How to Accelerate Genome Analysis by Using NVIDIA Parabricks

Minerva Scientific Computing Environment

https://labs.icahn.mssm.edu/minervalab

Hyung Min Cho, PhD
The Minerva HPC Team

November 6, 2024

Icahn
School of
Medicine at
Mount
Sinai

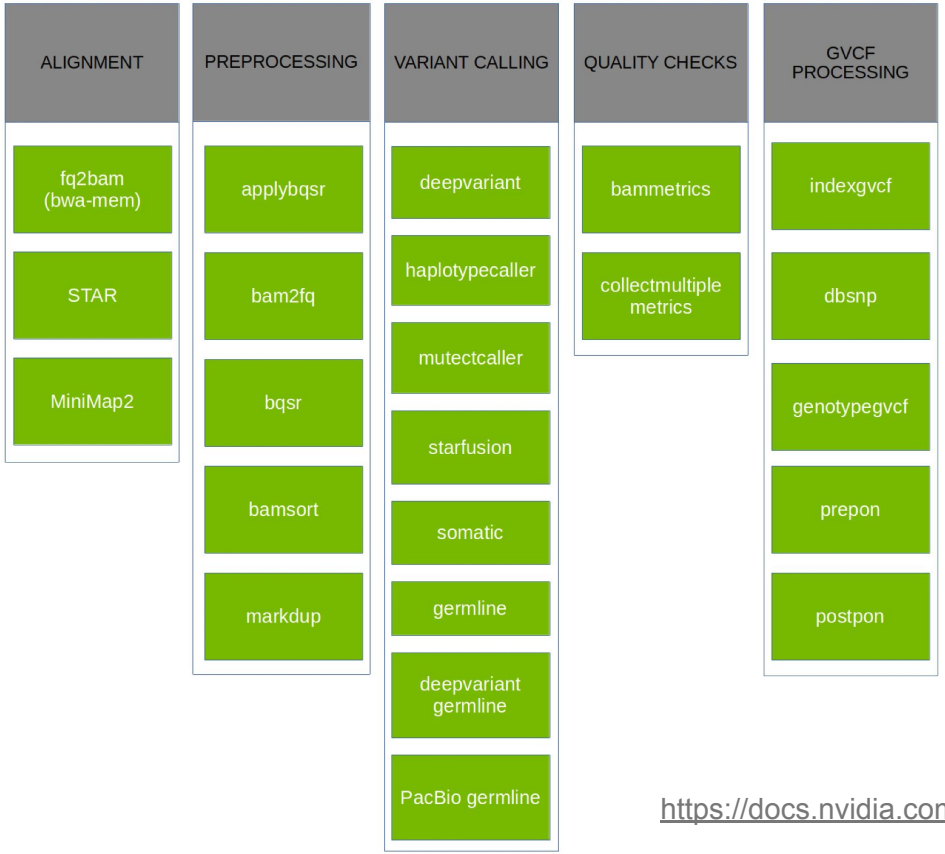
What is NVIDIA Parabricks?

A software suite designed to accelerate genomic data analysis by leveraging GPU (graphics processing unit) computing.

- High-speed genomic analysis
- Integrates with widely used genomics analysis tools and pipelines (such as GATK)
- Compatible with multiple bioinformatics formats
- Preserves the accuracy of standard CPU-based genomics workflows
- Compatible with common workflow managers WDL and NextFlow (https://github.com/clara-parabricks-workflows)

