AlphaFold 2.3.2

(brief notes about installation)

Python environment

We here assume miniforge is installed in /apl/alphafold/miniforge3 and code of AlphaFold 2.3.2 is in /apl/alphafold/2.3.2.

```
(base) [user@ccfep4 2.3.2]$ conda install -y -c conda-forge openmm=7.5.1 cudatoolkit==11.2.2 cudnn pdbfixer pip python=3.8
(base) [user@ccfep4 2.3.2]$ conda install -y -c bioconda hmmer==3.3.2 hhsuite==3.3.0 kalign2==2.04
(base) [user@ccfep4 2.3.2]$ CONDA_OVERRIDE_CUDA=11.2 conda install jax==0.3.25 jaxlib=0.3.25=*cuda*
(base) [user@ccfep4 2.3.2]$ pip install absl-py==1.0.0 biopython==1.79 chex==0.0.7 dm-haiku==0.0.9 dm-tree==0.1.6 immutabledict==2.0.0 ml-collections==0.1.0 numpy==1.21.6 pandas==1.3.4 scipy==1.7.0 tensorflow-cpu==2.11.0
(base) [user@ccfep4 2.3.2]$ cd /apl/alphafold/miniforge3/lib/python3.8/site-packages/
(base) [user@ccfep4 site-packages]$ patch -p0 < ../../../../2.3.2/docker/openmm.patch
```

- When we employed CUDA 11.1.1 (according to official one), target version of jax couldn't be installed. CUDA 11.2.2 was tentatively employed.

DB

The latest versions of pdb_mmcif and pdb_seqres were newly installed. For other DBs, those for 2.3.1 installed on Jan 30, 2023 were employed.

AlphaFold

- alphafold/common/stereo_chemical_props.txt was placed as usual.
- patch for alphafold/data/tools/hhblits.py applied (number of threads for hhblits can be modified via env variable)

```
--- hhblits.py.org      2024-02-06 16:27:37.000000000 +0900
+++ hhblits.py  2024-02-07 12:19:54.000000000 +0900
@@ -94,6 +94,14 @@
    self.p = p
    self.z = z

+    n_cpu_env = os.getenv("HHBLITS_NTHREADS")
+    if n_cpu_env:
+        try:
+            n_cpu_env = int(n_cpu_env)
+            self.n_cpu = n_cpu_env
+        except:
+            pass
+
  def query(self, input_fasta_path: str) -> List[Mapping[str, Any]]:
    """Queries the database using HHblits."""
    with utils.tmpdir_manager() as query_tmp_dir:
```

- patch for alphafold/data/tools/jackhmmer.py applied (number of threads for jackhmmer can be modified via env variable)

```
--- jackhmmer.py.org    2024-02-06 16:33:47.000000000 +0900
+++ jackhmmer.py        2024-02-07 12:20:23.000000000 +0900
@@ -87,6 +87,14 @@
    self.get_tblout = get_tblout
    self.streaming_callback = streaming_callback

+    n_cpu_env = os.getenv("JACKHMMER_NTHREADS")
+    if n_cpu_env:
+        try:
+            n_cpu_env = int(n_cpu_env)
+            self.n_cpu = n_cpu_env
+        except:
+            pass
+
  def _query_chunk(self,
```

```
    input_fasta_path: str,
    database_path: str,
```

