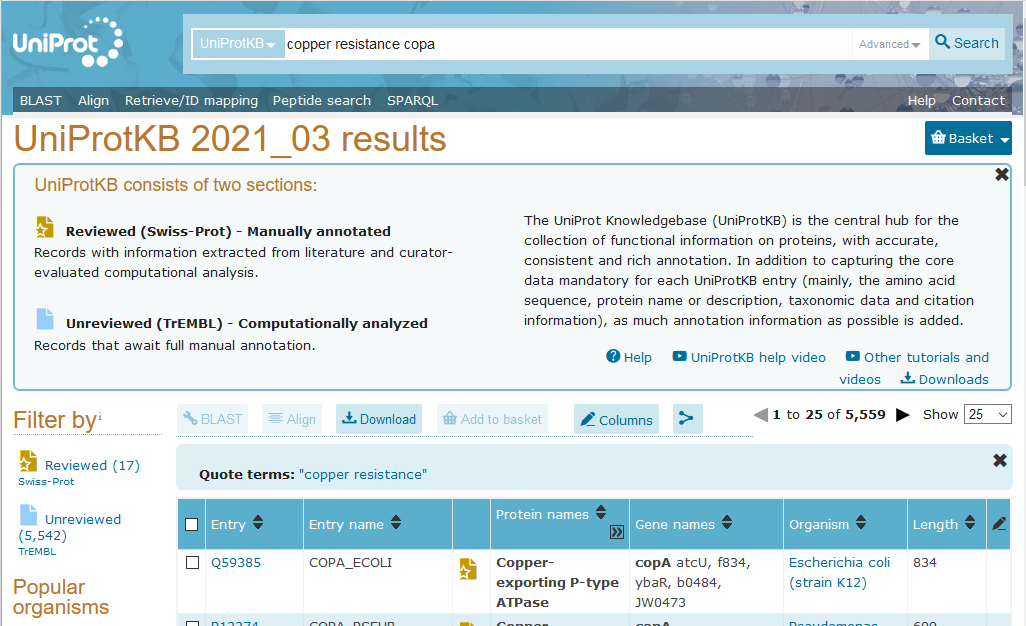
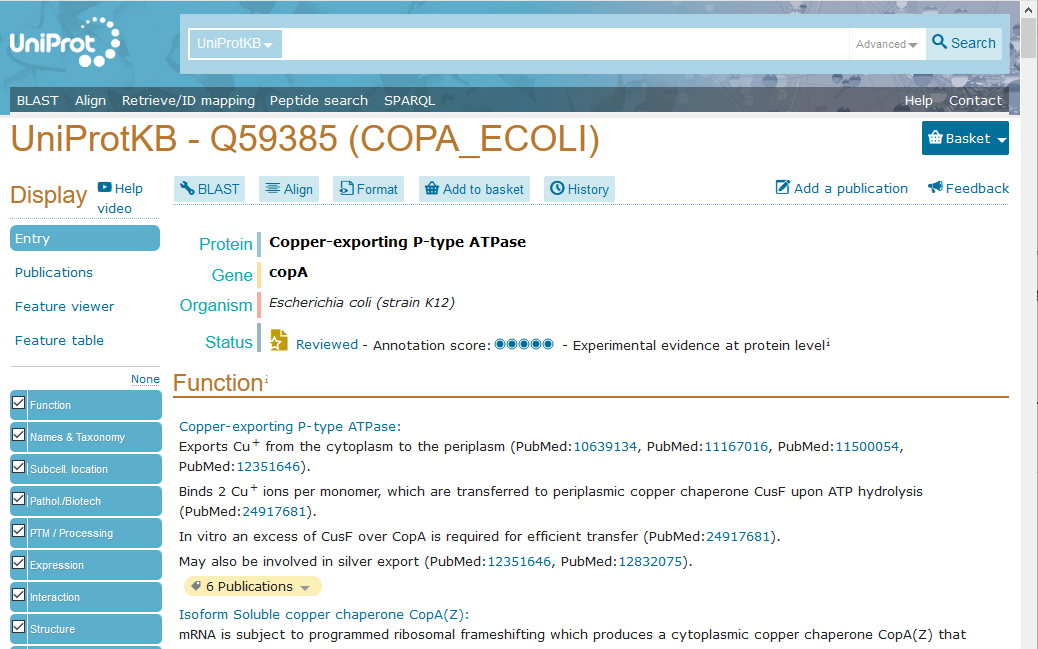
# A manual for detecting CopA of high confidence from metagenomes

