A Comprehensive Guide to Deploying IntOGen on Seqera

Jiajun Zhan

Collaborative Centre for Genomic Cancer Medicine

jiajun.zhan.1@.unimelb.edu.au

Contents:

1. Introduction

1.1 Objective

The primary goal of this project is to deploy the IntOGen pipeline on the Seqera platform, leveraging its AWS Batch-based architecture and S3 bucket storage environment. This deployment aims to make the pipeline more accessible and convenient for researchers by providing a scalable, efficient, and user-friendly solution for executing IntOGen. By utilizing AWS S3, data storage and retrieval become seamless, ensuring that the pipeline can handle large-scale genomic datasets with high availability and reliability.

1.2 Background

What is intOGen?

IntOGen is a framework for automatic and comprehensive knowledge extraction based on mutational data from sequenced tumor samples from patients. The framework identifies cancer genes and pinpoints their putative mechanism of action across tumor types.

How does intOGen work?

Given a dataset of somatic point mutations from a cohort of tumor samples, intOGen first preprocesses the input mutations, next it runs seven different methods for cancer driver gene identification and, finally, it combines the output of these methods to produce a compendium of driver genes and a repository of the mutational features that can be used to explain their mechanisms of action.

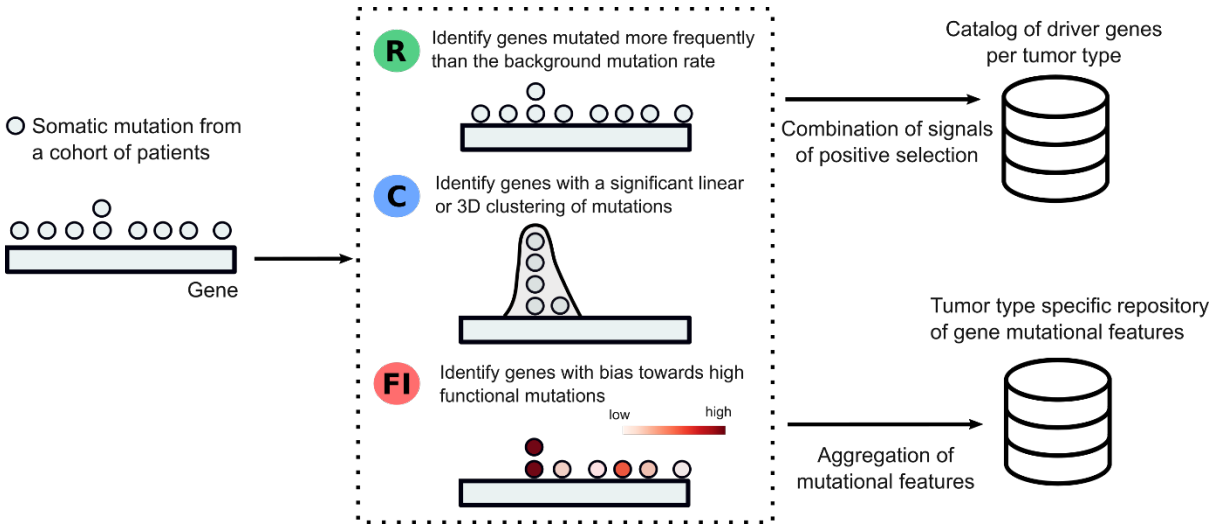

Figure1. Schema summarizing the intOGen pipeline

What is Seqera?

The Australian Nextflow Seqera Service provides access to a centralised web platform for managing, launching and monitoring the execution of Nextflow bioinformatics pipelines on Seqera-compatible compute environments. Australian research organisations and individual researchers can add compute infrastructure they have access to and publish pipelines to private or shared workspaces to run data analyses at scale. The service provides access to all features available through the Seqera Platform including data exploration and analysis as well as notebook analysis.

Australian BioCommons (whose lead agent is the University of Melbourne) operates the Australian Nextflow Seqera Service in collaboration with Pawsey Supercomputing Research Centre, National Computational Infrastructure (NCI), and Seqera. The Service was established as an output of the Australian BioCommons Bring Your Own Data Expansion Project and is hosted on Amazon Web Services (AWS). It is supported by Bioplatforms Australia via NCRIS funding.

How does Seqera work?

Seqera works by providing a cloud-native platform for executing computational workflows, designed to simplify and streamline the deployment of pipelines on cloud infrastructure. It builds on Nextflow, a workflow orchestration tool, and leverages AWS Batch and S3 storage to provide scalability, reliability, and ease of use.

1.3 Environment

The deployment of the IntOGen pipeline on Seqera is carried out using the following tools, compute environments, and storage resources:

Pipeline Version

- **IntOGen Version**: v2024, a robust pipeline designed for cancer driver gene identification.
- **Pipeline Repository:** Intogen_pipeline_2024
    - Revision: main
- **Documentation**: Refer to the official IntOGen documentation for detailed information on features, usage, and requirements.

Docker Container

- **Container Storage**: GitHub Packages
    - The IntOGen pipeline exclusively uses Docker containers stored in this repository to ensure compatibility with AWS Batch, as AWS Batch does not support Singularity. This adaptation ensures seamless execution in the AWS cloud environment.