https://docs.nvidia.com/clara/parabricks/4.3.0/

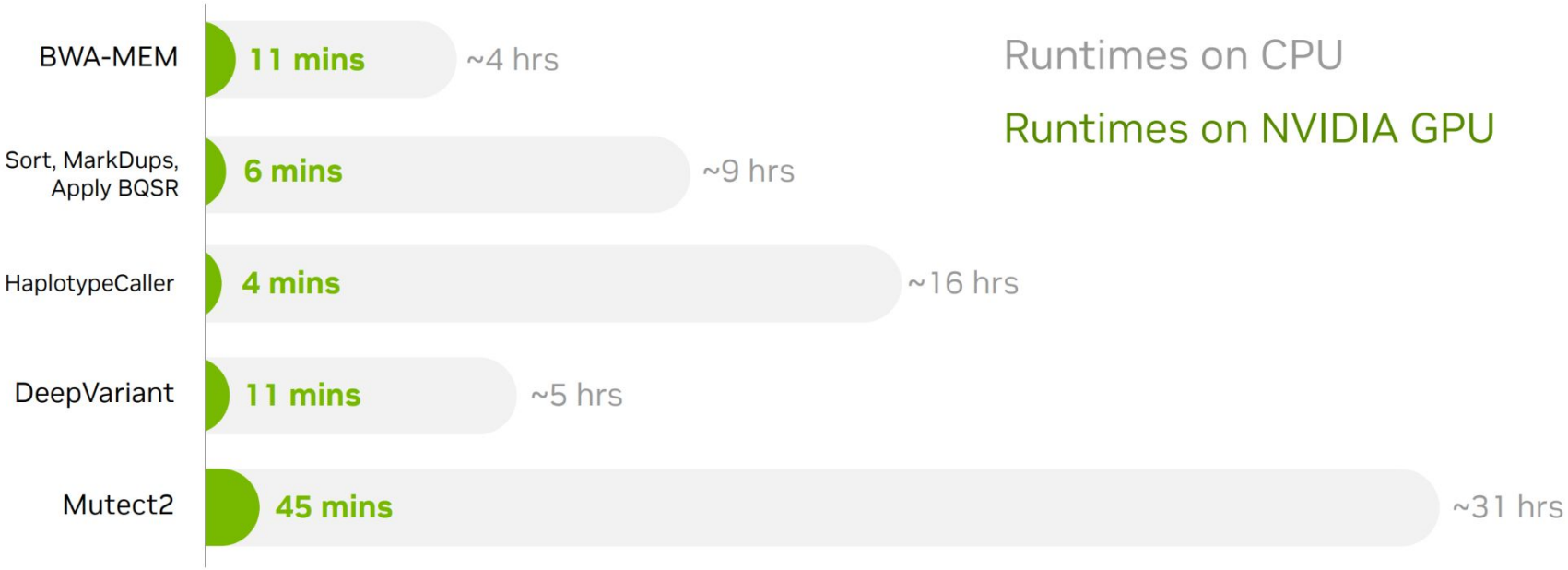
Tools Supported by Parabricks



https://docs.nvidia.com/clara/parabricks/4.3.0/toolreference.html

Up to 80x Acceleration

Gold-standard results, faster



v3.8 Benchmarks
Dataset: HG002 30x WGS, except Mutect2 on SEQC2 50x WGS
CPU: m5.24xlarge; GPU: 8xA100, except DeepVariant & Mutect2 on 8xV100

NVIDIA

Parabricks Testing in the Command Line

```
[choh07@li03c02 ~]$ bsub -P acc_hpcstaff -q gpu -gpu num=1 -R a100 -n 8 -R span[hosts=1] -R rusage[mem=8GB] -W 1:00 -XF -Is /bin/bash
Job <144134351> is submitted to queue <gpu>.
<<ssh X11 forwarding job>>
<<Waiting for dispatch ...>>
<<Starting on lg07c05.chimera.hpc.mssm.edu>>
[choh07@lg07c05 ~]$
```

```
[choh07@lg07c05 ~]$ ml parabricks
The singularity image is /hpc/packages/minerva-centos7/parabricks/4.3.0-1/clara-parabricks_4.3.0-1.sif.
[choh07@lg07c05 ~]$
[choh07@lg07c05 ~]$ PARABRICKS=/hpc/packages/minerva-centos7/parabricks/4.3.0-1/clara-parabricks_4.3.0-1.sif
[choh07@lg07c05 ~]$
[choh07@lg07c05 ~]$ ml

Currently Loaded Modules:
  1) singularity/3.6.4   2) cuda/12.1.1   3) parabricks/4.3.0-1
```

