trinity: de novo reconstruction

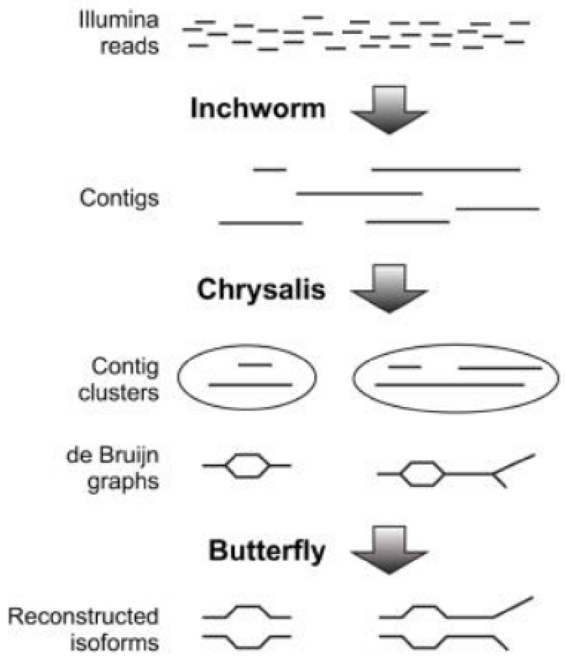
Program Description:

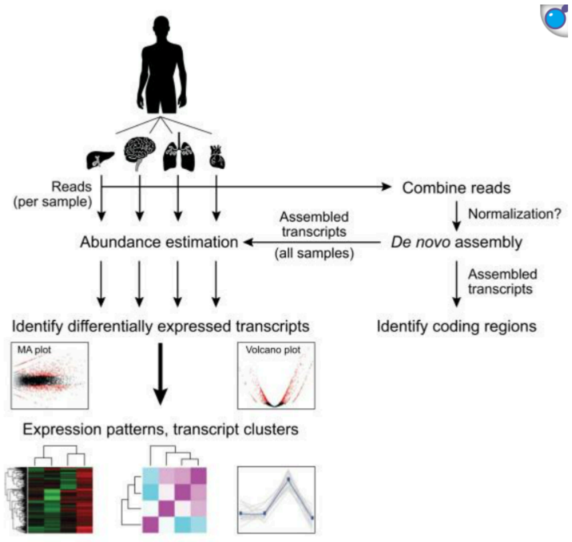
- Efficient and robust de novo reconstruction of transcriptomes from RNA-seq data
- To load for use in Linux environment
    - `module load trinityrnaseq`
    - Depending on where you're working there may be more than one version of trinityrnaseq available
    - # this shows which modules are available for loading
      `module avail trinityrnaseq`
- LOTS of intermediate files (500k - 1 mil)
- Output to /scratch
    - `option \”--output/scratch/trinity\”`
- Copy the fasta assembly file back to personal directory when Trinity has finished
- Can run trimmomatic through trinity

[RNA-Seq De novo Assembly Using Trinity]

Intro

- Trinity combines three independent software modules: Inchworm, Chrysalis, and Butterfly, applied sequentially to process large volumes of RNA-seq reads (don't need to know this, but thought it was cool)
    - Inchworm: assembles RNA-seq data into unique seq of transcripts
    - Chrysalis: clusters Inchworm contigs into clusters and constructs complete de Bruijn graphs for each cluster
        - Each cluster represents the full transcriptional complexity for a given gene
        - Then partitions the full read set among these disjoint graphs
    - Butterfly: processes the individual graphs in parallel, ultimately reporting full-length transcripts





De novo transcriptome assembly and analysis workflow

Running Trinity

# this is for multiple sets of fastq files that correspond to different types

```
Trinity --seqType fq --max_memory 50G  \
         --left <forward file> \
         --right <backward file> \
         --CPU 6 --output <file_name>
```

Intro to Trinity RNA-seq tutorial : This explains how to run Trinity in R

Step 1: Set up environment

1. Create Trinity environment (conda)
   ```
   module load conda
   conda config --add channels defaults
   conda config --add channels bioconda
   conda config --add channels conda-forge
   conda create --prefix ~/envs/Trinity
   ```
   # if ~/envs/Trinity does not exist, conda will create the directory for you
   ```
   conda activate ~/envs/Trinity
   conda install Trinity
   ```
   # proceed with installation, following the prompts (Y to download new packages)

   # once everything is downloaded all necessary dependencies necessary for running
2. Create `SBATCH` script using nano
   ```
   nano <script_name>
   ```
   # I make the script name species-specific