Platform

- **Seqera Platform**: Seqera, version 2024, a cloud-native workflow orchestration platform powered by AWS Batch.

Compute Environment

- **AWS Batch Compute Environment**: aws_batch_spot_ap-southeast-2
    - Region: Asia Pacific (Sydney).
    - Uses AWS Spot instances for cost-effective scalability.

Workspace

- **AWS S3 Bucket for Workspace**:
    - s3://org.umccr.nf-tower.tower/intogen-plus-2024/

Data Storage

- **AWS S3 Bucket for Data Storage**:
    - s3://org.umccr.nf-tower.general/intogen-plus-2024/
    - This bucket is used for storing input datasets, reference datasets, and final results.

2. Prerequisites

2.1 System and Environment Requirements

Compute Resource Needs

The IntOGen pipeline requires specific computational resources tailored to its tasks:

- **CPU**: Multi-core processors for parallel execution (at least 4cores).
- **Memory**: Minimum of 32 GB, with more recommended for large genomic datasets.
- **Storage**:
  - Temporary storage for intermediate files.
  - Persistent storage in AWS S3 buckets for input data, reference datasets, and outputs (>160 GB).
- **Details**: Resource configurations for each task are detailed in the local.conf file.

Software dependencies

The following software is required to deploy and run the IntOGen pipeline on Seqera:

- **Nextflow**: Workflow orchestration tool.
  - **Version**: NXF_VER=22.10.6 (ensure this specific version is installed for compatibility with the pipeline).
  - Installation guide: Nextflow Documentation.
- **Singularity:** Used for local container builds before deploying on AWS Batch.
  - **Version:** 2.x (compatible with the IntOGen pipeline's requirements).
  - Installation guide: Singularity v2.x Documentation.
- **Docker**: Used to execute tasks within containers, ensuring reproducibility and compatibility with AWS Batch.
  - Singularity is not supported by AWS Batch.
- **Java**: Required by Nextflow.
  - Recommended version: Java 11.
- **AWS CLI**: For managing S3 buckets.
- **IntOGen Pipeline Building Dependencies**:
  - Pre-installed bioinformatics tools and scripts within the Docker container (refer to the container's GitHub Packages).

3. IntOGen Pipeline building locally

3.1 Official steps:

```
wget https://github.com/bbglab/intogen-plus/archive/refs/heads/master.zip
unzip master.zip
cd intogen-plus-master
cd build
conda env create -p / intogen-plus-master/build/intogen -f env.yml
conda activate /intogen-plus-2024/build/intogen
make
```

The following environment variable +must* be defined:

- COSMIC_KEY: it is used to access and retrieve the CGC dataset from COSMIC. It can be defined as: export COSMIC_KEY=$(echo "<email>:<password>" | base64)

Currently there are several environment variables that can be defined:

- INTOGEN_DATASETS: path where to store the datasets. Default ../datasets.
- INTOGEN_CONTAINERS: path where to store the containers. Default ../containers.
- ensembl: specifies the ensembl version
- cadd: specifies the CADD version (used for OncodriveFML)
- cores: amount of cores to use by processes that can be parallelized

Important notes

The jar file with the latest version of MutPanning needs to be manually downloaded from http://www.cancer-genes.org/ and placed in containers/mutpanning.

The scores used by OncodriveFML are build querying directly the CADD scores from https://cadd.gs.washington.edu/download The process can be significantly faster and less error prone if you download it first and replace the CADD_URL variable in datasets/oncodriverfml/fml.mk with the full path where you have downloaded the CADD scores.

Note:

- ✧ Finish these changes below before make.
- ✧ Build time may take 20 to 40 hours, depending on computer performance and Internet speed
- ✧ If the building failed, try to use *sudo make*
- ✧ Some datasets link issues (404), refer to https://github.com/bbglab/intogen-plus/issues/34

3.2 Dndscv fixing:

Containers:

```
cd /build/containers/dndscv/
nano dndscv.R
# Edit to this (add cv parameter):
    covs <- as.matrix(readRDS(file.path(Sys.getenv("INTOGEN_DATASETS"), "dndscv",
"covariates.rds")))
    result <- dndscv(muts, refdb=file.path(Sys.getenv("INTOGEN_DATASETS"),
"dndscv", "RefCDS.rda"), cv = covs)
nano Singularity
# Edit to
From: debian:bullseye-slim
# Add to the last line

echo "if (!requireNamespace('BiocManager', quietly = TRUE))

install.packages('BiocManager', repos='https://cloud.r-project.org');

BiocManager::install('GenomicRanges')" | R --no-save
```

Datasets:

In datasets/dndscv folder,
replace original RefCDS.rda to
https://github.com/im3sanger/dndscv_data/blob/master/data/RefCDS_human_GRCh38_GencodeV18_recommended.rda
Rename to RefCDS.rda
Download
https://github.com/im3sanger/dndscv_data/blob/master/data/covariates_hg19_hg38_epigenome_pcawg.rda
Rename to covariates.rds

