Welcome to
Introduction to Accelerated Genomic Analysis
March 12 at 3:00-4:00 PM

UPCOMING EVENT

NVIDIA Global Technology Conference (GTC)
March 17-21, 2024
Free to Attend Virtually
https://www.nvidia.com/gtc

Accelerated Compute and AI in Genomics
Introduction to GPU-Accelerated Genomics with Parabricks
Training DeepVariant Models using Parabricks
BioGPT-Powered Synonym Classification for Biomedical Entities
GPU-Accelerated Gap-Affine DNA Pairwise Alignment
Accelerating AI-Enabled De-Novo Molecule Generation using GPU and In-Vitro Validation
Everything, All at Once: Processing Spatial Transcriptomics Data using Accelerated Computing
...
Welcome to
Introduction to Accelerated Genomic Analysis
March 12 at 3:00-4:00 PM

UPCOMING EVENT

High-Performance Python for GPUs

Tuesday, March 19 at 4:30-6:00 PM
Henry Schreiner, Computational Physicist and Lecturer, PICSciE

https://researchcomputing.princeton.edu/workshops
Welcome to

Introduction to Accelerated Genomic Analysis
March 12 at 3:00-4:00 PM

GPU User Group

Next meeting is Friday, March 22 at 3:00-4:00 PM
Lewis Library 245
Write to rcinfo@princeton.edu to join

https://researchcomputing.princeton.edu/learn/user-groups
Welcome to
Introduction to Accelerated Genomic Analysis
March 12 at 3:00-4:00 PM

Princeton Open (GPU) Hackathon

Does your research group have a code that would make a much greater scientific impact if it were accelerated? Teams of 3-6 members work with two expert mentors to speed-up the code. Free and open to Princeton researchers.

Application Deadline: March 26
Hackathon: June 4, 12-14
Introduction to Accelerated Genomics Analysis

Huiwen Ju, PhD
Solutions Architect, Higher Education & Research, NVIDIA
hju@nvidia.com
Princeton University, 3/12/2024
NVIDIA in Healthcare & Life Sciences

NVIDIA in Genomics

NVIDIA Parabricks for Accelerated Secondary Analysis

Getting Started with Parabricks

RAPIDS and CuPy for Accelerated Bioinformatics
NVIDIA in Healthcare & Life Sciences
NVIDIA Clara for Healthcare and Life Sciences

World’s Largest Data Industry | 36% CAGR by 2025

NVIDIA CLARA APPLICATION FRAMEWORKS

MEDICAL DEVICES
GENOMICS
DRUG DISCOVERY
SMART HOSPITALS

HOLOSCAN
MONAI
FLARE
PARABRICKS
BIONEMO

NVIDIA AI
NVIDIA Omniverse
NVIDIA in Genomics
Genomics Projects Will Exceed 40 Exabytes in the Next Decade

“Our ability to sequence DNA has far outpaced our ability to decipher the information it contains, so genomic data science will be a vibrant field of research for many years to come.”

National Human Genome Research Institute

Source: https://www.genome.gov/about-genomics/fact-sheets/Genomic-Data-Science
Accelerated Genomic Workflows Saving Lives, Costs and Time

Precision Medicine Evolution - What is important about gleaning insights quickly?

Cancer

NICU + Critical Care

Population Studies

Research

- a variant calling identifies cancer mutation profiles in the 1000 Genomes Project

- TRACT

- Section of de novo variants (DNVs) is critical for studies of disease-related variation and clinical outcomes. We developed a GPU-based workflow to rapidly call DNVs (HAT) and demonstrated its effectiveness by applying it to the Simons Simplex Collection (SSC) whole-exome sequenced parent-child trios from DNA derived from blood. In our SSC DNA data, we identified 79 ± 15 DNVs per individual, 10% ± 4% at CpG sites, 75% ± 9% phased to the paternal chromosome of origin, and an average allele balance of 0.49. These calculations are in line with DNV expectations. We sought to build a control DNV database by running HAT on whole-exome sequenced parent-child trios from DNA derived from lymphoblastoid cell lines (LCLs) from the publicly available 1000 Genomes Project (1000G). In our 1000G DNA data, we identified 240 ± 66 DNVs per individual, 14% ± 4% at CpG sites, 91% ± 11% phased to the paternal chromosome of origin, and an average allele balance of 0.41. Of the 66210 DNVs, 74 had >100 DNVs and we hypothesized that excess DNVs were cell line artifacts. Some of our evidence in our data suggest that this is true and that 1000G does not appear to be the reference. By mutation profile analysis, we tested whether these cell line artifacts were idem and found that 40% of individuals in 1000G did not have random DNV profiles; rather they had DNV profiles matching B-cell lymphoma. Furthermore, we saw significant excess non-coding DNVs in 1000G in the gene KLRD1 that has already been implicated in this line. As a result of cell line artifacts, 1000G has variants present in DNA-repair genes we
Genomics Technologies
NVIDIA Delivers Solutions Across Genomics