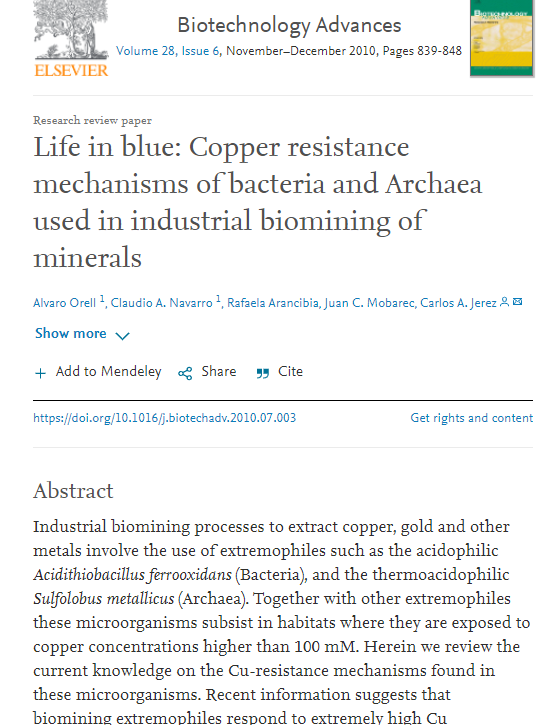
## Preparation of local CopA database

* 1. The local CopA database comprises of known CopA whose function have been tested with experimental evidence or crystal structure resolved. Some were from the copper resistant strain whose genomes have been fully sequenced.
  2. The sequences were retrieved from the Uniprot database (<https://www.uniprot.org/>). Specifically, go to the official website, search against the online database with the entry of ‘copper resistance copA’. The returned items were checked one by one, in terms of their annotation, experimental evidence and sequence features.



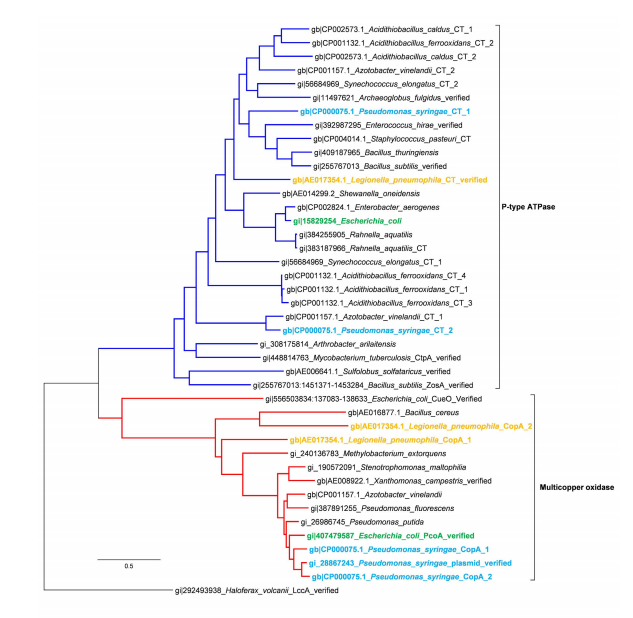


* 1. Known CopA were also searched in the literature where copper resistant bacteria were reported and their genomes were available for annotation in NCBI (https://www.ncbi.nlm.nih.gov/genome/).

