A brief intro into AMBER PACKAGE, including visualization tools.

In this assignment you will run a few semi-realistic simulations that will help you master the very basics of the AMBER suite of programs, which you will use for your main project. The goal is to become familar with the input/output and key parameters. Please provide TYPED solutions, no more than 3 pages long, including pictures. One report per group, please. Indicate who did what and % effort for each group member. Use of Windows OS for file manipulation is discouraged, as it is poorly suited for scientific computing.

• Warm-up. Skim through first few pages of "Simple Simulation of Alanine Dipeptide" that can be found here:
`http://ambermd.org/tutorials/Introductory.php`. The set-up of these simulations is hardcore, but, fortunately, you will not be going through all of that rigmarole, as all the input files are available for you in that tutorial (and the project). Create "Tutoral" directory as suggested in your group's directory, this is where this work will be done.

• Carefully do all of the steps in section "Prepare Amber MD pmemd and sander input files". This is where you will actually do some type of minimization followed by heating of the molecule, followed by "making it come alive" – a molecular dynamics at constant temperature. These are the very basic steps that will be useful for you in your project. Use the "supercomputer on the desk" machine assigned to this class; I have explained in class how to login, contact your GTA with questions.

Each group creates one working directory named accordingly. Use the parm7 (prmtop) and rst7 (inpcrd) from the tutorial website – links right above "Prepare Amber MD pmemd and sander input files". DO NOT attempt to create the input files for these simulations, just take them from the tutorial. Note that this tutorial has been available for years, and successfully completed by thousands of people, which means it is guaranteed to work if you follow the steps carefully. All the necessary code has been pre-installed for you. Some directories may have been moved in the latest AMBER, *e.g.* you may want to look into $AMBERHOME-/bin/.

Then, do "Run Amber MD pmemd" part, followed by "Visualize the results" part. Under "Run production MD" 19., the command

```
pmemd -O -i 03_Prod.in -o 03_Prod.out -p parm7 -c 02_Heat.ncrst \
  -r 03_Prod.ncrst -x 03_Prod.nc -inf 03_Prod.info &
```

will run on the cpu. To run in "supercomputer" mode (and significantly speed up the MD simulation) see the README file in the kuprin home directory. In the unix shell, export the GPU corresponding to your group number,
e.g. CUDA_VISIBLE_DEVICES=2,
and use `pmemd.cuda` instead of `pmemd` in the code snippet above. Important: do not change any of the input files above.

For visualization, you must install VMD or/and PyMol on your own machine/laptop, transfer the relevant files from the supercomputer, and visualize them locally. VMD is free and open source. PyMol is available free through VT Network Software. It is reommended that you install both, and learn how they work. Some features work better in one than the other. Save a couple of snapshots from the trajectory you have generated, include them in your report.

VMD: `http://www.ks.uiuc.edu/Research/vmd/`

(Windows users may run into probelms with netcdf trajectory format on VMD. A workaround is decribed in the README in the top directory.)

Notes on using PyMol with AMBER: (1)

`https://wikis.ch.cam.ac.uk/ro-walesdocs/wiki/index.php/Loading_AMBER_prmtop_and_inpcrd_f`

(2) `https://pymolwiki.org/index.php/Load_traj`

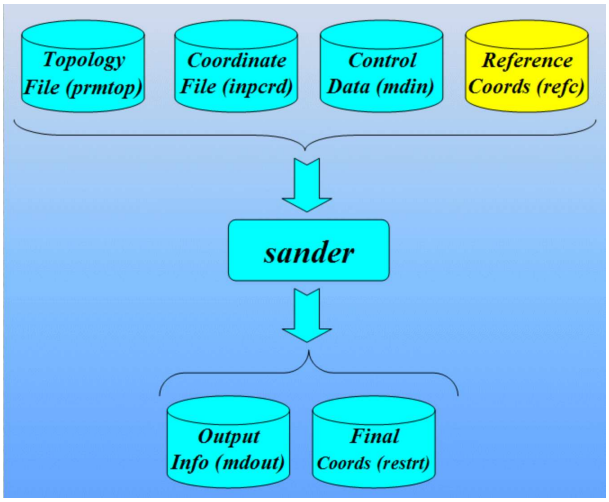

Figure 1: Data flow in Amber. prmtop - a file containing a description of the molecular topology and the necessary energy function (force-field) parameters. inpcrd (or a restrt from a previous run) - a file containing a description of the atom coordinates and optionally velocities. mdin - the sander input file consisting of a series of namelists and control variables that determine the options (e.g. how often to output snapshots) and type of simulation to be run, e.g. whether this is a minimization run or a run at constant temperature. mdout – run info, including energy of every snapshot.

For the coming exercises, which is separate from the above and relies on a separate set of input files, copy the `EXAMPLES` directory to your own *personal* directory, within your particular group's folder. Name it whatever you like. Further mentions of the `EXAMPLES` directory will refer to your *personal* copy.

• Explore the power of your supercomputer. Familiarize yourself with the contents of the EXAMPLES directory. See README. First, run MD on a single CPU and notice the time it takes to perform 2000 steps. Then run the same one in supercomputer mode. Look at the bottom of `mdout` file for the total time, or use the unix "time" utility. What is the speed-up? That is the ratio of the two times? If you were to simulate thioredoxin for 1 ns on GPU, how much computer time would that take?

• Explore how the GPU speed-up depends on the structure size. Inside the "folding miniprotein" folder – that's where all the key files for your project are – you will find `minip.prmtop` and `minip.inpcrd`; these are your input files to use in MD - you have just learned how to set initial conditions for your simulations. (rename the files to minip.top and minip.crd). Use those instead of the defaults in the `EXAMPLES` directory (make sure you do this run in your own directory, not in `EXAMPLES`!). It is possible that the simulation breaks down, because `minip.inpcrd` is not minimized (gradients are too large for the integrator to handle). In this case, use `min.rst` (from the same "folding miniprotein" folder)

instead of `minip.inpcrd`. Report the run time and speed-up numbers in a table as a function of the number of atoms in each of the proteins you have tested. You can look up the number of atoms in "mdout" output file. You can find the number of atoms in one of the AMBER output files, just look carefully. Or you can just count the atoms in the PDB file that you can make from the input amber files using `cpptraj` or `ambpdb -p minip.top < minip.crd > minip.pdb`.

• Now that you have an idea of what is possible, create a VMD or PyMol movie of the example protein (2trx) dynamics. Then, estimate number of steps you can afford to run in about 1 hour on GPU (supercomputer mode). Then, figure out how to save, say, 100 equidistant snapshots (frames) from start to finish. Run the simulation and use VMD (or PyMol) to make a movie of the protein trajectory; include a few snaphots in your report.

• Look inside "mdout" file, it gives you energy of your protein as a function of simulation time, that is at each step of the trajectory (for each frame). Use unix tools `grep` and `awk` to output the potential energy as a function of time (frame number). For example, in the following line: `Etot = -325.3714 EKtot = 1048.6677 EPtot = -1374.0391` you need the value of `EPtot`. Present a plot for Eptot(t).