ColabFold 1.5.5 (local db version)

Webpage

https://github.com/sokrypton/ColabFold

Brief installation procedure

Ref: https://github.com/sokrypton/ColabFold/wiki/Running-ColabFold-in-Docker
We have installed databases and conda environment for ColabFold 1.5.5. Docker/apptainer(singularity) was not used in this procedure; don't run local MSA server, use only local DB for MSA.

Python environment

```
$ sh Miniforge3-Linux-x86_64.sh
$ cd /apl/colabfold/1.5.5
$ ./bin/conda shell.bash hook > conda_init.sh
$ ./bin/conda shell.tcsh hook > conda_init.csh
$ . /apl/colabfold/1.5.5/conda_init.sh
$ conda install cudatoolkit=11.8.0
$ CONDA_OVERRIDE_CUDA=11.8.0 conda install -c conda-forge -c bioconda colabfold=1.5.5=* jaxlib=*=*cuda* libabseil libgrpc python mmseqs2
vmtouch
```

Get databases

- We followed the procedure described in setup_databases.sh in ColabFold 1.5.5 manually
  - We use mmseqs version specfied in MsaServer/setup-and-start-local.sh in this step.
  - Index generation (createindex) was skipped.
  - Databases are available in /apl/colabfold/1.5.5/MsaServer/databases.
  - AlphaFold2 parameters are downloaded in /apl/colabfold/1.5.5/MsaServer/params.
  - we don't use msa-server.

Scatter large files among OSTs (this step is specific to lustre file system).

```
$ lfs migrate -c 2 colabfold_envdb_202108_db_h.index # 17GB
$ lfs migrate -c 2 colabfold_envdb_202108_db_seq.index # 18GB
$ lfs migrate -c 2 pdb100_foldseek_230517.tar.gz # 18GB
$ lfs migrate -c 3 colabfold_envdb_202108_db_h # 24GB
$ lfs migrate -c 3 colabfold_envdb_202108_db # 25GB
$ lfs migrate -c 3 colabfold_envdb_202108_db_aln # 27GB
$ lfs migrate -c 3 uniref30_2302_aln.tsv # 29GB
$ lfs migrate -c 3 colabfold_envdb_202108_h.tsv # 30GB
$ lfs migrate -c 4 colabfold_envdb_202108.tsv # 38GB
$ lfs migrate -c 5 uniref30_2302_db_h # 41GB
$ lfs migrate -c 5 uniref30_2302_h.tsv # 44GB
$ lfs migrate -c 5 colabfold_envdb_202108_aln.tsv # 52GB
$ lfs migrate -c 6 pdb100_a3m.ffdata # 60GB
$ lfs migrate -c 8 uniref30_2302_db_seq # 78GB
$ lfs migrate -c 10 colabfold_envdb_202108_db_seq # 87GB
$ lfs migrate -c 10 uniref30_2302.tar.gz # 96GB
$ lfs migrate -c 11 colabfold_envdb_202108.tar.gz # 110GB
$ lfs migrate -c 12 uniref30_2302_seq.tsv # 128GB
$ lfs migrate -c 12 colabfold_envdb_202108_seq.tsv # 128GB
```

Sample job script for PBS

```
#!/bin/sh
#PBS -l select=1:ncpus=128:mpiprocs=1:ompthreads=128
#PBS -l walltime=24:00:00

if [ ! -z "${PBS_O_WORKDIR}" ]; then
 cd "${PBS_O_WORKDIR}"
fi

# input data and intermediate/work dirs
```

```
NUM_RELAX=1               # number of structures to be relaxed after inference
INPUTFASTA=./monomer.fasta # input sequence
MSADIR=./msas             # intermediate msa directory
OUTPUTDIR=./output        # output of colabfold_batch
MMSEQS_NUM_THREADS=$OMP_NUM_THREADS # number of threads; exporting this may work

# common params
COLABFOLDROOT=/apl/colabfold/1.5.5
. ${COLABFOLDROOT}/conda_init.sh

# search options
MMSEQS=${COLABFOLDROOT}/bin/mmseqs
DBDIR=${COLABFOLDROOT}/MsaServer/databases
colabfold_search --mmseqs ${MMSEQS} \
                 --threads ${MMSEQS_NUM_THREADS} \
  ${INPUTFASTA} ${DBDIR} ${MSADIR}

# run prediction
AF2WEIGHTS=${COLABFOLDROOT}/MsaServer # af2 weight
colabfold_batch \
  --num-relax ${NUM_RELAX} \
  --data ${AF2WEIGHTS} \
  ${MSADIR} ${OUTPUTDIR}
```

Notes

- When msa-server is launched on computation node, the prediction by colabfold_batch does not finish correctly. (There are some errors about templates.)
    - We decided not to use msa-servers. (It may be difficult to get some performance improvement by msa-server. It is difficult to load the data onto memory beforehand on RCCS system.)
- For a prediction of small peptides, you should use colabfold_batch and official msa-server. This would be very much easy and fast. (As described in the official site.)
- When mmseqs2 specified in MsaServer/setup-and-start-local.sh, prediction for complex didn't finish correctly.)There is an error message about pairing option.) Mmseqs2 available from conda works fine.
    - For complex, you need to concatenate amino acid sequences with ":" (e.g. AAAAAA:GGGGG). Structural relaxation after the prediction also works fine in this way.
- If we prepare index of DB (createindex), performance of prediction decreases significantly. (As described in the official site.)