Parabricks Testing in the Command Line

```
[choh07@lg07c05 ~]$ singularity exec $PARABRICKS pbrun --help
Please visit https://docs.nvidia.com/clara/#parabricks for detailed documentation

usage: pbrun <command> [<args>]
Help: pbrun -h

command can be a TOOL or FULL PIPELINE. Example:
pbrun fq2bam --ref genome.fa --in-fq sample_1.fq.gz sample_2.fq.gz --out-bam sample.bam
pbrun germline --ref genome.fa --in-fq sample_1.fq.gz sample_2.fq.gz --out-bam sample.bam --out-variants sample.vcf

command options for standalone TOOL

applybqsr               - Apply BQSR report to a BAM file and generate a new BAM file
bam2fq                  - Convert a BAM file to FASTQ
bammetrics              - Collect WGS Metrics on a BAM file
bamsort                 - Sort a BAM file
bqsr                    - Collect BQSR report on a BAM file
collectmultiplemetrics  - Collect multiple classes of metrics on a BAM file
dbsnp                   - Annotate variants based on a dbsnp
deepvariant             - Run GPU-DeepVariant for calling germline variants
fq2bam                  - Run bwa mem, co-ordinate sorting, marking duplicates, and Base Quality Score Recalibration
fq2bam_meth             - Run GPU-accelerated bwa-meth compatible alignment, co-ordinate sorting, marking duplicates, and Base Quality Score Recalibration
fq2bamfast              - Run newly optimized version of bwa mem, co-ordinate sorting, marking duplicates, and Base Quality Score Recalibration
genotypegvcf            - Convert a GVCF to VCF
haplotypecaller         - Run GPU-HaplotypeCaller for calling germline variants
indexgvcf               - Index a GVCF file
markdup                 - Identifies duplicate reads
minimap2                - Align long read sequences against a large reference database to convert FASTQ to BAM/CRAM
mutectcaller            - Run GPU-Mutect2 for tumor-normal analysis
postpon                 - Generate the final VCF output of doing mutect pon
prepon                  - Build an index for PON file, which is the prerequisite to performing mutect pon
rna_fq2bam              - Run RNA-seq data through the fq2bam pipeline
starfusion              - Identify candidate fusion transcripts supported by Illumina reads

command options for commonly used FULL PIPELINES

deepvariant_germline    - Run the germline pipeline from FASTQ to VCF using a deep neural network analysis
pacbio_germline         - Run the germline pipeline from FASTQ to VCF by aligning long read sequences with minimap2 and using a deep neural network analysis
germline                - Run the germline pipeline from FASTQ to VCF
somatic                 - Run the somatic pipeline from FASTQ to VCF

Information about the software
```

Parabricks Testing in the Command Line

```
[choh07@lg07c05 ~]$ singularity exec $PARABRICKS pbrun bam2fq --help
Please visit https://docs.nvidia.com/clara/#parabricks for detailed documentation

usage: pbrun bam2fq [<args>]
Help: pbrun bam2fq -h

Run bam2fq to convert BAM/CRAM to FASTQ.

optional arguments:
  -h, --help            show this help message and exit

Input Output file options:
  Options for Input and Output files for this tool.

  --ref REF             Path to the reference file. This argument is only required for CRAM input. (default: None)
  --in-bam IN_BAM       Path to the input BAM/CRAM file to convert to fastq.gz. (default: None)
  --out-prefix OUT_PREFIX
                        Prefix filename for output fastq files. (default: None)

Tool Options:
  Options specific to the tool.

  --out-suffixF OUT_SUFFIXF
                        Output suffix used for paired reads that are first in pair. The suffix must end with ".gz". (default: _1.fastq.gz)
  --out-suffixF2 OUT_SUFFIXF2
                        Output suffix used for paired reads that are second in pair. The suffix must end with ".gz". (default: _2.fastq.gz)
  --out-suffixO OUT_SUFFIXO
                        Output suffix used for orphan/unmatched reads that are first in pair. The suffix must end with ".gz". If no suffix is provided,
                        these reads will be ignored. (default: None)
  --out-suffixO2 OUT_SUFFIXO2
                        Output suffix used for orphan/unmatched reads that are second in pair. The suffix must end with ".gz". If no suffix is provided,
                        these reads will be ignored. (default: None)
  --out-suffixS OUT_SUFFIXS
                        Output suffix used for single-end/unpaired reads. The suffix must end with ".gz". If no suffix is provided, these reads will be
                        ignored. (default: None)
  --rg-tag RG_TAG       Split reads into different fastq files based on the read group tag. Must be either PU or ID. (default: None)
  --remove-qc-failure   Remove reads from the output that have abstract QC failure. (default: None)

Performance Options:
```