Short-Read

Long-Read

Single Cell

Spatial
NVIDIA is Helping Across the Computational Genomics Workflow
From sequencing sensor to biological insights
How NVIDIA is Solving Big Genomics Data

Hardware and Software Solutions

**Onboard Sequencers**
Addressing velocity with real-time analysis solutions

**Data Center Scale**
Addressing volume with scalable solutions in data centers and cloud

**Full Stack Solutions**
Addressing variety with dedicated software and AI models
NVIDIA GPUs for Instrument Secondary Analysis
The H100 Dynamic Programming Core

Dynamic Programming
Exponential to polynomial time problem solving

Supercharged Smith-Waterman
35x acceleration over CPU
7x acceleration over Ampere
NVIDIA Parabricks for Accelerated Secondary Analysis
NVIDIA Parabricks for Alignment & Variant Calling

**Speed, Scale, Accuracy**

- **Universal Analysis**
  Industry-standard tools for all major sequencers, ported to GPU

- **Up to 100x Acceleration**
  Up to 100x faster for WGS compared to CPU-only

- **Up to 50% Lower Cost**
  Up to 50% lower compute cost for WGS compared to CPU-only

- **Higher Accuracy with AI**
  The power of deep learning for customized high accuracy analysis
Key Applications of NVIDIA Parabricks
Accelerated and Deep Learning Genomic Analysis

Population Genomics
Cancer Genomics
RNA Sequencing

Gold Standard Processing and Quality Control
- Sort BAM
- Mark duplicates
- BQSR
- BAM CRAM
- Collect multiple metrics
- BAM2FQ

High-Accuracy Variant Calling
- DeepVariant
- Mutect2
- GenotypeGVCF
- HaplotypeCaller
- STAR-Fusion
- IndexGVCF

FastQ

BWA-MEM
STAR
Higher Speed
From hours to minutes

End-to-end germline sample analysis with industry-standard tools in under 15 mins on the new NVIDIA H100 GPUs

Germline GATK (FQ2BAM + HaplotypeCaller)
14 mins

Germline DeepVariant (FQ2BAM + DeepVariant)
14 mins

~18 hrs

Runtimes on CPU (m5.24xlarge)

Runtimes on NVIDIA GPU (8xH100)

~30 hrs
Up to 80x Acceleration
Gold-standard results, faster

Runtimes on CPU

Runtimes on NVIDIA GPU

<table>
<thead>
<tr>
<th>Task</th>
<th>CPU Runtimes</th>
<th>NVIDIA GPU Runtimes</th>
</tr>
</thead>
<tbody>
<tr>
<td>BWA-MEM</td>
<td>11 mins</td>
<td>~4 hrs</td>
</tr>
<tr>
<td>Sort, MarkDups,</td>
<td>6 mins</td>
<td>~9 hrs</td>
</tr>
<tr>
<td>Apply BQSR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HaplotypeCaller</td>
<td>4 mins</td>
<td>~16 hrs</td>
</tr>
<tr>
<td>DeepVariant</td>
<td>11 mins</td>
<td>~5 hrs</td>
</tr>
<tr>
<td>Mutect2</td>
<td>45 mins</td>
<td>~31 hrs</td>
</tr>
</tbody>
</table>

v3.8 Benchmarks
Dataset: HG002 30x WGS, except Mutect2 on SEQC2 50x WGS
CPU: m5.24xlarge; GPU: 8xA100, except DeepVariant & Mutect2 on 8xV100
NVIDIA’s Clara Parabricks workflows in Terra bring GPU acceleration to genomic analysis

Geraldine Van der Auwera  
September 20, 2022

The past few years have seen a massive surge in the development of advanced analytical methods for biomedical research, fueled in part by technological innovations that allow computational scientists to crunch data at ever-increasing speed and scale. A growing number of technology companies have joined the effort to help researchers tackle emerging challenges, ranging from large-scale genomics to multi-modal analysis of the myriad data types associated with medical records — including doctors’ notes, which are famously easy to read and interpret.

Today NVIDIA, a pioneer in AI and accelerated computing, announced a new partnership with the Broad Institute that will pool the two organizations’ respective expertise in deep learning, accelerated compute, and biomedical research. This partnership builds on an existing collaboration between NVIDIA and the Broad’s GATK team, who have already been working together to improve some of the deep learning algorithms in GATK. (Keep an eye on the GATK blog for an upcoming release announcement.)

The NVIDIA team released a Clara Parabricks workspace in Terra that makes their GPU-accelerated genomic analysis toolkit available on the cloud at the click of a button. As

Run Parabricks with bioinformatics workflow managers that deploy directly on cloud

Where lower runtime converts to lower cost

Runtime per genome

- 2 hrs

Cost per genome

- $2

~24 hrs

Runtimes on CPU (N2 64vCPU)

Runtimes on NVIDIA GPU (4xT4)
Parabrics on DNAnexus
Accelerated and Scalable Analysis

Seamlessly run Parabrics apps directly on DNAnexus

Regeneron analyzed exomes 11X faster at 40% of the cost with Parabrics DeepVariant on DNAnexus

<table>
<thead>
<tr>
<th>Runtime per exome</th>
<th>Cost per exome</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 mins</td>
<td>$0.0976</td>
</tr>
<tr>
<td>58 mins</td>
<td>$0.265</td>
</tr>
</tbody>
</table>

Runtimes on CPU (m5d.4xlarge)
Runtimes on NVIDIA GPU (4xT4)
Parabricks on AWS HealthOmics
Ready2Run Workflows

Seamlessly run 13 NVIDIA Parabricks Ready2Run workflows spanning 5x, 30x and 50x germline workflows for DeepVariant and HaplotypeCaller as well as a 50x somatic workflow.

