

Technical Information

Parallel Processing Winter Term 2020/21

Parallel Processing Exercises under Windows

It is strongly recommended that you use a Linux system for solving the exercises. If you are using Windows, the easiest way is to install a Linux system in a virtual machine. For instructions, please refer, e.g., to [this tutorial](#)¹.

The exercises may also be solved under Windows, with the exception of the 3rd exercise sheet and a part of exercise sheet 4 (which requires the `scalasca` profiler).

Setup

The easiest way to setup a development environment under Windows is to download the Visual Studio Community installer from [the Visual Studio Homepage](#)². At the initial setup of Visual Studio the *Desktop Development with C++* workload can be chosen, which will install all the necessary components needed to solve the exercises, except *MPI*, which will be explained below.

Additionally it may be helpful to have the java JDK installed for easier verification of some exercises. It can be downloaded from [this web page](#)³.

Each exercise comes with a number of C++ *source* and *header* files. To open these files in Visual Studio, first create a new empty project (in the *File* menu or using the *New Project* button in the startup screen) and select *Empty Project* as the template (Make sure it is an empty C++ Project and not another language). Then simply drag all the C++ *source* and *header* files onto the new project in the solution explorer, or right-click on the project in the solution explorer and select *Add* → *Add existing Item* and then select the C++ *source* and *header* files.

Exercise Sheet 1 (Threads)

If Visual Studio Community is installed with the *Desktop Development with C++* workload, no other configuration is needed to solve this exercise. For the second exercise (profiling) the Visual Studio profiling tools may be used. To access them use the *Debug* → *Performance Profiler* menu item.

Exercise Sheet 2 (Jacobi and Gauss/Seidel with OpenMP)

To enable OpenMP support in the Visual Studio project, right-click on the project and select *Properties*. In the *C/C++* → *Language* section, OpenMP support can be set to *Yes*.

Exercise Sheet 3 (Sokoban with OpenMP)

Unfortunately this exercise sheet cannot be solved using the Microsoft C++ compiler, since it only supports OpenMP up to version 2.0. For this exercise, however, at least version 3.0 is required.

¹<https://itsfoss.com/install-linux-in-virtualbox/>

²<https://visualstudio.microsoft.com/de/vs/community/>

³<https://www.oracle.com/java/technologies/javase-downloads.html>

Exercise Sheet 4 (Jacobi and Gauss/Seidel with MPI)

The Microsoft MPI library can be downloaded from [this web page](#)⁴ under “MS-MPI Downloads”. Both the “msm-pisetup.exe” and the “msmpisdsk.msi” are needed. After installing both, the Visual Studio project for these exercises must be configured to use them.

Right-click on the project for the exercise and select “Properties” at the very bottom. For the configuration and platform dropdowns at the top, select “All Configurations” and “All Platforms”. Then go to “C/C++ → General” and edit the “Additional Include Directories” via the dropdown. Click on “New Line” in the top right of the opened dialog and enter “C:\Program Files (x86)\Microsoft SDKs\MPI\Include” (if you have installed the MS-MPI SDK in the default location). After that, under “Linker → General” edit “Additional Library Directories” add the line “C:\Program Files (x86)\Microsoft SDKs\MPI\Lib\\$(PlatformTarget)” and under “Linker → Input” edit “Additional Dependencies” and add the line “msmpi.lib”. MS-MPI programs can now be compiled under Visual Studio.

To run an MS-MPI program, Visual Studio’s debugging facilities cannot be used. The program will start, but it will of course only be executed with a single process. To run the program with multiple processes, open a command prompt in the project’s output directory where the built executable resides and execute “mpiexec <executable>”.

To find the output directory, right-click on the Visual Studio’s Solution Explorer and select “Open Folder in File Explorer”. Then open the folders with the name of the platform and configuration respectively⁵. Then hold down shift and right-click in an empty part of the Folder in the Explorer window and select “Open Powershell window here”.

⁴<https://docs.microsoft.com/en-us/message-passing-interface/microsoft-mpi>

⁵i.e. Open “x64\Release” for the platform “x64” and the “Release” configuration, The selected platform and configuration can be seen in the toolbar of Visual Studio, left of the green run button.