Parabricks Testing in the Command Line

```
[choh07@lg07c05 ~]$ singularity exec --nv $PARABRICKS pbrun fq2bam -h
Please visit https://docs.nvidia.com/clara/#parabricks for detailed documentation


usage: pbrun fq2bam [<args>]
Help: pbrun fq2bam -h

Run GPU-bwa mem, co-ordinate sorting, marking duplicates, and Base Quality Score Recalibration to convert FASTQ to BAM/CRAM.

optional arguments:
  -h, --help            show this help message and exit

Input Output file options:
  Options for Input and Output files for this tool.

  --ref REF             Path to the reference file. (default: None)
  --in-fq [IN_FQ [IN_FQ ...]]
                        Path to the pair-ended FASTQ files followed by optional read groups with quotes (Example:
                        "@RG\tID:foo\tLB:lib1\tPL:bar\tSM:sample\tPU:foo"). The files must be in fastq or fastq.gz format. All sets of inputs should have a read
                        group; otherwise, none should have a read group, and it will be automatically added by the pipeline. This option can be repeated multiple
                        times. Example 1: --in-fq sampleX_1_1.fastq.gz sampleX_1_2.fastq.gz --in-fq sampleX_2_1.fastq.gz sampleX_2_2.fastq.gz. Example 2: --in-fq
                        sampleX_1_1.fastq.gz sampleX_1_2.fastq.gz "@RG\tID:foo\tLB:lib1\tPL:bar\tSM:sample\tPU:unit1" --in-fq sampleX_2_1.fastq.gz
                        sampleX_2_2.fastq.gz "@RG\tID:foo2\tLB:lib1\tPL:bar\tSM:sample\tPU:unit2". For the same sample, Read Groups should have the same sample
                        name (SM) and a different ID and PU. (default: None)
  --in-se-fq [IN_SE_FQ [IN_SE_FQ ...]]
                        Path to the single-ended FASTQ file followed by optional read group with quotes (Example:
                        "@RG\tID:foo\tLB:lib1\tPL:bar\tSM:sample\tPU:foo"). The file must be in fastq or fastq.gz format. Either all sets of inputs have a read
                        group, or none should have one, and it will be automatically added by the pipeline. This option can be repeated multiple times. Example
                        1: --in-se-fq sampleX_1.fastq.gz --in-se-fq sampleX_2.fastq.gz . Example 2: --in-se-fq sampleX_1.fastq.gz
                        "@RG\tID:foo\tLB:lib1\tPL:bar\tSM:sample\tPU:unit1" --in-se-fq sampleX_2.fastq.gz "@RG\tID:foo2\tLB:lib1\tPL:bar\tSM:sample\tPU:unit2" .
                        For the same sample, Read Groups should have the same sample name (SM) and a different ID and PU. (default: None)
  --in-fq-list IN_FQ_LIST
```

Parabricks Testing in the Command Line

```
[choh07@lg07c05 ~]$ singularity exec --nv --bind $DATA_DIR:$DATA_DIR $PARABRICKS pbrun fq2bam --ref $REF --in-fq $INP_fq1 $INP_fq2 --out-bam $OUT_BAM --num-gpus 1
Please visit https://docs.nvidia.com/clara/#parabricks for detailed documentation



[Parabricks Options Mesg]: Checking argument compatibility
[Parabricks Options Mesg]: Automatically generating ID prefix
[Parabricks Options Mesg]: Read group created for /sc/arion/scratch/choh07/pbtest/Data/sample_1.fq.gz and
/sc/arion/scratch/choh07/pbtest/Data/sample_2.fq.gz
[Parabricks Options Mesg]: @RG\tID:HK3TJBCX2.1\tLB:lib1\tPL:bar\tSM:sample\tPU:HK3TJBCX2.1
[PB Info 2024-Nov-04 22:28:03] ------------------------------------------------------------------------------
[PB Info 2024-Nov-04 22:28:03] ||                Parabricks accelerated Genomics Pipeline                ||
[PB Info 2024-Nov-04 22:28:03] ||                            Version 4.3.0-1                             ||
[PB Info 2024-Nov-04 22:28:03] ||                     GPU-BWA mem, Sorting Phase-I                       ||
[PB Info 2024-Nov-04 22:28:03] ------------------------------------------------------------------------------
[M::bwa_idx_load_from_disk] read 0 ALT contigs
[PB Info 2024-Nov-04 22:28:06] GPU-BWA mem
[PB Info 2024-Nov-04 22:28:06] ProgressMeter    Reads           Base Pairs Aligned
[PB Info 2024-Nov-04 22:28:32] 5043564          580000000
[PB Info 2024-Nov-04 22:28:50] 10087128 1160000000
[PB Info 2024-Nov-04 22:29:08] 15130692 1740000000
[PB Info 2024-Nov-04 22:29:24] 20174256 2320000000
[PB Info 2024-Nov-04 22:29:42] 25217820 2900000000
[PB Info 2024-Nov-04 22:29:59] 30261384 3480000000
[PB Info 2024-Nov-04 22:30:17] 35304948 4060000000
[PB Info 2024-Nov-04 22:30:33] 40348512 4640000000
[PB Info 2024-Nov-04 22:30:50] 45392076 5220000000
[PB Info 2024-Nov-04 22:31:08] 50435640 5800000000
[PB Info 2024-Nov-04 22:31:25]
GPU-BWA Mem time: 199.060446 seconds
[PB Info 2024-Nov-04 22:31:25] GPU-BWA Mem is finished.

[main] CMD: /usr/local/parabricks/binaries/bin/bwa mem -Z ./pbOpts.txt -F 0 /sc/arion/scratch/choh07/pbtest/Ref/Homo_sapiens_assembly38.fasta /sc/arion/scratch/cho
h07/pbtest/Data/sample_1.fq.gz /sc/arion/scratch/choh07/pbtest/Data/sample_2.fq.gz @RG\tID:HK3TJBCX2.1\tLB:lib1\tPL:bar\tSM:sample\tPU:HK3TJBCX2.1
[main] Real time: 202.232 sec; CPU: 1558.157 sec
[PB Info 2024-Nov-04 22:31:25] ------------------------------------------------------------------------------
[PB Info 2024-Nov-04 22:31:25] ||        Program:                    GPU-BWA mem, Sorting Phase-I        ||
[PB Info 2024-Nov-04 22:31:25] ||        Version:                                    4.3.0-1             ||
[PB Info 2024-Nov-04 22:31:25] ||        Start Time:                     Mon Nov  4 22:28:03 2024        ||
[PB Info 2024-Nov-04 22:31:25] ||        End Time:                       Mon Nov  4 22:31:25 2024        ||
[PB Info 2024-Nov-04 22:31:25] ||        Total Time:                         3 minutes 22 seconds        ||
```