Runtimes on CPU instance (m5.24xlarge)

<table>
<thead>
<tr>
<th>Workflow</th>
<th>Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germline GATK (FQ2BAM + HaplotypeCaller)</td>
<td>75 mins</td>
</tr>
<tr>
<td>Germline DeepVariant (FQ2BAM + DeepVariant)</td>
<td>120 mins</td>
</tr>
<tr>
<td>Somatic workflow</td>
<td>18 hrs</td>
</tr>
</tbody>
</table>

(*includes instance provisioning & data load time)
MULTI-TOOL IMPLEMENTATION PROVIDES THE BEST RESULTS

Comparing germline calling between GATK Haplotype and Google’s DeepVariant

<table>
<thead>
<tr>
<th></th>
<th>False Positives</th>
<th>False Negatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>HaplotypeCaller only</td>
<td>20,305</td>
<td>9,321</td>
</tr>
<tr>
<td>DeepVariant only</td>
<td>5,101</td>
<td>5,374</td>
</tr>
<tr>
<td>Union</td>
<td>22,906</td>
<td>3,893</td>
</tr>
<tr>
<td>Intersect</td>
<td>2,500</td>
<td>10,802</td>
</tr>
</tbody>
</table>

HG002 Target genome, analyzing only high confidence regions
Modular Tools for Flexible Deployment

With NVIDIA AI Enterprise for Production

- Individual Containers for Each Tool
- Agile Releases as and when required
- Lean Deployment on Sequencers or as part of custom workflows

Top of Tree Individual Containers
Available on NGC

Stable Unified Container
Available on NGC
Enterprise Support Available

- Includes every Parabricks tool
- Regular Releases timed with NVIDIA AI Enterprise
- Option of Purchasing Enterprise Support thru NVIDIA AI Enterprise

All NVIDIA Parabricks Containers are available publicly in the NGC Catalog
## Now Free or NVIDIA Enterprise Support

<table>
<thead>
<tr>
<th>Feature</th>
<th>Do it Yourself</th>
<th>NVIDIA AI Enterprise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accelerated tools and high throughput workflows</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Community Forum support</strong></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Submit questions to the <a href="#">Developers Forum</a> for community support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support for many NVIDIA tools, including Parabricks, RAPIDS, TensorRT and more</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Long-term support – Up to 3 years</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Security reviews and notifications</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>API stability and compatibility across releases</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Optimized and tested for production deployment in multiple environments</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Single source of enterprise support with SLAs</td>
<td>X</td>
<td>✓</td>
</tr>
</tbody>
</table>
Universal Sequencing Analysis
The only hardware-accelerated solution for multiple sequencers

Accelerated DeepVariant models, for up to 75x faster variant calling

Reference WDL/NextFlow workflows for multiple sequencers

- **Illumina sequencing**
  - Alignment
  - Small Variant Calling
    - BWA-MEM
    - DeepVariant
      - Illumina model

- **Oxford Nanopore sequencing**
  - Basecalling + Integrated Alignment
    - Dorado
    - Minimap2
  - Small Variant Calling
    - DeepVariant
      - ONT model

- **PacBio sequencing**
  - Alignment
  - Small Variant Calling
    - Minimap2
    - DeepVariant
      - PacBio model

<table>
<thead>
<tr>
<th>Sequence Type</th>
<th>Time</th>
<th>GPU Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Illumina data (30x)</td>
<td>~5 hrs</td>
<td>NVIDIA GPUs</td>
</tr>
<tr>
<td>PacBio data (30x)</td>
<td>~5 hrs</td>
<td>NVIDIA GPUs</td>
</tr>
<tr>
<td>Oxford Nanopore data (55x)</td>
<td>~8 hrs</td>
<td>NVIDIA GPUs</td>
</tr>
</tbody>
</table>

- CPU-only (m5.24xlarge)
- NVIDIA GPUs (DGX A100)
Parabricks DeepVariant Training Framework
Easily Train a Custom Model for Optimal Accuracy

Training
Custom deep learning models optimized for very specific data

For re-training on different sequencers/data types

For finetuning to individual labs protocols
Parabricks is fully compatible with common workflow managers WDL and NextFlow for deploying at scale

- Intertwine GPU and CPU powered tasks with different compute requirements
- Reference workflows and recommended compute configs at: github.com/clara-parabricks-workflows
Real-world Use Cases for Parabricks
Making a Real-World Impact
NVIDIA Parabricks for sequencing centers, national programs, and groundbreaking research

