Cygwin coding under Windows

This tutorial helps you set up a coding environment on Windows with the support for C/C, OpenMP, MPI, as well as compiling and running the TMAC package. If you consider using MinGW, please see this tutorial.

Overview

- **Cygwin** is a unix-like command-line evironment for Windows. It comes with lots of packages for coding and other purposes.
- **OpenMP** is an inter-thread communication specification; it often comes with compilers (e.g., latest GCC). In principle, threads can share memory, but processes cannot. Processes can share data through message passing.
- **MPI** (Message Passing Interface) is a specification for inter-process communication via message passing. MS-MPI is Microsoft's implementation of MPI. It comes with header and library files, as well as some exe's, that you need to compile and execute your codes with the MPI support. Besides MS-MPI, Windows supports other MPI implementations.
- **BLAS** is a collection of routines you can call from your codes to perform basic vector and matrix operations. They come with header and library files, and they are typically more efficiently than your own implementations. Besides the NetLib implementation, BLAS has other implementations from ATLAS, INTEL MKL, Open BLAS, etc.
- **TMAC** is a C++11 framework that lets you quickly develop your own codes for solving a variety of optimization problems in a parallel fashion. In particular, you can add your operators to TMAC and run your algorithms based on operator splitting and coordinate update methods. TMAC makes it easy to test both single-threaded and synchronous, as well as asynchronous, parallel algorithms.

Install Cygwin

- start with no cygwin on your system (otherwise, please uninstall them)
- run the installer (this tutorial used setup-x86_64.exe and installed “from Internet” to `c:\cygwin64`)
- **select packages**: (not all are necessary, but they are generally useful)
  - under **Devel**: autoconf, autoconf2.1, autogen, automake, all automake1.*, bison, flex, gcc-core/fortran/g, gdb, git, gperf, help2man, libtool, make, patch, patchutils, pkg-config, swig, texinfo, texinfo-tex, mingw64-x86_64-gcc-core/g++
  - under **Libs**: libgcc1, lapack, liblapack-devel/doc/0, libopenmpi/-devel/12/cxx1
  - under **Perl**: perl
  - under **Utils**: diffutils, dos2unix
  - under **Web**: wget

Verify installation

- open Cygwin64 Terminal

```
> gcc -v    # test gcc
```

- check gcc version >= 4.9 and you see “Thread model: posix”
- (optional) if you (want to) use “apt-get” to manage packages, then check out apt-cyg

Run the codes for Brent's lectures

- this part requirements: `gcc` (ver>=4.9) and `MPI` (the above MS-MPI is Okay)
- open Cygwin64 Terminal
- download Brent's code package from GitHub

```
> git clone https://github.com/BrentEdmunds/ThreadsandMPI.git
```

OpenMP

- enter folder with an OpenMP cpp code, and then type

```
> make
```

- now, run the created exe-file

MPI

- instead of calling `make`, you must compile and run an MPI code by yourself as follows

```
> cd ThreadsandMPI/MPI/HellowWorld/
> rm helloworld.exe    # if it exists, delete it
> mpic++ helloworld.cpp -o helloworld.exe
> mpirun -np 4 helloworld.exe    # run 4 processes
```

- if Windows Firewall asks for your permission, check both “private” and “public” boxes and click on “Allow”
- if you mpirun gives you an error “tcp_peer_send_blocking:”, then go to Windows, open Networks Connections, and disable any unused/disconnected network interfaces there.

Run TMAC

- download the code from GitHub (if you have the access)

```
> git clone https://github.com/uclaopt/tmac.git
```

- open `Makefile` in the project root in an editor

```
> make
> ./bin/tmac_prs_demo -problem_size 1500 -nthread 1    # run Peaceman-Rachford Splitting algorithm with 1 thread
> ./bin/tmac_prs_demo -problem_size 1500 -nthread 2    # run 2 threads
> ./bin/tmac_prs_demo -problem_size 1500 -nthread 4    # run 4 threads
```

updated 2016-12-29, by jemdoc