   # this will open `nano`, where you will write a script for the `BASH` session
   1. Start `BASH` session

      # this is done with `nano`
      ```
      #!/bin/bash
      #SBATCH --job-name=<choose_name>
      ```
      # I prefer to make my names species-specific (i.e. `trinity_Buckthorn`)
      ```
      #SBATCH --cpus-per-task=16
      #SBATCH --mem=100G
      #SBATCH --time=12:00:00
      #SBATCH --partition=interactive
      ```
      # can see available partitions with `sinfo`
      ```
      #SBATCH --output=<output_name.log>
      #SBATCH --error=<error_name.log>
      #SBATCH --mail-type=END
      #SBATCH --mail-user=<your_email@example.com>
      ```
      # yes, keep #'s
   2. Copy over Trinity source code from Robert Alvarez-Quinto shared directory
      ```
      cd /home/alvar419/shared/trinity
      scp -r trinityrnaseq-Trinity-v2.15.2 <path to destination>
      ```
      # personally, for my destination file I do `~/src`

      # Now we should be able to run Trinity

c. Run Trinity for paired-end FASTQ files
```
Trinity --seqType fq \
     --max_memory 90G \
     -<left reads_1.fq.gz> \
     -<right reads_2.fq.gz> \
     --CPU 16 \
     --output <trinity_output_dir>
```
# --seqType fq: specifies input as FASTQ format
# --max_memory 100G: make this a little less than requested for your `SBATCH`
# --left and --right: input files for paired-end reads
# --CPU 16: number of CPU cores to use (adjust based on availability)
# --output trinity_output: directory where results will be stored
    # Trinity will automatically create the output directory
    # ensure the output will be created in `scratch.global`
    # this will create LOTS of intermediate files
    # name must include 'trinity', otherwise there will be an error
    # I strongly recommend making the output file species-specific, aka "buckthorn_trinity_output"

d. Now we can safely log out of MSI and have the program running in the background!

Step 2: Run Trinity!

3. Close and save `SBATCH nano` script
    a. `ctrl+X`
       # this closes `nano`
    b. `Y`
       # confirm to save script
    c. `<sbatch_script_name.sh>`
       # type in a name
    d. `Enter`
4. Activate your `SBATCH`
```
sbatch <sbatch_script_name.sh>
```

Step 3: Monitor the job

5. Monitor progress: trinity outputs progress logs
   # to view progress logs
```
squeue -u <user_name>
```
    a. To cancel SBATCH job
```
scancel <JOBID>
```
       # check `<JOBID>` w/ `squeue` command

   # to check output and error files
```
tail -f <output.log>
tail -f <error.log>
```

Step 4: Check results

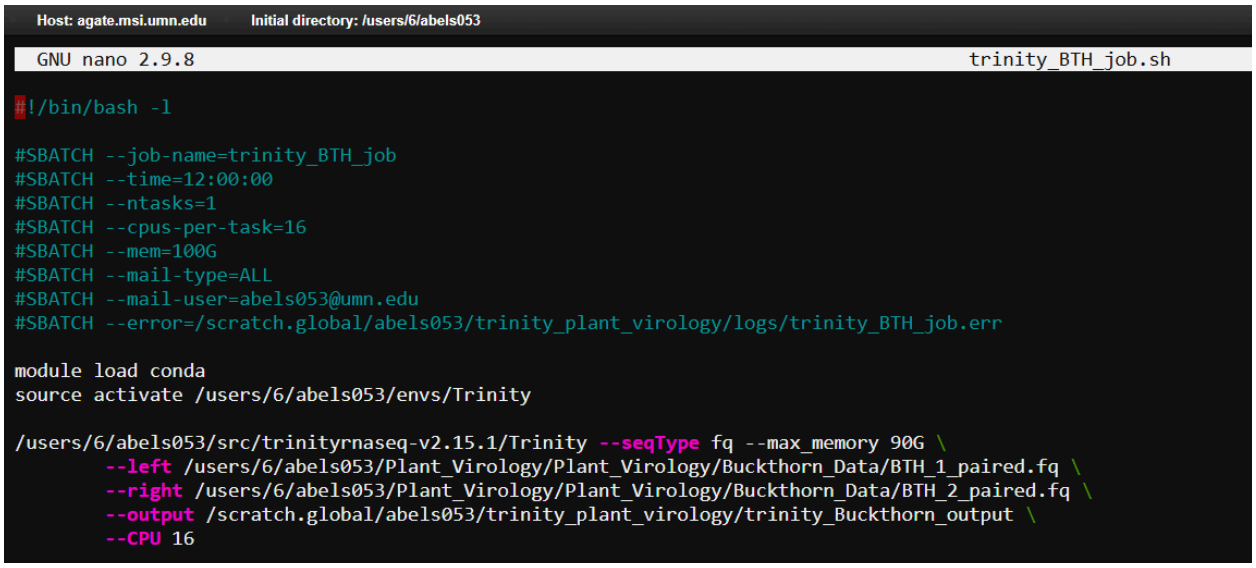
6. Once Trinity completes, the output directory (in scratch.global) will contain
    a. `Trinity.fasta` = final assembled transcriptome