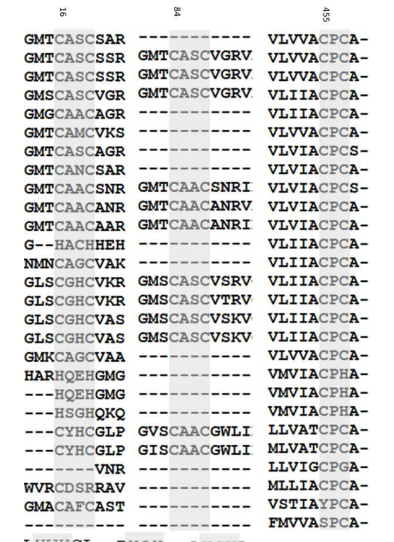




* 1. Finally a local database named ‘CopA\_reviewed.fasta’ was obtained.
  2. Phylogenetic analysis was performed to test their evolutionary relationship.

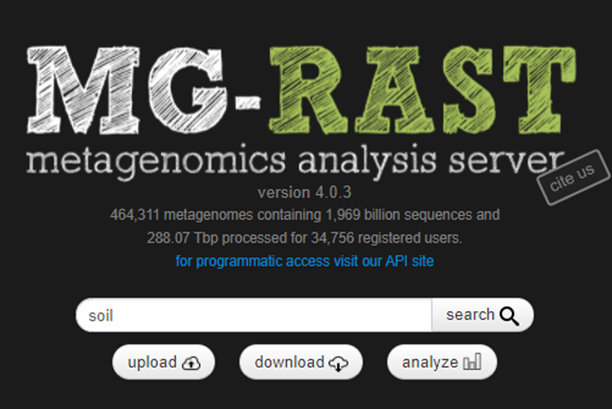


* 1. Further evolution trace analysis was done to see their length, conservative regions, key motifs for assessing novel CopA sequences.

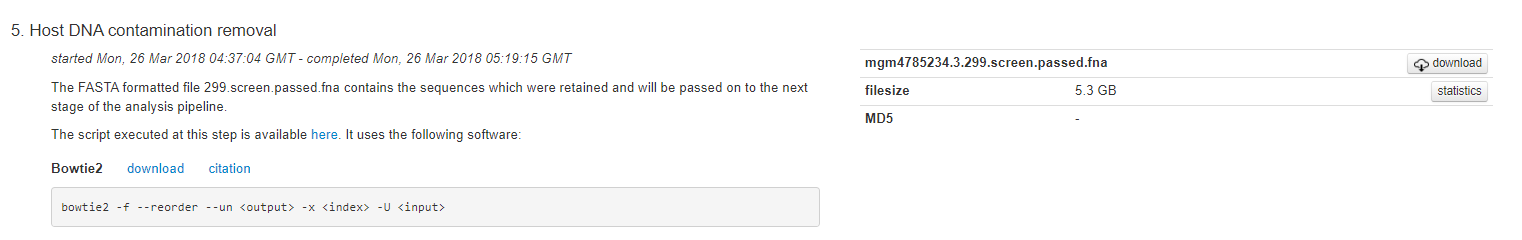


## Preparation of global metagenomes for local BLAST

* 1. Download data from MG-RAST (http://www.mg-rast.org/). Some metagenomes were downloaded as raw sequencing data, and some were processed protein assemblages.



* 1. Metadata of the metagenomes including the location, data size, etc. were compiled manually for reference.



* 1. Assembly of raw data using metaWRAP.
     1. Install metaWRAP from <https://github.com/bxlab/metaWRAP> v=1.2.1
     2. Install miniconda2 by