wrapper script

```
#!/bin/bash
# Description: AlphaFold non-docker version
# Author: Sanjay Kumar Srikakulam
#
#
# RCCS notes:
# This script was customized for RCCS by M. Kamiya (IMS).
# original: https://github.com/kalininalab/alphafold_non_docker

# This script is for AlphaFold 2.3.2!
# Former AlphaFold versions may not be compatible with this script!

# RCCS default value
af2root="/apl/alphafold/2.3.2"
data_dir="/apl/alphafold/databases/20240206"

max_template_date="2024-02-06"
benchmark=false
db_preset="full_dbs"
model_preset="monomer"
use_gpu=false
MYOPTS="" # variable for misc options

usage() {
    echo ""
    echo "Usage: $0 <OPTIONS>"
    echo "Required Parameters:"
    echo "-o <output_dir>   Path to a directory that will store the results."
    echo "-f <fasta_path>   Path to a FASTA file containing one sequence"
    echo ""
    echo "Optional Parameters:"
    echo "-a <alphafolddir> Path to alphafold code"
    echo "-d <data_dir>     Path to directory of supporting data"
    echo "-t <max_template_date> Maximum template release date to consider (ISO-8601 format - i.e. YYYY-MM-DD). Important if folding historical test sets (default: 2021-11-05)"
    echo "-Q                show also pTM score etc. (alias of -m monomer_ptm)"
    echo "-b <benchmark>    Run multiple JAX model evaluations to obtain a timing that excludes the compilation time, which should be more indicative of the time required for inferencing many proteins (default: 'False')"
     echo "-g                Enable NVIDIA runtime to run with GPUs"
    echo "-a <gpu_devices>  Comma separated list of devices to pass to 'CUDA_VISIBLE_DEVICES' (default: '')"
    echo "-r <relax_tgt>    Choose relax target from all ('all'), most confidential mode ('best'), or skip relaxation ('none')"
    echo "-R                Skip running MSA tools and use precomputed one. NOTE: this will not check if sequence/db/conf have changed."
    echo "-s <seeds per model>  Number of seeds per model for multimer system. (Number of models (usually 5)) * (number of seeds; this param) predictions will be performed. (default: 5)"
    echo "-p <db_preset>    Choose db preset - no ensembling (full_dbs), reduced version of dbs (reduced_dbs) (default: 'full_dbs')"
    echo "-m <model_preset> Choose model preset - monomer model (monomer), monomer with extra ensembling (monomer_casp14), monomer model with pTM head (monomer_ptm), or multimer model (multimer) (default: 'monomer')"
    echo ""
    exit 1
}

while getopts ":a:d:o:f:t:a:p:s:m:r:bgQR" i; do
    case "${i}" in
    a)
        echo "INFO: set AF2 root to $OPTARG"
        af2root=$OPTARG
    ;;
    d)
        echo "INFO: set database root to $OPTARG"
        data_dir=$OPTARG
    ;;
```

```
    o)
        output_dir=$OPTARG
    ;;
    f)
        fasta_path=$OPTARG
    ;;
    t)
        max_template_date=$OPTARG
    ;;
     b)
        benchmark=true
    ;;
    g)
        use_gpu=true
    ;;
    Q)
        echo "INFO: set model_preset=monomer_ptm"
        model_preset="monomer_ptm"
    ;;
    a)
        gpu_devices=$OPTARG
    ;;
    p)
        db_preset=$OPTARG
    ;;
    m)
        model_preset=$OPTARG
    ;;
    r)
        MYOPTS="$MYOPTS --models_to_relax=$OPTARG"
     ;;
    s)
        MYOPTS="$MYOPTS --num_multimer_predictions_per_model=$OPTARG"
    ;;
    R)
        MYOPTS="$MYOPTS --use_precomputed_msas=True"
    ;;
    esac
done

# Parse input and set defaults
if [[ "$data_dir" == "" || "$output_dir" == "" || "$fasta_path" == "" ]] ; then
  usage
fi

if [[ "$db_preset" != "full_dbs" && "$db_preset" != "reduced_dbs" ]] ; then
  echo "Unknown db_preset! Using default ('full_dbs')"
  db_preset="full_dbs"
fi

if [[ "$model_preset" != "monomer" && "$model_preset" != "monomer_casp14" && "$model_preset" != "monomer_ptm" && "$model_preset" !=