Human Genome Center of Tokyo
Deploying rapid analysis of human genomes with NVIDIA Parabricks and 80 NVIDIA V100 GPUs in their SHIROKANE compute cluster

Regeneron & UKBioBank
Regeneron Genetics Center has sequenced the exomes of all 450,000 participants of UKBioBank, processing the vast data with Clara Parabricks

Stanford University World Record
World record for fastest DNA sequencing, set in collaboration with NVIDIA, using NVIDIA Parabricks’ accelerated DeepVariant
UNIVERSITY OF TOKYO ACCELERATES GENOMIC ANALYSIS BY 40X ON JAPAN’S SHIROKANE

Challenge

Human Genome Center (HGC) at University of Tokyo needed to analyze massive amounts of genomic data quickly and accurately.

Goal is to glean insights about germline and somatic variants and create a genome database of 92,000 patients that will be utilized by research institutions, pharmaceutical companies, and university hospitals.

Solution

HGC leveraged NVIDIA Clara Parabricks’ GPU-accelerated genomic analysis software and NVIDIA DGX A100 to power SHIROKANE, with 2 petaFLOPS of performance, for accurate and fast genomic analysis.

The solution accelerated genomic analysis by 40X compared to a CPU-based environment.

SHIROKANE is Japan’s #1 supercomputer for life sciences and is accessible to many universities and research institutions.

“The whole-genome data analysis capability is equivalent to hundreds of conventional CPU servers and was implemented on the GPU server. We will realize a state-of-the-art high-speed whole-genome data analysis environment that greatly accelerates genome research for SHIROKANE users.”

— Professor Seiya Imoto, Director of HGC, University of Tokyo
Large Population Study
UK Biobank's 470,000+ exomes analyzed by Regeneron as of July 2022

- Regeneron Genetics Center Sequencing close to 500,000 Exomes
- Exomes were analyzed in 5 minutes with NVIDIA Parabricks versus 1 hour in CPUs
- The cost went down 60% on GPUs
- DeepVariant optimized for RGC outputs, especially alignment.
- Analysis of DNA given to researchers on UK Biobank which span academia, pharma and other scientists interested in genomic variants for diseases
As Sequencing Gets Faster, Informatics is the Bottleneck

Wet Lab
4 hrs 48 mins

Informatics
4 hrs 46 mins

Basecalling &
Alignment

Small & Structural
Variant Calling &
Annotation

Filtering,
Prioritization &
Curation

World Record for Fastest DNA Sequencing Technique for Critical Care (2022)
7 hrs 18 mins
Stanford University, Oxford Nanopore, Google & NVIDIA
ULTRARAPID NANOPORE ANALYSIS PIPELINE — UNAP

End-to-End Acceleration of Nanopore WGS

- UNAP runs on a single DGX A100 and containerized on NGC for easy deployment
- High accuracy, AI whole genome variant calling with nanopore data
- Reduce compute cost from $568 to $183 per patient
- Achieves real-time AI basecalling and alignment

Available Now
nvidia.com/unap

Guinness World Record
(60x WGS, patient 11)

NVIDIA Optimized
(60x WGS, HG002)

Basecalling & Alignment
64 GPUs
2hr37m

Multiple Variant Calling
56 GPUs
1hr46m

DGX A100
4hr10m
THE IMPACT OF AN AT-RICH GENOME ON ANALYSIS TIME
Malaria Project with Assistant Prof. Giovanna Carpi, Purdue University

<table>
<thead>
<tr>
<th></th>
<th>Human</th>
<th>Malaria</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Chromosome</strong></td>
<td>46</td>
<td>14</td>
</tr>
<tr>
<td><strong>Size</strong></td>
<td>6,200 megabases</td>
<td>23 megabases</td>
</tr>
<tr>
<td><strong>GC content</strong></td>
<td>41%</td>
<td>19%</td>
</tr>
<tr>
<td><strong>WGS CPU analysis time</strong></td>
<td>~20-30 hours</td>
<td>~6 hours</td>
</tr>
</tbody>
</table>

Source: [https://www.sanger.ac.uk/resources/downloads/protozoa/plasmodium-falciparum.html](https://www.sanger.ac.uk/resources/downloads/protozoa/plasmodium-falciparum.html)
Tychele Turner • 1st
Assistant Professor at Washington University in St. Louis

Running Google DeepVariant through new NVIDIA Parabricks 4.1 on brand new PacBio Revio data from a cell line we are characterizing in @TNTurnerLab on our Lambda Workstation with 2 NVIDIA RTX 6000 Ada GPUs is blazing fast.

Here are the run time statistics:
SMRT cell 1 (30x genome): 1422 seconds
SMRT cell 2 (32x genome): 1542 seconds
SMRT cell 3 (36x genome): 1740 seconds
Combined Revio data (98x genome): 4585 seconds

#genomics #genetics #gpu #longRead #letsgo

Recording available on NVIDIA On Demand - Accelerating Gene Variant Detection With Deep Learning
Initial results from The Francis Crick Institute show their end-to-end analysis of human whole genomes can be done in just over two hours with NVIDIA Parabricks, compared to approximately 13 hours on their Nemo CPU cluster.