Running Parabricks (fq2bam)

Run Parabricks on Minerva

```
singularity exec --nv --bind $WORK_DIR:$WORK_DIR \
    $PARABRICKS pbrun fq2bam \
    --ref $WORK_DIR/${REFERENCE_FILE} \
    --in-fq $WORK_DIR/${INPUT_FASTQ_1} $WORK_DIR/${INPUT_FASTQ_2} \
    --knownSites $WORK_DIR/${KNOWN_SITES_FILE} \
    --out-bam $WORK_DIR/${OUTPUT_BAM} \
    --out-recal-file $WORK_DIR/${OUTPUT_RECAL_FILE}
```

Compatible CPU-based BWA-MEM, GATK4 Commands

```
# Run bwa-mem and pipe the output to create a sorted BAM.
$ bwa mem \
    -t 32 \
    -K 10000000 \
    -R '@RG\tID:sample_rg1\tLB:lib1\tPL:bar\tSM:sample\tPU:sample_rg1' \
    <INPUT_DIR>/${REFERENCE_FILE} <INPUT_DIR>/${INPUT_FASTQ_1} <INPUT_DIR>/${INPUT_FASTQ_2} | \
  gatk SortSam \
    --java-options -Xmx30g \
    --MAX_RECORDS_IN_RAM 5000000 \
    -I /dev/stdin \
    -O cpu.bam \
    --SORT_ORDER coordinate

# Mark duplicates.
$ gatk MarkDuplicates \
    --java-options -Xmx30g \
    -I cpu.bam \
    -O mark_dups_cpu.bam \
    -M metrics.txt

# Generate a BQSR report.
$ gatk BaseRecalibrator \
    --java-options -Xmx30g \
    --input mark_dups_cpu.bam \
    --output <OUTPUT_DIR>/${OUTPUT_RECAL_FILE} \
    --known-sites <INPUT_DIR>/${KNOWN_SITES_FILE} \
    --reference <INPUT_DIR>/${REFERENCE_FILE}
```

Running Parabricks (rna_fq2bam)

Run Parabricks on Minerva

```
singularity exec --nv --bind $WORK_DIR:$WORK_DIR \
    $PARABRICKS pbrun rna_fq2bam \
    --in-fq $WORK_DIR/${INPUT_FASTQ_1} $WORK_DIR/${INPUT_FASTQ_2} \
    --genome-lib-dir $WORK_DIR/${PATH_TO_GENOME_LIBRARY}/ \
    --output-dir $WORK_DIR/${PATH_TO_OUTPUT_DIRECTORY} \
    --ref $WORK_DIR/${REFERENCE_FILE} \
    --out-bam $WORK_DIR/${OUTPUT_BAM} \
    --read-files-command zcat
```