wgethttps://repo.continuum.io/miniconda/Miniconda2-latest-Linux-x86\_64.sh

bash Miniconda2-latest-Linux-x86\_64.sh

* + 1. Install metaWRAP in the conda environment

# Note: the order is important

conda config --add channels defaults

conda config --add channels conda-forge

conda config --add channels bioconda

conda config --add channels ursky

conda install -y -c ursky metawrap-mg

# Note: may take a while

# To fix the CONCOCT endless warning messages in metaWRAP=1.2, run

conda install -y blas=2.5=mkl

* + 1. Set up the conda environment

conda create -y -n metawrap-env python=2.7

conda activate metawrap-env

# Note: the order is important

conda config --add channels defaults

conda config --add channels conda-forge

conda config --add channels bioconda

conda config --add channels ursky

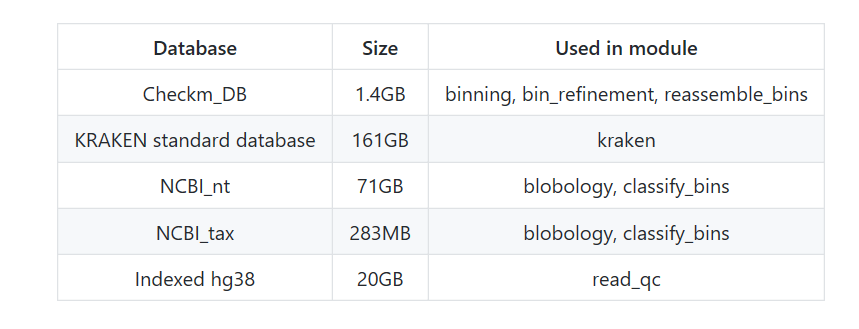
conda install -y -c ursky metawrap-mg

# Note: may take a while

# To fix the CONCOCT endless warning messages in metaWRAP=1.2, run

conda install -y blas=2.5=mkl

* + 1. Download the database to local



mkdir MY\_CHECKM\_FOLDER

checkm data setRoot

# CheckM will prompt to to chose your storage location... Give it the path to the folder you just made.

#Now manually download the database @ https://github.com/bxlab/metaWRAP/blob/master/installation/database\_installation.md cd MY\_CHECKM\_FOLDER

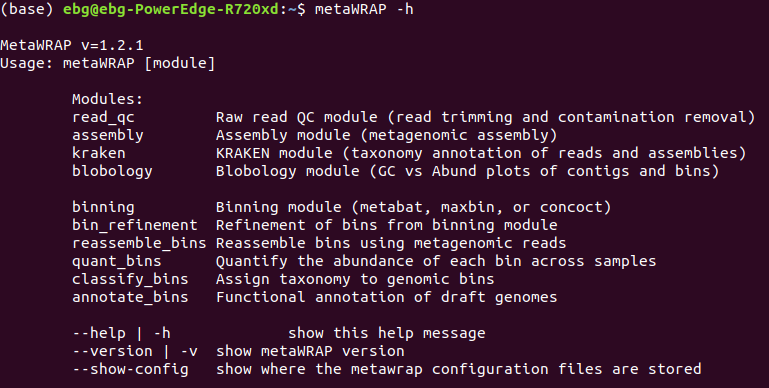
wget;https://data.ace.uq.edu.au/public/CheckM\_databases/checkm\_data\_2015\_01\_16.tar.gz

tar -xvf \*.tar.gz

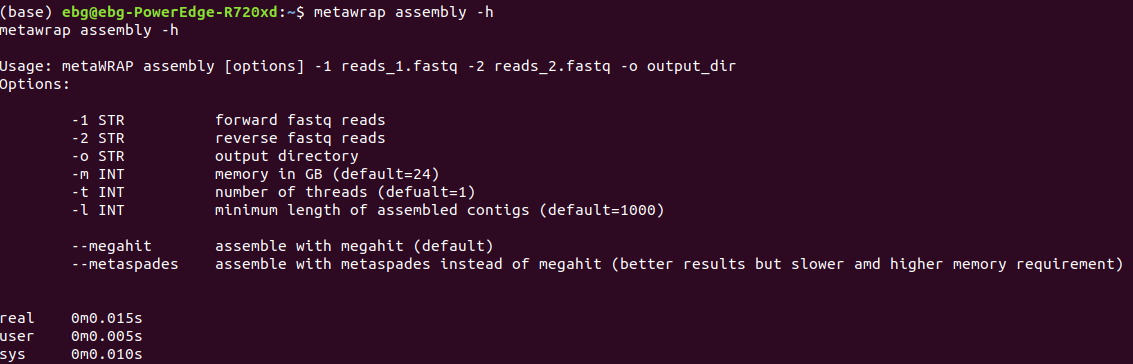
rm \*.gz

* + 1. Assemble the raw data

MataWRAP -h



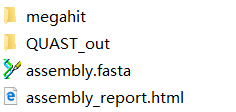
metawrap assembly -h



$ metawrap assembly -1 data\_1.fasta -2 data\_2.fasta -o output \_dir

##“data.fasta” is the dataset，“data\_1.fasta” is the forward sequences，“data\_2.fasta” is the reverse sequences.