For the TRACERx EVO project alone, it is estimated this will save nearly 9 years of bioinformatics processing time - an improvement described as a “game-changer in terms of the feasibility of the analysis pipelines for the project” by Mark S. Hill, Principal Research Fellow, TRACERx EVO.
Blogs for NVIDIA PARABRICKS 4.1

- NVIDIA PB 4.1 blog with Regeneron & Agilent exomes
- Terra PB 4.1 blog - PB workspace & highlight Singular
- Analyze Ancient DNA like from the Woolly Mammoth [NVIDIA blog highlighting Parabricks use on ancient DNA]

Use Parabricks for Human, Animal, Pathogen, Plant, ...!
Getting Started with Parabricks
Where Can I Run NVIDIA Parabricks?

Public Cloud Providers

Platforms & Partners

On-Premise

All Parabricks containers are available publicly in the NGC Catalog
Parabricks reference workflows (WDL & NextFlow) are available on the public Parabricks Workflows GitHub
Running NVIDIA Parabricks
Requirements

Hardware Requirements

- Any NVIDIA GPU that supports CUDA architecture 70, 75, 80, 86, 89 or 90 and has at least 16GB of GPU RAM. Clara Parabricks has been tested on the following NVIDIA GPUs:
  - V100
  - T4
  - A10, A30, A40, A100, A6000
  - L4, L40
  - H100

- The `fg2bam` tool requires at least 24 GB of GPU memory by default; the `--low-memory` option will reduce this to 16 GB of GPU memory at the cost of slower processing. All other tools require at least 16 GB of GPU memory per GPU.

- System Requirements:
  - A 2 GPU system should have at least 100GB CPU RAM and at least 24 GPU threads.
  - A 4 GPU system should have at least 196GB CPU RAM and at least 32 GPU threads.
  - A 8 GPU system should have at least 392GB CPU RAM and at least 48 GPU threads.

Software Requirements

- An NVIDIA driver with version 526.60.13 or greater.
- Any Linux Operating System that supports `nvidia-docker2` Docker version 20.10 (or higher)

NVIDIA Parabricks documentation v4.2.1
Running NVIDIA Parabricks

Checking requirements

Checking available NVIDIA hardware and driver

To check your NVIDIA hardware and driver version, use the `nvidia-smi` command:

```
$ nvidia-smi
```

This shows the following important information:

- The NVIDIA driver version is 525.60.3.
- The supported CUDA driver API is 12.0.
- The GPU has 16 GB of memory.

Checking available CPU RAM and threads

To see how much RAM and CPU threads in your machine, you can run the following:

```
# To check available memory
$ cat /proc/meminfo | grep MemTotal
```
```
# To check available number of threads
$ cat /proc/cpuinfo | grep processor | wc -l
```

Checking nvidia-docker2 installation

To make sure you have nvidia-docker2 installed, run this command:

```
$ docker run --rm --gpus all nvidia/cuda:12.0.0-base-ubuntu20.04 nvidia-smi
```

When it finishes downloading the container, it will run the `nvidia-smi` command and show you the same output as above.

Checking python version

To see which version of Python you have, enter the following command:

```
$ python3 --version
```

Make sure it's at least version 3 (3.6.9, 3.7, etc).
Running NVIDIA Parabricks
Using the command line

Getting the Software

The Clara Parabricks Docker image can be obtained by running the following command:

```
$ docker pull nvcr.io/nvidia/clara/clara-parabricks:4.1.1-1
```

At this point the software is ready to use.

Running Clara Parabricks

From the Command Line

Clara Parabricks is deployed using a Docker image. There are two parts to customizing a Parabricks run:

› Customizing Docker container specific options: These are the options that are passed to the `docker` command before the name of the container. For example, the user should mount their data directories within the Docker container by passing the `--volume` option to Docker. See the Tutorials for more detailed examples.

› Parabricks specific options: These options are passed to the Parabricks command line to customize the Parabricks run. For example, you can choose which tool to run and pass tool-specific options.

For example, to run the Parabricks `fq2bam (FQ2BAM + BWA-MEM)` tool using a Docker container, use the following command:

```
$ docker run \
   --gpus all \
   --rm \
   --volume /host/data:/input_data \
   --volume /host/results:/outputdir \
   --workdir /input_data \
   nvcr.io/nvidia/clara/clara-parabricks:4.1.1-1 \
   pbrun fq2bam \
   --ref /input_data/Ref/Homo_sapiens_assembly38.fasta \
   --in-fq /input_data/Data/sample_1.fq.gz /input_data/Data/sample_2.fq.gz \
   --out-bam /outputdir/fq2bam_output.bam
```

Sample data is freely available. See the Getting The Sample Data section in the Tutorials for instructions on obtaining the sample data, and a step-by-step guide to using both `fq2bam` and Haplotype Caller.
Running NVIDIA Parabricks
Tutorials & drop-in command replacements

The tutorials walk you through a simple use case for Clara Parabricks, giving a brief introduction of how it works. You will start by downloading some sample data:

- A reference file (`Homo_sapiens_assembly38.fasta`) and its index
- A 'known indels' file and its index
- Two FASTQ files
- Associated index files

The tutorials then walk through the following steps:

- Alignment (FASTA + FASTQ => BAM)
- Variant calling (BAM => VCF)

The tutorials are meant to be simple and straightforward and to only cover a single, specific use case. You should be able to copy and paste the commands into a terminal window and get the same results as shown. The How-Tos cover more general problem solving using Clara Parabricks.

