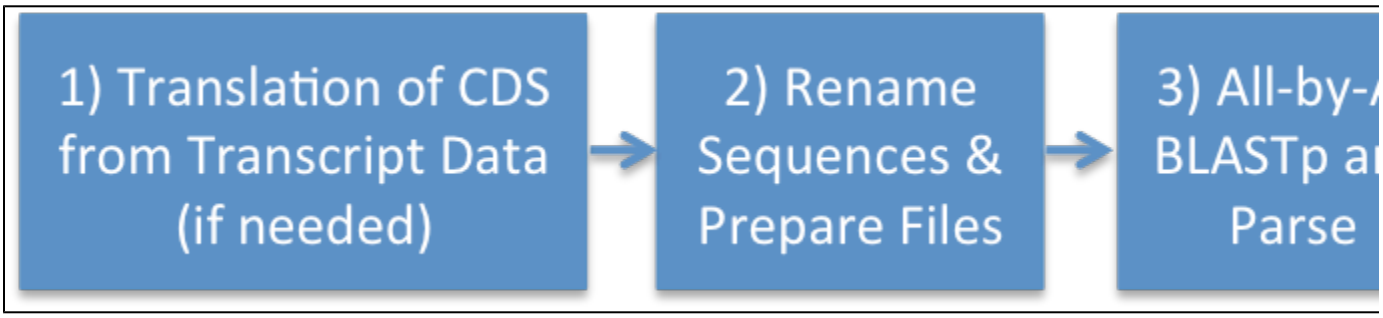
Cluster Orthologs and Paralogs and Assemble Custom Gene Sets (Workflow Tutorial)

Goal

Input entire protein-encoding gene or transcript repertoires from genomes of interest, and cluster homologs (orthologs and paralogs). Then, query clusters to assemble gene sets based on presence/absence and copy number.

Conceptual overview



Related tutorials

A draft tutorial is below, as a 'Workflow Overview'.

Workflow overview

Input and output test data for this workflow appears directly in the Discovery Environment in the Data window under *Community Data -> iplantcollaborative -> example_data -> homolog_clustering.*

- Directories are named for apps in the workflow. Files produced from an app at one step may have been renamed for use in subsequent apps (noted below).
- It is a good idea to keep track of the numbers of sequences and headers in your input files, and compare them to the outputs to ensure that output faithfully represents input.

1. **Step 1: Translation of CDS from Transcript Data: Transcript Decoder 1.0**
   - *Optional step if your data contains a species for which amino acid sequences are not available. If, instead, you have transcripts that you would like to search for CDS and include their amino acid sequences, Transdecoder can help. Transdecoder searches*

for CDS in transcript data and produces nucleotide and amino acid fasta files (among several other outputs) of detected CDS.

a. *Input*
   - *Individual fasta files of transcript data for each species' genome*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 0_Transcript_Decoder_1.0_input.*

b. *Output*
   - *Amino acid fasta file of detected CDS*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering-> 1_Transcript_Decoder_1.0_output*
       - *'best_candidates.eclipsed_orfs_removed.pep' was selected from the example output, renamed and used in step 2 below.*

- *Caveats*
  - *Examine Transdecoder documentation to select the appropriate output file for your experiment.*

2. ## Step 2: Rename Sequences and Prepare Input Files: fastaRename

a. *Input*
   - *Fasta files of amino acid sequences. One fasta file per species. Each file should represent, as nearly as possible, a complete protein-encoding gene repertoire.*
   - *User-defined 2-letter abbreviations for each input species genome.*
   - *Test input files are from 4 species: Plasmodium falciparum, Toxoplasma gondii, Neospora caninum, and Theileria annulata. Each file represents a 'complete' protein-encoding gene repertoire for that species. N. caninum sequences were produced from transcript data in step 1. For the interested, these are Apicomplexan species, chosen for their relatively small gene repertoires. Data are for testing purposes. The latest data for each species can be found at EuPath DB.*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 2_fastaRename_input*

b. *Output (for each species)*
   - *Fasta file of renamed amino acid sequences*
   - *GG file for use by OrthoMCL below*
   - *Mapping file*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 3_fastaRename_output*