    # This is the ONLY important file that needs to be copied over
    b. Logs and intermediate files
    # you do NOT need these
7. Optional: Check the assembly stats

Step 6: Copy results back to home directory

8. `cp -r <output_file.fasta> <path to local/home directory>`

Screenshots:



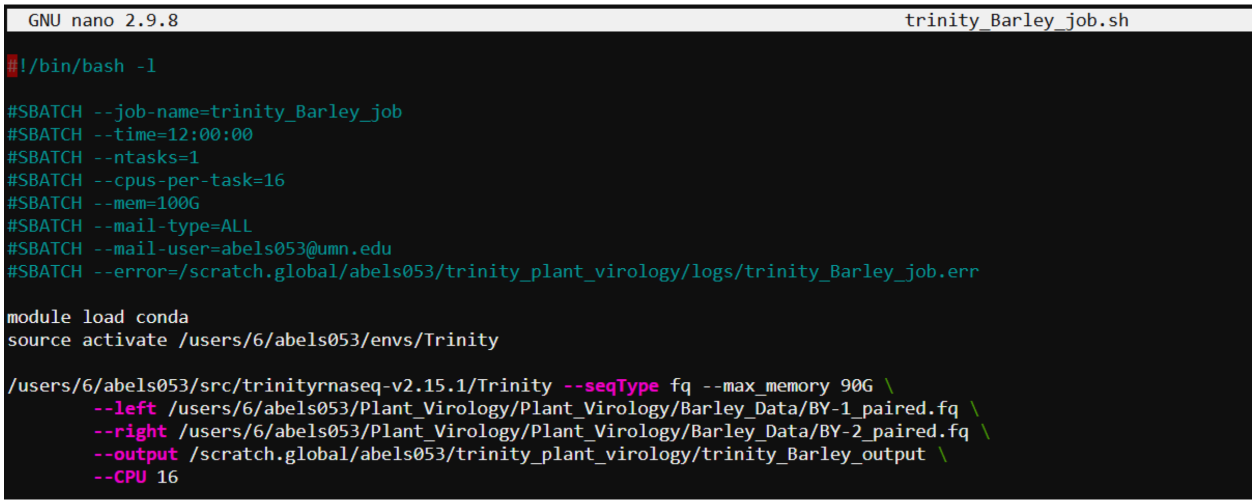
```
Host: agate.msi.umn.edu     Initial directory: /users/6/abels053
 GNU nano 2.9.8                                                          trinity_BTH_job.sh

#!/bin/bash -l

#SBATCH --job-name=trinity_BTH_job
#SBATCH --time=12:00:00
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=16
#SBATCH --mem=100G
#SBATCH --mail-type=ALL
#SBATCH --mail-user=abels053@umn.edu
#SBATCH --error=/scratch.global/abels053/trinity_plant_virology/logs/trinity_BTH_job.err

module load conda
source activate /users/6/abels053/envs/Trinity

/users/6/abels053/src/trinityrnaseq-v2.15.1/Trinity --seqType fq --max_memory 90G \
        --left /users/6/abels053/Plant_Virology/Plant_Virology/Buckthorn_Data/BTH_1_paired.fq \
        --right /users/6/abels053/Plant_Virology/Plant_Virology/Buckthorn_Data/BTH_2_paired.fq \
        --output /scratch.global/abels053/trinity_plant_virology/trinity_Buckthorn_output \
        --CPU 16
```