Steps in the Tutorial

- Getting The Sample Data
- FQ2BAM Tutorial
- HaplotypeCaller Tutorial

Quick Start

This command assumes all the inputs are in INPUT_DIR and all the outputs go to OUTPUT_DIR.

docker run --rm -gpus all --volume INPUT_DIR:/workdir --volume OUTPUT_DIR:/outputdir \
   -v /workdir:/workdir \
   nvidia/claraparabricks:4.1.1-1 \
   pbrun autocaller \
   --ref /workdir/$REFERENCE_FILE \
   --tumor-name tumor \
   --in-tumor-bam /workdir/$INPUT_TUMOR_BAM \
   --in-normal-bam /workdir/$INPUT_NORMAL_BAM \
   --normal-name normal \
   --out-vcf /outputdir/$OUTPUT_VCF

Compatible GATK4 Command

The command below is the GATK4 counterpart of the Clara Parabricks command above. The output from this command will be identical to the output from the above command. See the Output Comparison page for comparing the results.

```
gatk Mutect2 \
   -R $REFERENCE_FILE \
   -I $INPUT_DIR/$INPUT_TUMOR_BAM \
   -tumor-sample tumor \
   -I $INPUT_DIR/$INPUT_NORMAL_BAM \
   -n $normal-sample normal \
   -O $OUTPUT_DIR/$OUTPUT_VCF
```
In this section we present several step-by-step guides to running Clara Parabricks on several cloud platforms.

- Running Clara Parabricks on AWS
- Running Clara Parabricks on DNAnexus
- Running Clara Parabricks on GCP
- Running Clara Parabricks on Terra


Starting a Compute Instance

In this section, we will show how to start a Compute Instance on GCP.

Begin by navigating to the Google Cloud homepage and selecting Compute Engine from the left sidebar. This will take us to the VM instances page.

At the top of the page, select Create Instance. Here we can configure all the settings for our VM instance. Under Name let’s call the instance “parabricks” and select an appropriate region. For the purpose of this guide, the region can be anything.
Getting Started with NVIDIA Parabricks

NVIDIA Parabricks Landing Page

NVIDIA Parabricks Product Sheet & Solution Briefs

Download NVIDIA Parabricks

Experience an Exome Workflow on LaunchPad
Third-party Applications Accelerated on NVIDIA Platforms

Accelerated Apps Catalog
Collaboration Between NVIDIA and The Broad Institute

NVScoreVariants

- NVIDIA are collaborating with The Broad Institute to bring more deep learning to genomics and GATK

- NVScoreVariants is a rebuild of the tool CNNScoreVariants in PyTorch
  
  - Enabling DL filtering of HaplotypeCaller output with CNNs
  
  - And opening the tool to the PyTorch community for further developments

RAPIDS and CuPy for Accelerated Bioinformatics
GPU DATA SCIENCE

💡 ACCELERATED DATA SCIENCE
The RAPIDS suite of open source software libraries gives you the freedom to execute end-to-end data science and analytics pipelines entirely on GPUs.
Learn about RAPIDS ➤

🔍 SCALE OUT ON GPUs
Seamlessly scale from GPU workstations to multi-GPU servers and multi-node clusters with Dask.
Learn about Dask ➤

🎨 PYTHON INTEGRATION
Accelerate your Python data science toolchain with minimal code changes and no new tools to learn.
Learn about our libraries ➤

🌟 TOP MODEL ACCURACY
Increase machine learning model accuracy by iterating on models faster and deploying them more frequently.
Learn about RAPIDS for model optimization ➤

⏰ REDUCED TRAINING TIME
Drastically improve your productivity with more interactive data science tools like XGBoost.
Learn about XGBoost ➤
Learn about accelerated ML with cuML ➤

🔗 OPEN SOURCE
RAPIDS is an open source project. Supported by NVIDIA, it also relies on Numba, Apache Arrow, and many more open source projects.
Learn about our projects ➤
# DATA SCIENCE TOOLSETS

