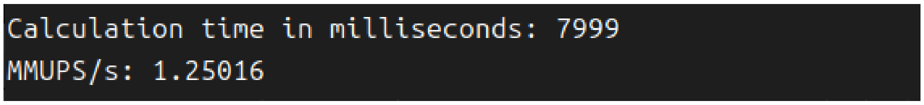
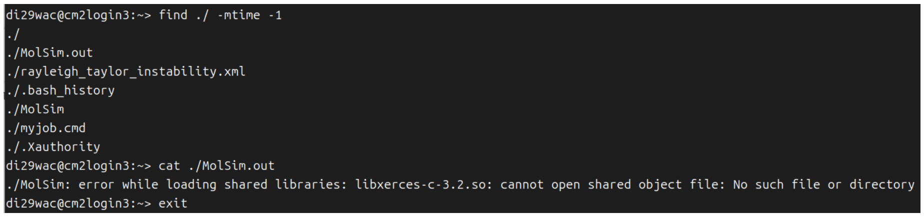
For the contest, aka running the rayleigh big experiment for first 1000 iterations, the result of our group is as follows:



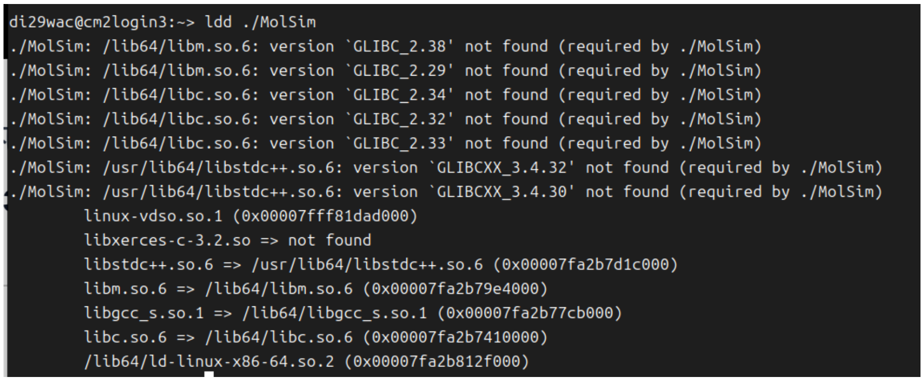
```
Calculation time in milliseconds: 7999
MMUPS/s: 1.25016
```

Now I may introduce the whole process. Besides the complicated login procedures, the biggest problem is the failure of linking the libxerces library and thus failed to read xml files in the cluster. At first, I built the program on local and loaded the executable, input file and file into the cluster. I got the error as follows:



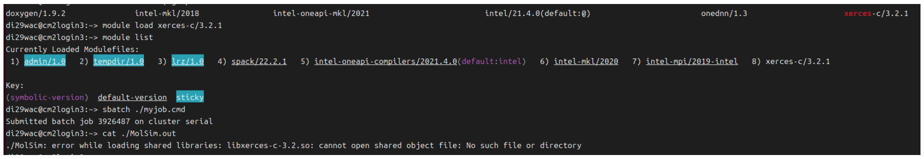
```
di29wac@cm2login3:~> find ./ -mtime -1
./
./MolSim.out
./rayleigh_taylor_instability.xml
./.bash_history
./MolSim
./myjob.cmd
./.Xauthority
di29wac@cm2login3:~> cat ./MolSim.out
./MolSim: error while loading shared libraries: libxerces-c-3.2.so: cannot open shared object file: No such file or directory
di29wac@cm2login3:~> exit
```

After taking a deep look into the dependency of our executable, it turned out that it could not find several related libraries used by the program.



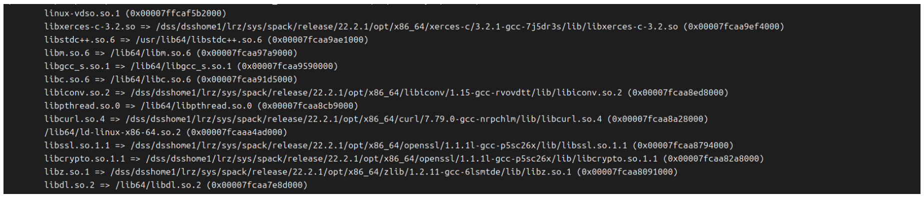
```
di29wac@cm2login3:~> ldd ./MolSim
./MolSim: /lib64/libm.so.6: version `GLIBC_2.38' not found (required by ./MolSim)
./MolSim: /lib64/libm.so.6: version `GLIBC_2.29' not found (required by ./MolSim)
./MolSim: /lib64/libc.so.6: version `GLIBC_2.34' not found (required by ./MolSim)
./MolSim: /lib64/libc.so.6: version `GLIBC_2.32' not found (required by ./MolSim)
./MolSim: /lib64/libc.so.6: version `GLIBC_2.33' not found (required by ./MolSim)
./MolSim: /usr/lib64/libstdc++.so.6: version `GLIBCXX_3.4.32' not found (required by ./MolSim)
./MolSim: /usr/lib64/libstdc++.so.6: version `GLIBCXX_3.4.30' not found (required by ./MolSim)
        linux-vdso.so.1 (0x00007fff81dad000)
        libxerces-c-3.2.so => not found
        libstdc++.so.6 => /usr/lib64/libstdc++.so.6 (0x00007fa2b7d1c000)
        libm.so.6 => /lib64/libm.so.6 (0x00007fa2b79e4000)
        libgcc_s.so.1 => /lib64/libgcc_s.so.1 (0x00007fa2b77cb000)
        libc.so.6 => /lib64/libc.so.6 (0x00007fa2b7410000)
        /lib64/ld-linux-x86-64.so.2 (0x00007fa2b812f000)
```