- *Caveats*
  - *The sequences used as input can greatly affect downstream clustering output. Largely incomplete protein-encoding gene repertoires, or the inclusion of distantly related species, must be considered when interpreting output.*

3. ## Step 3: All-by-All BLASTp and Parse

- *This stage consists of 4 'sub steps'. Selected output from each of these serves as input for the next.*

a. Concatenate Multiple Files
   - *Separately for each file type, concatenate fasta, gg, and map files produced in the last step*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 4_Concatenate_Multiple_Files_output*

b. Create BLAST Database
   - *Create BLASTp database from concatenated fasta file in last step*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 5_Create_BLAST_Database_output*
   - *'condor-stdout-0' in example output is selected from app log files and shows the number of sequences formatted as part of the BLASTp database*

c. *All-By-All BLASTp of combined fasta file*
   - *Select Combined fasta file as query, and folder with BLASTp database as subject*
   - *Under 'Options' for this App, choose 'pairwise' for Output Format Format. This will provide the appropriate output for parsing in the next step.*
   - *'Under 'Options for this App, in the boxes 'Option 1:' and 'Option 2:' add the flags '-num_descriptions 250' and '-num_alig nments 250' respectively. This will limit the reported matches for each query to 250 and ensure that the number of one-line descriptors matches the number of alignments in the BLASTp output. If you feel you need more than 250 matches for each query, you may increase the number, but the numbers for the two flags must be equal or the BLAST parser will fail in the next step.*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 6_Blastp_output*

d. *parseBlastBpo to parse BLASTp output to produce OrthoMCL BPO file.*
   - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 7_parseBlastBpo_output -> parsedBLASToutput_bpo.txt*

- *Caveats*
  - *BLASTp parameters will affect clustering output*
  - *Once you are satisfied that your BLASTp output has been successfully parsed, you have the option to be a good data manager and delete the potentially large BLASTp output file. You can always recreate it later if you need it again. Take care not to accidentally delete the parsed BPO file, you need that for the next step.*

4. ## Step 4: Cluster Homologs (orthologs and paralogs), optionally add unclustered sequences to OrthoMCL output, generate reports on the number of clusters in and between species.

- *This stage consists of 3 'sub steps'. Input and output of each step are explained below.*

a. *Cluster homologs with OrthoMCL v1.4*
   - *Inputs*
     - *Concatenated GG file from step 3*
       - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 4_Concatenate_Multiple_Files_output -> GG_Combined.txt*
     - *BPO file from step 3*
       - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 7_parseBlastBpo_output -> parsedBLASToutput_bpo.txt*
   - *Outputs*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 8_OrthoMCL_output*
     - *See OrthoMCL v1.4 documentation for a description of the output files*
     - *Note that if all you need are OrthoMCL-generated homolog clusters, you can stop here. The example file '**all_orthomcl.out**' contains the final output of the OrthoMCL program. The remaining steps provide analysis and parsing of OrthoMCL output.*

b. *Add unclustered sequences with appendUnclustered - OPTIONAL but recommended*
   - *OrthoMCL does not include unclustered sequences in its output. For each species, some sequences will remain unclustered. If you intend to study sequences with no detected homologs you will want to add unclustered sequences to the OrthoMCL output. This step will add each unclustered sequence to the OrthoMCL output, each as a cluster with only one sequence. Remember to account for these manually added clusters later.*
   - *Inputs*
     - *Concatenated GG file from step 3*
       - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 4_Concatenate_Multiple_Files_output -> GG_Combined.txt*
     - *mcl directory from OrthoMCL output*
       - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 8_OrthoMCL_output -> Nov14 -> mcl/*
   - *Outputs*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 9_appendUnclustered_output*
     - *Seeappendunclustered documentation for a description of the output files*

