *vtk_spacing*,

   where heat is the name of your executable running on *N* processes in parallel.

   nx and *ny* are the number of grid intervals in x- and y-direction, respectively. Time discretization parameters are *timesteps* for the number of time steps and $\tau > 0$ specifying the time step length. κ is the thermal diffusivity and $\alpha \in [0, 1]$ the parameter determining the time discretization scheme.

   The primary stopping criterion for the CG method is the residual threshold specified by *eps*. The maximum number of iterations (in one time step) is limited by *c*.

   vtk_spacing is the number of time steps that between two time steps when writing VTK files. (*vtk_spacing* $= 0$: VTK output is generated after each time step, *vtk_spacing* $= 1$: VTK output is generated every second time step, ...).

   In every time step, your program has to write the number of the CG iteration steps and the residual norm.

3. Your program must generate VTK output files [2] at the end of the first time step and then after every vtk_spacing+1 time step. All VTK files are saved in a subfolder called *vtk* (your program may assume that this subfolder already exists in your working directory when your program is executed). For every time step which is saved, each process generates one

   time_step_$(CURRENT_TIME_STEP)_$(PROCESS_RANK).vtu

   file which contains the simulation data of process $(PROCESS_RANK). In addition, the master process generates the file

   time_step_$(CURRENT_TIME_STEP).pvtu

   which references all vtu files of this time step. At the end of your simulation, the master process generates the file called *solution.pvd* which references all pvtu files.
   VTK files can be visualized with Paraview [3].

4. Instead of writing a Makefile, you have to supply a CMakeLists.txt file which can be used by CMake [4] to generate a Makefile automatically. Your CMakeLists.txt file must meet the following requirements:

   (a) Your executable must be called *heat*.

   (b) If the detected compiler is a gnu cc compatible compiler, the following flags should be added to the compilation process: `-Wall -Wextra -Wshadow -std=c++11`

   (c) Add a CMake option `ARCHITECTURE_OPTIMIZATION` which is ON by default (but could be switched to OFF before generating the Makefile). If this option is ON, the flag `-march=native` must be added to the compiler if the detected compiler is a gnu cc compatible compiler.

   (d) Add another CMake option `ENABLE_WERROR` which is also ON by default. If this option is ON, the flag `-Werror` must be added to the compiler if the detected compiler is a gnu cc compatible compiler.

5. Hand in your solution by Monday, February 1, 2016, 8 a.m. Make sure the following requirements are met:

   (a) Use double precision for your computations.

   (b) Supply a valid CMakeLists.txt file for the compilation.

   (c) The program must be callable as specified above.

   (d) The program must generate VTK output files as specified above.

   (e) Include a specialized `pbs` script.

   (f) The solution must contain well commented source files and instructions how to use your program. When submitting the solution, remove all temporary files from your project directory with the name `ex04_groupXX/` and pack it using the following command:

   ```
   tar -cjf ex04_groupXX.tar.bz2 ex04_groupXX/
   ```

   where `XX` stands for your group number and `ex04_groupXX/` contains your solution. Then upload your solution in StudOn.

Credits

1. You are guaranteed to receive at least one point if your program correctly performs the above tasks and fulfills all of the above requirements. Submissions with compile errors will lead to zero points! The LSS cluster acts as reference environment.

2. Two points are awarded if, additionally, you implement an option to choose a second initial condition that does not lead to a starting configuration that looks like a 'dot' (produced by Equation 2) but more like a donut.

References

[1] `www.f.kth.se/~jjalap/numme/FDheat.pdf`

[2] `www.vtk.org/VTK/img/file-formats.pdf`

[3] `http://www.paraview.org/`

[4] `http://www.cmake.org/cmake/help/cmake_tutorial.html`