Compatible CPU-based STAR, GATK4 Commands

```
# STAR Alignment
$ ./STAR \
      --genomeDir <INPUT_DIR>/${PATH_TO_GENOME_LIBRARY} \
      --readFilesIn <INPUT_DIR>/${INPUT_FASTQ_1} <INPUT_DIR>/${INPUT_FASTQ_2} \
      --outFileNamePrefix <OUTPUT_DIR>/${PATH_TO_OUTPUT_DIRECTORY}/ \
      --outSAMtype BAM SortedByCoordinate \
      --readFilesCommand zcat

# Mark Duplicates
$ gatk MarkDuplicates \
    --java-options -Xmx30g \
    -I Aligned.sortedByCoord.out.bam \# This filename is determined by STAR.
    -O <OUTPUT_DIR>/${NAME_OF_OUTPUT_BAM_FILE} \
    -M metrics.txt
```

Running Parabricks (haplotypecaller)

Run Parabricks on Minerva

```
singularity exec --nv --bind $WORK_DIR:$WORK_DIR \
    $PARABRICKS pbrun haplotypecaller \
    --ref $WORK_DIR/${REFERENCE_FILE} \
    --in-bam $WORK_DIR/${INPUT_BAM} \
    --in-recal-file $WORK_DIR/${INPUT_RECAL_FILE} \
    --out-variants $WORKDIR/${OUTPUT_VCF}
```

Compatible CPU-based GATK4 Commands

```
# Run ApplyBQSR Step
$ gatk ApplyBQSR \
    --java-options -Xmx30g \
    -R Ref/Homo_sapiens_assembly38.fasta \
    -I mark_dups_cpu.bam \
    --bqsr-recal-file recal_file.txt \
    -O cpu_nodups_BQSR.bam

#Run Haplotype Caller
$ gatk HaplotypeCaller \
    --java-options -Xmx30g \
    --input cpu_nodups_BQSR.bam \
    --output result_cpu.vcf \
    --reference Ref/Homo_sapiens_assembly38.fasta \
    --native-pair-hmm-threads 16
```

Batch Job Submission Example (fq2bam)

```
[choh07@li03c02 pbtest]$ cat fq2bam.lsf
#!/bin/bash
#BSUB -J fq2bam                 # Job name
#BSUB -P acc_hpcstaff           # allocation account
#BSUB -q gpuexpress             # queue
#BSUB -n 64                     # number of compute cores
#BSUB -W 10                     # walltime in HH:MM
#BSUB -R h10080g
#BSUB -gpu num=4
#BSUB -R rusage[mem=4000]       # 256 GB of memory (4 GB/core * 64 cores)
#BSUB -R span[hosts=1]          # all cores from the same node
#BSUB -o %J.stdout              # output log (%J : JobID)
#BSUB -eo %J.stderr             # error log
#BSUB -L /bin/bash              # Initialize the execution environment


ml parabricks

# Set Input Parameters
PARABRICKS=/hpc/packages/minerva-centos7/parabricks/4.3.0-1/clara-parabricks_4.3.0-1.sif
DATA_DIR=/sc/arion/scratch/choh07/pbtest
INP_fq1=$DATA_DIR/Data/sample_1.fq.gz
INP_fq2=$DATA_DIR/Data/sample_2.fq.gz
REF=$DATA_DIR/Ref/Homo_sapiens_assembly38.fasta
OUT_BAM=$DATA_DIR/output.bam


# Run FQ2BAM (FASTA + FASTQ ==> BAM)
singularity exec \
    --nv --bind $DATA_DIR:$DATA_DIR $PARABRICKS \
    pbrun fq2bam --ref $REF --in-fq $INP_fq1 $INP_fq2 --out-bam $OUT_BAM --num-gpus 4
[choh07@li03c02 pbtest]$
[choh07@li03c02 pbtest]$
[choh07@li03c02 pbtest]$ bsub < fq2bam.lsf
Job <144140780> is submitted to queue <gpuexpress>.
[choh07@li03c02 pbtest]$
```