Finding out that I'm not the root user and don't have the right to import libraries into the cluster, I then tried to specify shared library search paths by adding `export LD_LIBRARY_PATH=/path/to/local/x86_64-linux-gnu:$LD_LIBRARY_PATH` into the `.bashrc` of the cluster, trying to link the library from my local to the program running on cluster. However, it failed to solve the problem. Then I found out that there are extra modules on the cluster that could be added and used, so by commands `module list`, `module avail` and `module load`, I searched and added the module `xerces-c/3.2.1` and checked the linked library again by command `ldd program`.



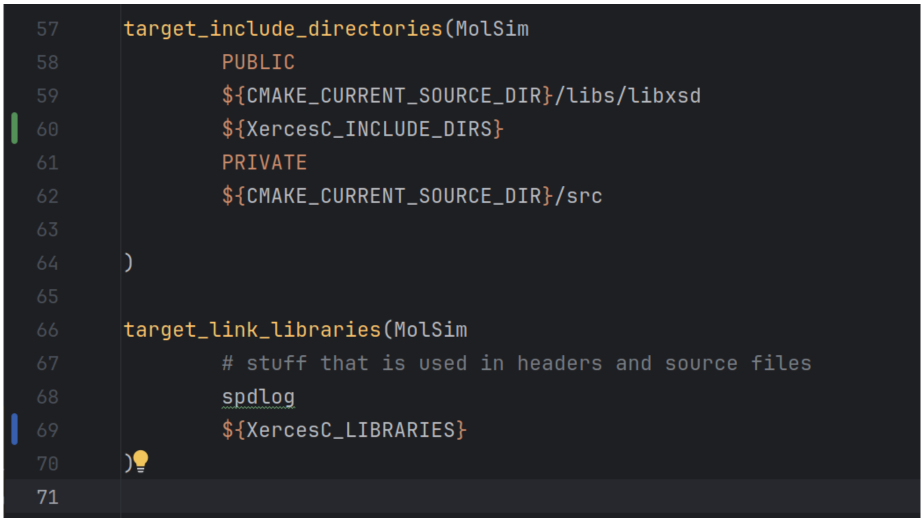
```
doxygen/1.9.2               intel-mkl/2018              intel-oneapi-mkl/2021              intel/21.4.0(default:@)              onednn/1.3              xerces-c/3.2.1
di29wac@cm2login3:~> module load xerces-c/3.2.1
di29wac@cm2login3:~> module list
Currently Loaded Modulefiles:
 1) admin/1.0   2) tempdir/1.0   3) lrz/1.0   4) spack/22.2.1   5) intel-oneapi-compilers/2021.4.0(default:intel)   6) intel-mkl/2020   7) intel-mpi/2019-intel   8) xerces-c/3.2.1

Key:
(symbolic-version)  default-version  sticky
di29wac@cm2login3:~> sbatch ./myjob.cmd
Submitted batch job 3926487 on cluster serial
di29wac@cm2login3:~> cat ./MolSim.out
./MolSim: error while loading shared libraries: libxerces-c-3.2.so: cannot open shared object file: No such file or directory
```

As the image below shows, all the related libraries should have been included, however, it still failed to run the executable.