# and you can view the output files here



## Local BLAST for detection of potential CopA

* 1. Download software blast 2.9.0 at ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/. Set up the user environmental variables. Create a new variable named BLASTDB in the user variable, and the value of the variable is C：\Blast\db. Add variable value C:\Blast\bin under the system variable “Path”.
  2. #Format the database

makeblastdb -in CopA\_reviewed.fasta -dbtype prot -title " CopA\_uniprot\_reviewed" -out NR

#Use NR as the local database, and assembly. fasta as subject database.

blastx.exe -db NR -query assembly.fasta -out assembly.out -evalue 0.000001 -max\_target\_seqs 5 -num\_threads 4 -outfmt6

blastp.exe -db NR -query dirp.fasta -out dirp.out -evalue 0.000001 -max\_target\_seqs 5 -num\_threads 4 -outfmt 6

## A file with .out will be obtained.

* 1. Check the sequence quality



Open .out file in this excel table, filter the sequences with proper similarity and bit score.

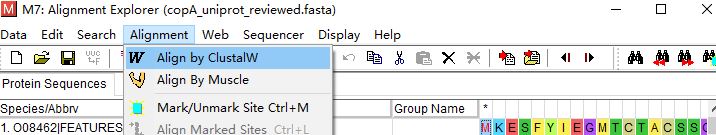
* 1. Manually curate the sequences of high quality, and retrieve the ORFs from the sequences through online ORFfinder (https://www.ncbi.nlm.nih.gov/orffinder/) and BLAST.



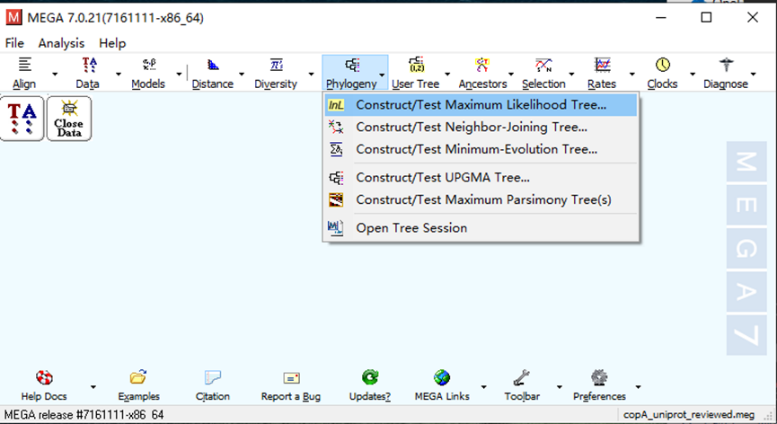
* 1. Collect the candidate *copA* genes and formed a new fasta file.



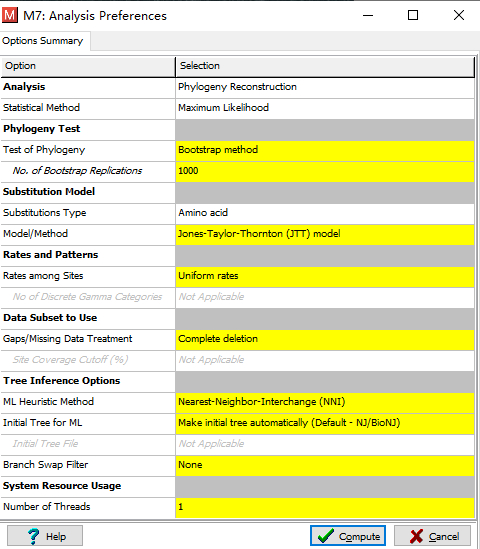
## Evolutionary trace analysis of known CopA