Batch Job Submission Example (deepvariant)

```
[choh07@li03c02 pbtest]$ cat deepvariant.lsf
#!/bin/bash
#BSUB -J deepvariant            # Job name
#BSUB -P acc_hpcstaff           # allocation account
#BSUB -q gpuexpress             # queue
#BSUB -n 64                     # number of compute cores
#BSUB -W 10                     # walltime in HH:MM
#BSUB -R h10080g
#BSUB -gpu num=4
#BSUB -R rusage[mem=4000]       # 256 GB of memory (4 GB/core * 64 cores)
#BSUB -R span[hosts=1]          # all cores from the same node
#BSUB -o %J.stdout              # output log (%J : JobID)
#BSUB -eo %J.stderr             # error log
#BSUB -L /bin/bash              # Initialize the execution environment

ml parabricks

# Set Input Parameters
PARABRICKS=/hpc/packages/minerva-centos7/parabricks/4.3.0-1/clara-parabricks_4.3.0-1.sif
DATA_DIR=/sc/arion/scratch/choh07/pbtest
INP_fq1=$DATA_DIR/Data/sample_1.fq.gz
INP_fq2=$DATA_DIR/Data/sample_2.fq.gz
REF=$DATA_DIR/Ref/Homo_sapiens_assembly38.fasta
IN_BAM=$DATA_DIR/output.bam
OUT_VARIANTS=$DATA_DIR/haplotype.vcf

# Run DeepVariant (BAM ==> VCF)
singularity exec \
    --nv --bind $DATA_DIR:$DATA_DIR $PARABRICKS \
    pbrun deepvariant \
        --ref $REF \
        --in-bam $IN_BAM \
        --out-variants $OUT_VARIANTS \
        --num-cpu-threads-per-stream 6 \
        --num-streams-per-gpu 2 \
        --num-gpus 4
```

Monitoring Resource Usage

```
[choh07@li03c02 pbtest]$ bjobs -gpu -l 144140780

Job <144140780>, Job Name <fq2bam>, User <choh07>, Project <acc_hpcstaff>, Appl
                     ication <default>, Status <DONE>, Queue <gpuexpress>, Job
                     Priority <50>, Command <#!/bin/bash;#BSUB -J fq2bam
```

⋮

```
Mon Nov  4 23:44:46: Done successfully. The CPU time used is 2052.0 seconds.
                     HOST: lg05g29; CPU_TIME: 2052 seconds
                        GPU ID: 0
                            Total Execution Time: 132 seconds
                            Energy Consumed: 8459 Joules
                            SM Utilization (%): Avg 15, Max 93, Min 0
                            Memory Utilization (%): Avg 1, Max 9, Min 0
                            Max GPU Memory Used: 17448304640 bytes

                        GPU ID: 1
                            Total Execution Time: 132 seconds
                            Energy Consumed: 8017 Joules
                            SM Utilization (%): Avg 14, Max 94, Min 0
                            Memory Utilization (%): Avg 1, Max 9, Min 0
                            Max GPU Memory Used: 17448304640 bytes

                        GPU ID: 2
                            Total Execution Time: 132 seconds
                            Energy Consumed: 8185 Joules
                            SM Utilization (%): Avg 13, Max 96, Min 0
                            Memory Utilization (%): Avg 1, Max 9, Min 0
                            Max GPU Memory Used: 17448304640 bytes

                        GPU ID: 3
                            Total Execution Time: 132 seconds
                            Energy Consumed: 9335 Joules
                            SM Utilization (%): Avg 12, Max 94, Min 0
                            Memory Utilization (%): Avg 1, Max 9, Min 0
                            Max GPU Memory Used: 17448304640 bytes


GPU Energy Consumed: 33996.000000 Joules

 RUNLIMIT
 10.0 min

 MEMLIMIT
    3.9 G

 MEMORY USAGE:
 MAX MEM: 36.6 Gbytes;  AVG MEM: 12.4 Gbytes; MEM Efficiency: 14.66%

 CPU USAGE:
 CPU PEAK: 21.63 ;  CPU PEAK DURATION: 60 second(s)
 CPU AVERAGE EFFICIENCY: 20.52% ;  CPU PEAK EFFICIENCY: 33.80%
```

Acknowledgements

- Supported by the Clinical and Translational Science Awards (CTSA) grant UL1TR004419 from the National Center for Advancing Translational Sciences, National Institutes of Health.

Last but not Least

Got a problem? Need a program installed? Send an email to:

hpchelp@hpc.mssm.edu