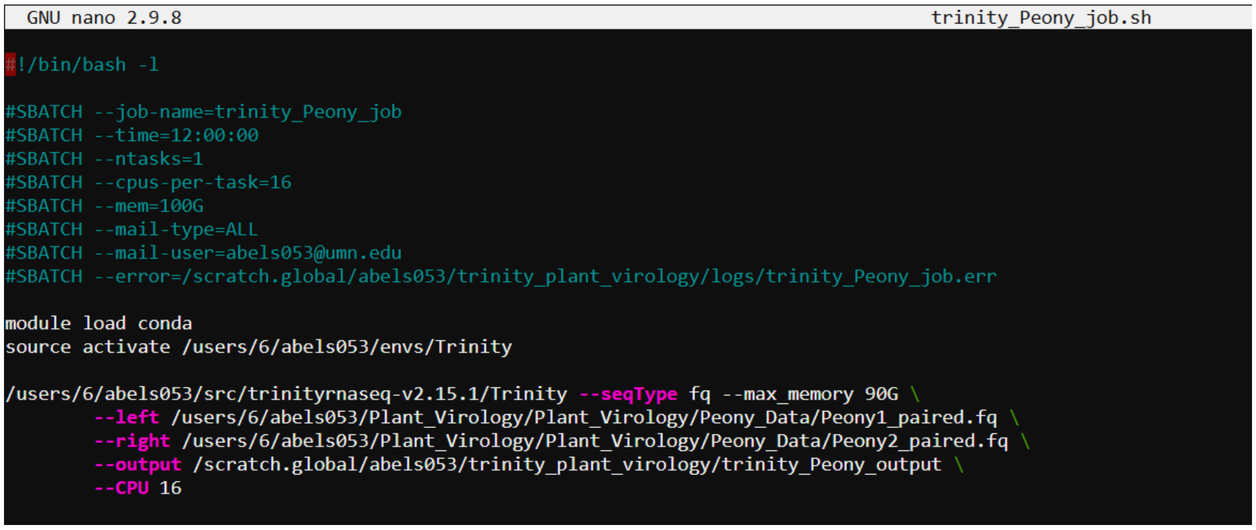
```
 GNU nano 2.9.8                                                          trinity_Barley_job.sh

#!/bin/bash -l

#SBATCH --job-name=trinity_Barley_job
#SBATCH --time=12:00:00
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=16
#SBATCH --mem=100G
#SBATCH --mail-type=ALL
#SBATCH --mail-user=abels053@umn.edu
#SBATCH --error=/scratch.global/abels053/trinity_plant_virology/logs/trinity_Barley_job.err

module load conda
source activate /users/6/abels053/envs/Trinity

/users/6/abels053/src/trinityrnaseq-v2.15.1/Trinity --seqType fq --max_memory 90G \
        --left /users/6/abels053/Plant_Virology/Plant_Virology/Barley_Data/BY-1_paired.fq \
        --right /users/6/abels053/Plant_Virology/Plant_Virology/Barley_Data/BY-2_paired.fq \
        --output /scratch.global/abels053/trinity_plant_virology/trinity_Barley_output \
        --CPU 16
```



```
 GNU nano 2.9.8                                                          trinity_Peony_job.sh

#!/bin/bash -l

#SBATCH --job-name=trinity_Peony_job
#SBATCH --time=12:00:00
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=16
#SBATCH --mem=100G
#SBATCH --mail-type=ALL
#SBATCH --mail-user=abels053@umn.edu
#SBATCH --error=/scratch.global/abels053/trinity_plant_virology/logs/trinity_Peony_job.err

module load conda
source activate /users/6/abels053/envs/Trinity

/users/6/abels053/src/trinityrnaseq-v2.15.1/Trinity --seqType fq --max_memory 90G \
        --left /users/6/abels053/Plant_Virology/Plant_Virology/Peony_Data/Peony1_paired.fq \
        --right /users/6/abels053/Plant_Virology/Plant_Virology/Peony_Data/Peony2_paired.fq \
        --output /scratch.global/abels053/trinity_plant_virology/trinity_Peony_output \
        --CPU 16
```

Step 1: Set up environment

1. Copy Trinity into personal directory:
   ```
   git clone https://github.com/trinityrnaseq/trinityrnaseq.git
   cd trinityrnaseq
   ```
2. Ensure all submodules are available and updated in order to run Trinity
   ```
   git submodule update --init --recursive
   ```
3. Ensure that all necessary modules are loaded.
   ```
   module load samtools
   module load jellyfish
   module load bowtie2
   module load salmon
   ```
4. Next, compile Trinity and its associated tools.
   # Make sure you're in your trinityrnaseq directory from before.
   ```
   make
   ```
5. Optional: double check that Trinity has been properly downloaded:
   ```
   ./Trinity --version
   ```
6. Connect to `/scratch.global/`
   # create a directory for yourself
   # since `scratch.global` is accessible by all MSI users
   ```
   mkdir <your_username>
   ```
   # change directory to recently created directory
   ```
   cd /scratch/<your_username>
   ```
   # you should now be in /scratch.global/<your_username>
   # you will now remain here for the entirety of running and working with Trinity
7. Create a working directory
   ```
   mkdir <trinity_project>
   cd <trinity_project>
   ```
   # you should now be in
   ```
    /scratch.global/<your_username>/<working_directory>
   ```