- *Generate report and custom output files with clusterReport*
  - *Input for this step may or may not include unclustered sequences. SeeclusterReport documentation for a detailed description of input and output files directly related to this workflow.*

  a. *Inputs*
     - *Concatenated GG file from step 3*
       - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 4_Concatenate_Multiple_Files_output -> GG_Combined.txt*
     - *mcl directory, either fromOrthoMCL v1.4 (**without unclustered added**) or fromappendUnclustered (**with unclustered added**)*
       - *Example: **without unclustered added:** Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 8_OrthoMCL_output -> Nov_14*
       - *Example: **with unclustered added:** Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 9_appendUnclustered_output*

  b. *Outputs*
     - *SeeclusterReport documentation for a description of the output files*
       - *Example **without unclustered added:** Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 10_clusterReport_output -> without_unclustered_added*
       - *Example:**with unclustered added:** Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 10_clusterReport_output -> with_unclustered_added*
   - *Caveats*
     - *Remember to verify that the correct species, numbers of sequences, and numbers of clusters are represented in each output step. The log files are useful for this.*
     - *A word on unclustered sequences. If a sequence is not clustered, this means that no homologs have been detected. Note that this may not mean a sequence has no biological homologs, only that none were detected using these programs, species, and parameters.*

5. Step 5: Query Clusters and Assemble Custom Gene Sets with queryOrthoMCL

- *Input for this step may or may not include unclustered sequences. SeequeryOrthoMCL documentation for a detailed description of input and output files directly related to this workflow.*

a. *Inputs*
   - *Concatenated GG file generated in step 3*
     - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 4_Concatenate_Multiple_Files_output -> GG_Combined.txt*
   - *orthomcl.index file from OrthoMCL v1.4 (**without unclustered added**) or from appendUnclustered (**with unclustered added**)*
     - *Example: **without unclustered added:** Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 8_OrthoMCL_output -> Nov_14 -> mcl -> orthomcl.index*
     - *Example: **with unclustered added:** Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 9_appendUnclustered_output -> orthomcl.index*
   - *orthomcl.mclout file from OrthoMCL v1.4 (**without unclustered added**) or from appendUnclustered (**with unclustered added**)*

        - *Example: **without unclustered added:** Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 8_OrthoMCL_output -> Nov_14 -> mcl -> orthomcl.mclout*
        - *Example: **with unclustered added:** Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 9_appendUnclustered_output -> orthomcl.mclout*
    - *minMax.txt file created by User (see queryOrthoMCL documentation for an example)*
        - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 11_queryOrthoMCL_input -> minMax.txt*

b. *Outputs*
    - *Query.group file with clusters that meet search criteria*
        - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 12_queryOrthoMCL_output -> Query.group*
    - *queryOrthoMCL.log file analysis summary*
        - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> _2_queryOrthoMCL_output -> 0_queryOrthoMCL.log_*
    - *Caveats*
        - *Be careful to keep track of whether or not you are including unclustered results.*
        - *Some testing of the minMax.txt input file will help you understand the search criteria. Note that for both the min and max values, only clusters that exactly meet these criteria will be returned. For example, if you search for min=1 and max=3, only clusters with between 1 and 3 genes for that species will be returned. If you search for min=0 max =999999, clusters with any number of genes for that species will be returned (assuming that the species has less than 999999 genes).*

6. **Step 6: Map Fasta Headers to clusterReport and/or queryOrthoMCL output with flattenClusters**

a. *Input*
    - *.group file from either queryOrthoMCL or clusterReport*
        - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 12_queryOrthoMCL_output -> Query.group*
    - *Concatenated Map file generated in step 3*
        - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 4_Concatenate_Multiple_Files_output -> Map_Combined.txt*

b. *Output*
    - *flattened.txt file with one gene per line*
        - *Example: Community Data -> iplantcollaborative -> example_data -> homolog_clustering -> 13_flattenClusters_output -> flattened.txt*
    - *See flattenClusters documentation for a description of the output file format.*

- *Caveats*
    - *Inputs must come from either clusterReportorqueryOrthoMCLand notOrthoMCL v1.4*

More information

Additional applications in DE

- fastaRename
- parseBlastBpo
- OrthoMCL v1.4
- appendUnclustered
- clusterReport
- queryOrthoMCL
- flattenClusters

CI enhancements documented

Issues logged or use cases defined for CI development as a part of integrating this workflow

- *None as of 2/24/2015*

Community members assisted

- *One as of 2/24/2015*

Publications facilitated using this workflow

- *None as of 2/24/2015*

Datasets associated

- Input and output test data for this workflow appears directly in the Discovery Environment in the Data window under *Community Data -> iplantcollaborative -> example_data -> homolog_clustering*.

Data Commons requirements

- *None as of 2/24/2015*

Presentations

- *None as of 2/24/2015e*