Adding a New Tool To Galaxy

Tool Name: Racon-GPU

Racon: Consensus module for raw de novo DNA assembly of long uncorrected reads.
The assembly of long reads from Pacific Biosciences and Oxford Nanopore Technologies typically requires resource-intensive error-correction and consensus-generation steps to obtain high-quality assemblies. The authors show that the error-correction step can be omitted and that high-quality consensus sequences can be generated efficiently with a SIMD-accelerated, partial-order alignment–based, stand-alone consensus module called Racon. Based on tests with PacBio and Oxford Nanopore data sets, they show that Racon used with miniasm enables consensus genomes with similar or better quality than state-of-the-art methods while being an order of magnitude faster.
Racon takes as input only three files: contigs in FASTA/FASTQ format, reads in FASTA/FASTQ format and overlaps/alignments between the reads and the contigs in SAM/PAF format. Output is a set of polished contigs in FASTA format.
Racon makes use of NVIDIA's ClaraGenomicsAnalysis SDK for CUDA accelerated polishing and alignment.
We ran the racon tool with 8 threads for both cpu and gpu. The execution without gpu support was 31 seconds and with gpu support was 27 seconds (for the gpu execution, the Number of batches for CUDA accelerated polishing was given as 5).

Adding a New Tool

- Add yourself as an administrator to the tool.
  - Edit line 1361 of config/galaxy.yml by adding your email: admin_users:user1@example.com . Then, restart Galaxy and check if the admin toolbar is existing or not.
- Adding Racon tool:
  - Requirements: gcc 4.8+ or clang 3.4+, cmake 3.2+ | CUDA Support, gcc 5.0+, cmake 3.10+, CUDA 9.0+
  - cd tools
- Get the Racon repository to tools directory and build it:
  - git clone --recursive https://github.com/lbcb-sci/racon racon
    cd racon
    mkdir build
    cd build
    cmake -DCMAKE_BUILD_TYPE=Release ..
  - or for GPU supported build, type in the command:
    cmake -DCMAKE_BUILD_TYPE=Release -Dracon_enable_cuda=ON ..
  - then, type in:
    make
  - The above command will create an executable in the bin file, which is inside the build (current) folder.
- Test the tool:

  cd bin

  ./racon ../../test/data/sample_reads.fastq.gz
../../test/data/sample_overlaps.paf.gz ../../test/data/sample_layout.fasta.gz

Racon.xml

```
<tool id="racon" name="Racon" version="1.3.1.1">
  <description>Consensus module for raw de novo DNA assembly of long uncorrected reads.</description>
  <command detect_errors="exit_code"><![CDATA[
    ln -s '$reads' reads.${reads.ext} &&
    #if $overlaps.ext == 'sam':
      ln -s '$overlaps' overlaps.${overlaps.ext} &&
    #else:
      ln -s '$overlaps' overlaps.paf &&
    #end if
    ln -s '$corrected_reads' corrected_reads.${corrected_reads.ext} &&

  #if  str($gpu_enable) == "gpu_true"
       <dir_of_gpu_executable>/racon
  #elif  str($gpu_enable) == "gpu_false"
       <dir_of_executable>/racon
  #end if#

       reads.${reads.ext}
       #if $overlaps.ext == 'sam':
         overlaps.${overlaps.ext}
       #else:
         overlaps.paf
       #end if
       corrected_reads.${corrected_reads.ext}
       ...
    -t $t
  #if  str($gpu_enable) == "gpu_true"
       -c $c
  #end if#
       > racon_polished_consensus.fa
  ]]></command>
  <inputs>
    ...
    <param argument="-t" type="integer" value="1" label="Number of Threads" />
  <param argument="-c" type="integer" value="1" label="Number of batches for CUDA accelerated polishing"
     />
  <param name="gpu_enable" type="boolean" truevalue="gpu_true" falsevalue="gpu_false" checked="False"
     label="Do you want GPU support?" />
  </inputs>
  <outputs>
    <data name="consensus" format="fasta" from_work_dir="racon_polished_consensus.fa" />
  </outputs>
</tool>
```

We must then make Galaxy aware of the Racon tool. To do this, we will add a section to the tool_conf.xml file, which is in the config/ directory:

```
<section name="racon" id="racon">
      <tool file="racon/racon.xml" />
</section>
```

Then, click the Analyze Data tab. On the left, click Get Data, and then "Upload File from your computer" from the dropdown menu, an upload bin will pop-up. Then, from the file explorer, open galaxy/tools/racon/test/data directory and drag and drop all the .gz files to the upload bin in Galaxy.

These are the input files that we must give to Racon to test its functionality using Galaxy. Now, in the Galaxy tool, on the left pane, there should be a search box for searching the tools. Search the name of the tool you added and test it.

Data Preparation for Racon

- *Download sample PacBio from the PBcR website*
  wget -O- http://www.cbcb.umd.edu/software/PBcR/data/selfSampleData.tar.gz | tar zxf -
- *Install minimap and miniasm (requiring gcc and zlib)*
  git clone https://github.com/lh3/minimap2 && (cd minimap2 && make)
  git clone https://github.com/lh3/miniasm && (cd miniasm && make)
- *Overlap for PacBio reads (or use "-x ava-ont" for nanopore read overlapping)*
  minimap2/minimap2 -x ava-pb -t8 selfSampleData/pacbio_filtered.fastq
selfSampleData/pacbio_filtered.fastq | gzip -1 > reads.paf.gz
- *Layout*
  miniasm/miniasm -f selfSampleData/pacbio_filtered.fastq reads.paf.gz > reads.gfa
- *Convert to fasta*
  head -n 1 reads.gfa | awk '{print ">"$2; print $3}' > assembly.fasta
- *We first use minimap again, this time with the original reads mapped against the 'raw' assembly:*
  minimap assembly.fasta selfSampleData/pacbio_filtered.fastq > reads_mapped.paf
- *Then use racon*
  racon -t 2 selfSampleData/pacbio_filtered.fastq reads_mapped.paf assembly.fasta output_name.fasta