①



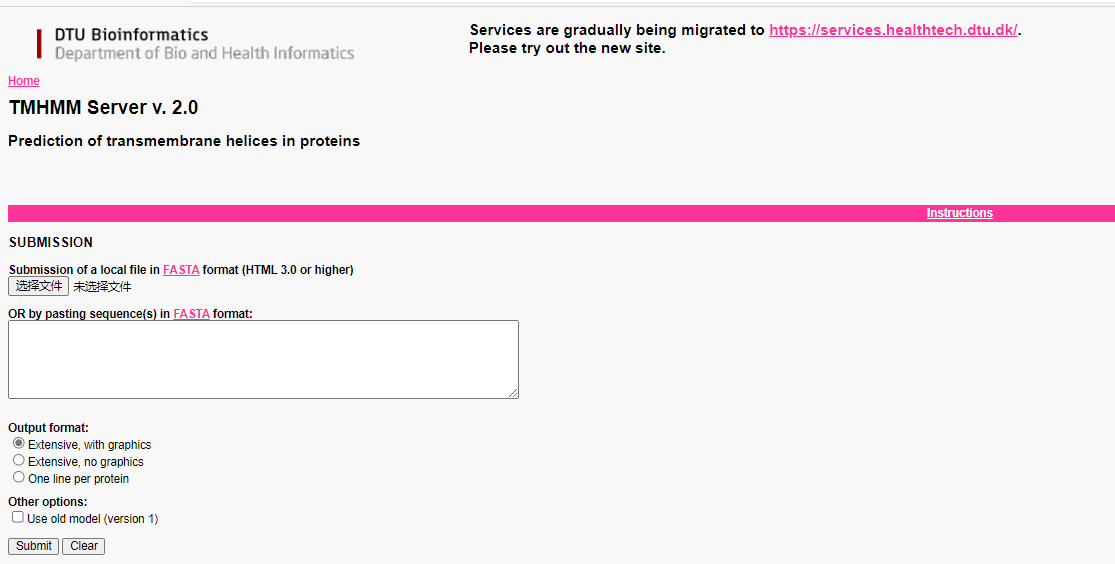
②



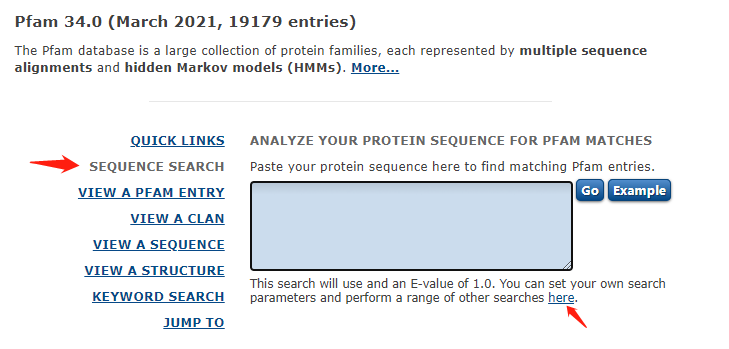
③

## Bioinformatic evaluation of potential CopA

* 1. Candidate ORFs with the length ranging from 500 to 900 amino acids were kept and subjected to the prediction of transmembrane helices using the TMHMM (http://www.cbs.dtu.dk/services/TMHMM-2.0/) online analysis platform.



* 1. Functional domains were then predicted using Pfam (http://pfam.xfam.org/) with an E-value of ≤ 1e-6, and sequences without metal transporting domains were eliminated.



## Phylogenetic analysis of candidate CopA

Same as evolutionary trace analysis of known CopA.

## Functional genomic verification of candidate CopA (details can be found in the main text)

* 1. Chemcial sysnthesis

Gene sequence optimization in order to make sure the expression of candidate *copA*. This is the key procedure of this step.

* 1. Preparation of host, positive control and MIC test

Synthesized genes were ligated with the PTR vector (with Kna and Amp resistance) individually, and transform the vector into *copA* defective strain JW-0473-4 to verify whether the genes can restore the copper resistance of the defective strain.

* 1. Drop assay and growth test

Experimental tests on the growth of the transformed strain by growing solid LB medium with Cd supplied.