"multimer" ]]; then
  echo "Unknown model_preset! Using default ('monomer')"
  model_preset="monomer"
fi

alphafold_script="$af2root/run_alphafold.py"
if [ ! -f "$alphafold_script" ]; then
  echo "Alphafold python script $alphafold_script does not exist."
  exit 1
fi

if "$use_gpu" ; then
 MYOPTS="$MYOPTS --use_gpu_relax=True"
else
 MYOPTS="$MYOPTS --use_gpu_relax=False"
```

```
fi

if [[ "$gpu_devices" ]] ; then
   export CUDA_VISIBLE_DEVICES=$gpu_devices
fi

export TF_FORCE_UNIFIED_MEMORY='1'
export XLA_PYTHON_CLIENT_MEM_FRACTION='4.0'

# Binary path (change me if required)
hhblits_binary_path=$(which hhblits)
hhsearch_binary_path=$(which hhsearch)
jackhmmer_binary_path=$(which jackhmmer)
kalign_binary_path=$(which kalign)

MYOPTS="$MYOPTS --hhblits_binary_path=$hhblits_binary_path"
MYOPTS="$MYOPTS --hhsearch_binary_path=$hhsearch_binary_path"
MYOPTS="$MYOPTS --jackhmmer_binary_path=$jackhmmer_binary_path"
MYOPTS="$MYOPTS --kalign_binary_path=$kalign_binary_path"

# uniref30 path
uniref_new=$(find $data_dir -maxdepth 1 -name 'UniRef*')
if [ ! -z "$uniref_new" ]; then
 uniref_name=$(basename $uniref_new)
 uniref30_database_path="$data_dir/$uniref_name/$uniref_name"
elif [ -d "$data_dir/uniref30" ]; then
 uniref30_database_path="$data_dir/uniref30/UniRef30_2021_03"
fi

# bfd path
if [[ "$db_preset" == "reduced_dbs" ]] ; then
 small_bfd_database_path="$data_dir/small_bfd/bfd-first_non_consensus_sequences.fasta"
 MYOPTS="$MYOPTS --small_bfd_database_path=$small_bfd_database_path"
 # uniref30 not necessary
else
 bfd_database_path="$data_dir/bfd/bfd_metaclust_clu_complete_id30_c90_final_seq.sorted_opt"
 MYOPTS="$MYOPTS --bfd_database_path=$bfd_database_path"
 # uniref30 required
 MYOPTS="$MYOPTS --uniref30_database_path=$uniref30_database_path"
fi

# Path and user config (change me if required)
if [ -f $data_dir/mgnify/mgy_clusters_2022_05.fa ]; then
 mgnify_database_path="$data_dir/mgnify/mgy_clusters_2022_05.fa"
else
 mgnify_database_path="$data_dir/mgnify/mgy_clusters.fa"
fi
template_mmcif_dir="$data_dir/pdb_mmcif/mmcif_files"
obsolete_pdbs_path="$data_dir/pdb_mmcif/obsolete.dat"
uniref90_database_path="$data_dir/uniref90/uniref90.fasta"

MYOPTS="$MYOPTS --mgnify_database_path=$mgnify_database_path"
MYOPTS="$MYOPTS --template_mmcif_dir=$template_mmcif_dir"
MYOPTS="$MYOPTS --obsolete_pdbs_path=$obsolete_pdbs_path"
MYOPTS="$MYOPTS --uniref90_database_path=$uniref90_database_path"

# for multimer (pdb70 must not be specified this case)
if [[ "$model_preset" == "multimer" ]]; then
 echo "INFO: appending database paths for multimer model..."
 uniprot_database_path="$data_dir/uniprot/uniprot.fasta"
 MYOPTS="$MYOPTS --uniprot_database_path=$uniprot_database_path"
 pdb_seqres_database_path="$data_dir/pdb_seqres/pdb_seqres.txt"
 MYOPTS="$MYOPTS --pdb_seqres_database_path=$pdb_seqres_database_path"
else
 pdb70_database_path="$data_dir/pdb70/pdb70"
 MYOPTS="$MYOPTS --pdb70_database_path=$pdb70_database_path"
```

```
fi

#echo $MYOPTS

# Run AlphaFold with required parameters
$(python $alphafold_script --data_dir=$data_dir --output_dir=$output_dir --fasta_paths=$fasta_path --max_template_date=$max_template_date --db_preset=$db_preset --model_preset=$model_preset --benchmark=$benchmark --logtostderr $MYOPTS)
```

Notes

- The CUDA version is different from official one due to the dependency problem.
- Number of threads for hhblits and jackhmmer can be modified by HHBLITS_NTHREADS and JACKHMMER_NTHREADS environment variables, respectively.
    - In some cases, better performance can be achieved by using slightly higher number of threads.
        - (This performance improvement might rely on the performance on lustre filesystem.)
    - Decreasing number of threads is not recommended. Using too many threads is not a wise idea.