<table>
<thead>
<tr>
<th></th>
<th>CPU</th>
<th>GPU/RAPIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data handling</td>
<td>pandas</td>
<td>cuDF</td>
</tr>
<tr>
<td>Machine learning</td>
<td>scikit-learn</td>
<td>cuML</td>
</tr>
<tr>
<td>Graph analytics</td>
<td>NetworkX</td>
<td>cuGraph</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>CPU</th>
<th>GPU/RAPIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viz</td>
<td>Bokeh/</td>
<td>cuXfilter</td>
</tr>
<tr>
<td></td>
<td>Datashader</td>
<td></td>
</tr>
<tr>
<td>Geospatial</td>
<td>GeoPandas/</td>
<td>cuSpatial</td>
</tr>
<tr>
<td></td>
<td>SciPy.spatial</td>
<td></td>
</tr>
<tr>
<td>Signals</td>
<td>SciPy.signal</td>
<td>cuSignal</td>
</tr>
<tr>
<td>Cyber</td>
<td>cyberpandas</td>
<td>CLX</td>
</tr>
</tbody>
</table>
RAPIDS PLATFORM

Dask

- cuDF: Data Prep/Handling
- cuML: Machine Learning
- cuGraph: Graph Analytics
- DL Frameworks: Deep Learning
- cuXfilter: Visualization

GPU Memory

Apache Arrow

Specialized package examples

- cuSpatial: Geospatial Analytics
- cuSignal: Signal Processing
- CLX: Cyber Analytics
GPU ANALYSIS OF 1 MILLION CELLS
From 3.5 hours to 8 minutes

<table>
<thead>
<tr>
<th>CPU Runtime</th>
<th>GPU runtime</th>
<th>GPU acceleration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n1-highmem-32</td>
<td>a2-highgpu-1g</td>
</tr>
<tr>
<td></td>
<td>32 vCPUs</td>
<td>Tesla A100 40GB GPU</td>
</tr>
<tr>
<td>Preprocessing</td>
<td>28m35s</td>
<td>3m21s</td>
</tr>
<tr>
<td>PCA</td>
<td>29.2s</td>
<td>11.4s</td>
</tr>
<tr>
<td>t-SNE</td>
<td>1hr23m10s</td>
<td>28s</td>
</tr>
<tr>
<td>KNN</td>
<td>3m5s</td>
<td>46s</td>
</tr>
<tr>
<td>UMAP</td>
<td>21m47s</td>
<td>13.4s</td>
</tr>
<tr>
<td>k-means</td>
<td>2m6s</td>
<td>1.9s</td>
</tr>
<tr>
<td>clustering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Louvain</td>
<td>15m5s</td>
<td>1.9s</td>
</tr>
<tr>
<td>clustering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Leiden</td>
<td>51m1s</td>
<td>1.4s</td>
</tr>
<tr>
<td>clustering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End-to-end runtime</td>
<td>3hr31m48s</td>
<td>8m22s</td>
</tr>
<tr>
<td>End-to-end cost</td>
<td>$6.682</td>
<td>$0.553</td>
</tr>
</tbody>
</table>

Repository for example jupyter notebooks: [https://github.com/NVIDIA-Genomics-Research/rapids-single-cell-examples](https://github.com/NVIDIA-Genomics-Research/rapids-single-cell-examples)
RAPIDS-SingleCell
Part of the scverse ecosystem

- A new library drawing inspiration from the rapids-single-cell-examples library and the ScanPy library

- Introduces GPU-optimized versions of the ScanPy (single cell) and SquidPy (spatial) functions

- The library’s primary objective is to blend the computational strength of GPUs with the user-friendly nature of the scverse ecosystem.
RAPIDS-SingleCell
The AnnData framework now supports CuPy arrays

Rapids-singlecell utilizes the scverse AnnData data framework, which supports dense and sparse CuPy arrays

https://anndata.readthedocs.io/en/latest/#
# RAPIDS-SingleCell

API based on ScanPy and SquidPy

<table>
<thead>
<tr>
<th>API</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scanpy-GPU</td>
<td><code>rapids_singlecell.pp.calculate_qc_metrics</code></td>
<td>Calculates basic qc Parameters.</td>
</tr>
<tr>
<td></td>
<td><code>rapids_singlecell.pp.filter_cells</code></td>
<td>Filter cell outliers based on counts and numbers of genes expressed.</td>
</tr>
<tr>
<td></td>
<td><code>rapids_singlecell.pp.filter_genes</code></td>
<td>Filter genes based on number of cells or counts.</td>
</tr>
<tr>
<td></td>
<td><code>rapids_singlecell.pp.normalize_total</code></td>
<td>Normalizes rows in matrix so they sum to <code>target_sum</code>.</td>
</tr>
<tr>
<td></td>
<td><code>rapids_singlecell.pp.log1p</code></td>
<td>Calculated the natural logarithm of one plus the sparse matrix.</td>
</tr>
<tr>
<td></td>
<td><code>rapids_singlecell.pp.highly_variable_genes</code></td>
<td>Annotate highly variable genes.</td>
</tr>
<tr>
<td></td>
<td><code>rapids_singlecell.pp.regress_out</code></td>
<td>Use linear regression to adjust for the effects of unwanted noise and variation.</td>
</tr>
<tr>
<td></td>
<td><code>rapids_singlecell.pp.scale</code></td>
<td>Scales matrix to unit variance and clips values</td>
</tr>
<tr>
<td></td>
<td><code>rapids_singlecell.pp.flag_gene_family</code></td>
<td>Performs PCA using the cuml decomposition function.</td>
</tr>
<tr>
<td></td>
<td><code>rapids_singlecell.pp.filter_highly_variable</code></td>
<td>Applies analytic Pearson residual normalization,</td>
</tr>
</tbody>
</table>