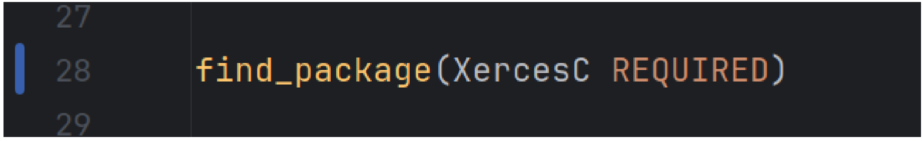


```
        linux-vdso.so.1 (0x00007ffcaf5b2000)
        libxerces-c-3.2.so => /dss/dsshome1/lrz/sys/spack/release/22.2.1/opt/x86_64/xerces-c/3.2.1-gcc-7j5dr3s/lib/libxerces-c-3.2.so (0x00007fcaa9ef4000)
        libstdc++.so.6 => /usr/lib64/libstdc++.so.6 (0x00007fcaa9ae1000)
        libm.so.6 => /lib64/libm.so.6 (0x00007fcaa97a9000)
        libgcc_s.so.1 => /lib64/libgcc_s.so.1 (0x00007fcaa9590000)
        libc.so.6 => /lib64/libc.so.6 (0x00007fcaa91d5000)
        libiconv.so.2 => /dss/dsshome1/lrz/sys/spack/release/22.2.1/opt/x86_64/libiconv/1.15-gcc-rvovdtt/lib/libiconv.so.2 (0x00007fcaa8ed8000)
        libpthread.so.0 => /lib64/libpthread.so.0 (0x00007fcaa8cb9000)
        libcurl.so.4 => /dss/dsshome1/lrz/sys/spack/release/22.2.1/opt/x86_64/curl/7.79.0-gcc-nrpchlm/lib/libcurl.so.4 (0x00007fcaa8a28000)
        /lib64/ld-linux-x86-64.so.2 (0x00007fcaaa4ad000)
        libssl.so.1.1 => /dss/dsshome1/lrz/sys/spack/release/22.2.1/opt/x86_64/openssl/1.1.1l-gcc-p5sc26x/lib/libssl.so.1.1 (0x00007fcaa8794000)
        libcrypto.so.1.1 => /dss/dsshome1/lrz/sys/spack/release/22.2.1/opt/x86_64/openssl/1.1.1l-gcc-p5sc26x/lib/libcrypto.so.1.1 (0x00007fcaa82a8000)
        libz.so.1 => /dss/dsshome1/lrz/sys/spack/release/22.2.1/opt/x86_64/zlib/1.2.11-gcc-6lsmtde/lib/libz.so.1 (0x00007fcaa8091000)
        libdl.so.2 => /lib64/libdl.so.2 (0x00007fcaa7e8d000)
```

After that I realized that I should have built the project on cluster instead of local. I tried it again from the beginning, using `scp /path/to/local/file <USER>@lxlogin<1234 89>.lrz.de: /path/to/cluster/directory` to upload all the program and input files and built the program on the login node with `xerces-c/3.2.1` module added. In the CMakeFile of the program, I also added xerces-c into the linked library.



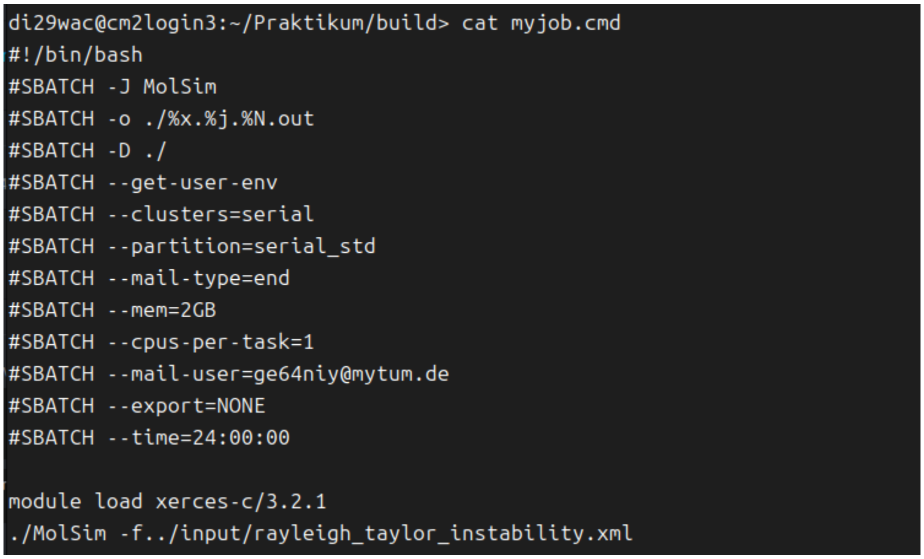
```
target_include_directories(MolSim
        PUBLIC
        ${CMAKE_CURRENT_SOURCE_DIR}/libs/libxsd
        ${XercesC_INCLUDE_DIRS}
        PRIVATE
        ${CMAKE_CURRENT_SOURCE_DIR}/src


)

target_link_libraries(MolSim
        # stuff that is used in headers and source files
        spdlog
        ${XercesC_LIBRARIES}
)

```



```

find_package(XercesC REQUIRED)

```

.

The slurm job in `myjob.cmd` is written as follows (also experienced quite a number of failures and changes):



```
di29wac@cm2login3:~/Praktikum/build> cat myjob.cmd
#!/bin/bash
#SBATCH -J MolSim
#SBATCH -o ./%x.%j.%N.out
#SBATCH -D ./
#SBATCH --get-user-env
#SBATCH --clusters=serial
#SBATCH --partition=serial_std
#SBATCH --mail-type=end
#SBATCH --mem=2GB
#SBATCH --cpus-per-task=1
#SBATCH --mail-user=ge64niy@mytum.de
#SBATCH --export=NONE
#SBATCH --time=24:00:00

module load xerces-c/3.2.1
./MolSim -f../input/rayleigh_taylor_instability.xml
```

Using command `sbatch myjob.cmd`, it finally succeeded in running the simulation on the compute nodes. In the output file `MolSim.3929689.i23r06c03s09.out`, the result of our contest (changing the times of iteration limit to 1000) turned out to be 7999 milliseconds and 1.25016 MMUPS/s.