1 Million Brain Cells

Author: Severin Dicks

To run this notebook please make sure you have a working environment with all necessary dependencies. Run the data_downloader notebook first to create the AnnData object we are working with. In this example workflow we’ll be looking at a dataset of 100000 brain cells from Nvidia.

```python
import scanpy as sc
import anndata
import cupy as cp

import time
import rapids_singlecell as rsc
from rapids_singlecell.cunnData import cunnData

import warnings
warnings.filterwarnings("ignore")

import rmm
from rmm.allocators.cupy import rmm_cupy_ALLOCATOR
rmm.reinitialize(
    managed_memory=False, # Allows oversubscription
    pool_allocator=False, # default is False
    devices=0, # GPU device IDs to register. By default registers only GPU 0.
)
```

CuPy: drop-in replacement for Numpy
NUMERICAL COMPUTING IN PYTHON

- Mathematical focus
- Operates on arrays of data
  - `ndarray`, holds data of same type
- Many years of development
- Highly tuned for CPUs

- NumPy like interface
- Trivially port code to GPU
- Copy data to GPU
  - CuPy `ndarray`
- Data interoperability with DL frameworks, RAPIDS, and Numba
- Uses high tuned NVIDIA libraries
- Can write custom CUDA functions
CUPY
A NumPy like interface to GPU-acceleration ND-Array operations

**BEFORE**

```python
import numpy as np
size = 4096
A = np.random.randn(size, size)
Q, R = np.linalg.qr(A)
```

**AFTER**

```python
import cupy as cp
size = 4096
A = cp.random.randn(size, size)
Q, R = cp.linalg.qr(A)
```

52x Speedup!
What is NUMBA? When do we use it?

Lower-level CUDA kernel development without leaving Python

<table>
<thead>
<tr>
<th>Just-in-time compiler</th>
<th>Opt-in</th>
<th>PyData ecosystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numba is a JIT compiler for Python functions that you specify. Numba targets both CPU and GPU.</td>
<td>Numba only compiles functions you specify. You don’t need to compile the full program</td>
<td>While not all functions in python can be compiled with Numba, the PyData ecosystem is well covered.</td>
</tr>
</tbody>
</table>

Numba provides the Python programmer a simple way to write customizable GPU accelerated code without needing CUDA C/C++

NVIDIA DLI online course: Fundamentals of Accelerated Computing with CUDA Python
Genomics Recommended Sessions to Attend

FEATURED TALKS

Kimberly Powell Special Address: Generative AI is Accelerating Healthcare into One of the Largest Technology Industries [S62604]
Kimberly Powell | General Manager and VP, Healthcare and Life Sciences
Tuesday, Mar 19 | 8:00 AM PDT

The Role of Generative AI in Modern Medicine [S62777]
Kimberly Powell | General Manager and VP, Healthcare and Life Sciences
Eric Topol | Professor and Executive Vice President
Catherine D. Wood | Chief Executive Officer/Chief Investment Officer
Peter Lee | Corporate Vice President of Research and Incubations
Tuesday, Mar 19 | 11:00 AM - 11:50 AM PDT

How Artificial Intelligence is Powering the Future of Biomedicine [S62283]
Priscilla Chan | Co-Founder and co-CEO, Chan Zuckerberg Initiative
Mona Flores | Global Head of Medical AI, NVIDIA
Tuesday, Mar 19 | 10:00 AM - 10:25 AM PDT

First-Ever Whole Transcriptome Imaging of Tissues using CosMx-SMI: Highest-Density Dataset Ever Collected [S61995]
Joseph Beechem | Senior Vice President of Research and Development, Nanostring
Wednesday, Mar 20 | 208:00 AM - 8:25 AM PDT

Nucleotide Transformer: Advancing Genomic Analysis with Large Language Models [S62438]
Karim Beguir | CEO and Co-founder, InstaDeep
Tuesday, Mar 19 | 9:00 AM - 9:25 AM PDT

Computer Vision for Rare Disease Genomic Medicine [S62535]
Wolfgang Pernice | Assistant Professor of Neurological Sciences, Columbia University In The City Of New York
Thursday, Mar 21 | 9:00 AM - 9:25 AM PDT

Introduction to GPU-Accelerated Genomics with Parabricks [S62322]
Harry Clifford | Genomics Product Lead, NVIDIA
Wednesday, Mar 20 | 4:30 PM - 4:55 PM PDT

WORKSHOPS & TRAININGS

Training DeepVariant Models using Parabricks [DLIT61813]
Thursday, Mar 21 | 2:00 PM - 3:40 PM PDT

Everything, All at Once: Processing Spatial Transcriptomics Data using Accelerated Computing [DLIT61337]
Tuesday, Mar 19 | 10:00 AM - 11:40 AM PDT