3.3 Hotmaps fixing:

This change is for adapt to aws batch environment, and it's also work locally.

```
cd /build/containers/hotmaps/
nano Singularity
# Edit to
From: debian:buster-slim
%post
    apt-get update && apt-get install -y --no-install-recommends \
        python2 python2-dev python-pip python-setuptools \
        libcurl4-openssl-dev libssl-dev zlib1g-dev make libncurses5-dev g++ bash
procps curl
    pip install --no-cache-dir --upgrade pip setuptools
    pip install --no-cache-dir numpy==1.16.5
```

```
    pip install --no-cache-dir pandas==0.24.2 tqdm futures bgreference
biopython==1.76 pyliftover pycurl
```

---

4. Deployment on Seqera

4.1 Converting to Docker image

AWS Batch environment only supports Docker image, so we need to convert singularity images to Docker images and push to some hub (e. g Docker hub, github packages) as you like. So that AWS batch can correctly load these images.

Example steps:

```
singularity build --sandbox hotmaps_sandbox /home/kairos/intogen-plus-
2024/containers/hotmaps.simg
cd hotmaps_sandbox/
tar -cvf docker_image.tar *
docker import docker_image.tar hotmaps:new
docker tag hotmaps:new ghcr.io/jiajzhan-kairos/hotmaps:fixed
docker push ghcr.io/jiajzhan-kairos/hotmaps:fixed
```

4.2 Uploading the Pipeline and Configuration

- Upload the main pipeline scripts to github repositorie.
- Upload the dataset and to AWS S3 bucket.
- Upload the containers to github packages
- Ensuring compatibility with Seqera.

4.3 Input, Output and pipeline Configuration

- Edit the nextflow.nf to adapt aws batch environment:
    - Main strategy is add extra step to download datasets from AWS S3 bucket to tmp folder, let every task can access like local file:

```
process DownloadDatasets {
    tag "Download datasets"
    label "core"

    output:
    path "./*" into REFERENCE_FILES

    script:
```

```
    """
    mkdir -p ./datasets/
    aws s3 cp s3://org.umccr.nf-tower.general/intogen-plus-
2024/datasets/ ./datasets/ --recursive
    mkdir -p ./config/
    aws s3 cp s3://org.umccr.nf-tower.general/intogen-plus-
2024/config/annotations.txt ./config/
    """
    }
```

Every task if needs to access datasets, need to point out input:

```
path referenceFiles from REFERENCE_FILES
```

Unless the task will not recognize the reference dataset paths

In the same time, you need to edit nextflow.config, to align the path of the datasets and containers.

Example

- Managing output results:
    - You also need to change the output path definition: (file() function will transfer S3 path to local path, let it unable to upload output to S3 path that we want):

```
OUTPUT = file(params.output)
STEPS_FOLDER = file(params.stepsFolder)
```

To

```
OUTPUT = params.output
STEPS_FOLDER = params.stepsFolder
```

- Memory call policy:
    - For each tasks, we need to customize the memory call policy, otherwise, the task will stop for lacking of memory. Example

---

5. Execution and Monitoring

5.1 Starting the Workflow

- How to launch the workflow, refer to **IntOGen_guidance** document.

5.2 Monitoring the Workflow

- Using Seqera's interface or logs to track progress.

---

6. Performance Optimization (To do)

6.1 Resource Allocation

- Adjusting resource allocation settings for each tasks.

6.2 Caching and Reuse

- Strategies for caching to minimize redundant computations and download steps.

---

7. Testing and Validation

7.1 Initial Validation

- Verifying the pipeline with a test dataset.

7.2 Results Verification

- Comparing results for accuracy.

7.3 Performance Testing

- Analyzing runtime and resource usage.

---

8. Summary and Reflection

8.1 Challenges faced during deployment

1. Due to AWS Batch's compute strategy being task submission-based rather than interactive, and the working directory being located in an AWS S3 bucket with the environment configured in containers, debugging errors can be very time-consuming and cumbersome. When an error is identified, it requires waiting for the AWS S3 task submission to run to completion, downloading the error logs from AWS S3, and then reviewing them to identify the cause. While errors in pipeline scripts can be directly modified in the GitHub repository, issues related to the environment require rebuilding the container after making changes, pushing it to the container hub, and resubmitting the task for testing. Each debugging cycle incurs high time and operational costs, with a single modification test potentially taking one to two hours.
2. The IntOGen Pipeline was originally designed for local and cluster computing environments and lacks adaptation for cloud computing environments and S3 storage

buckets. It requires restructuring the pipeline to accommodate cloud computing environments.

3. Since the IntOGen pipeline was initially released in 2013, some datasets and software versions have not been updated in a timely manner, resulting in a significant amount of time spent searching for compatible datasets.
4. When rebuilding the container, the outdated versions of some packages caused multiple issues with base images being too new in their repository versions. This led to spending considerable time testing and selecting a suitable base image.

8.2 Evaluation of the integration of IntOGen and Seqera

The integration of IntOGen with Seqera represents an important extension of genomic analysis workflows. It not only modernizes the pipeline but also ensures scalability, reproducibility, efficiency, and ease of use, making it a valuable tool for the research community.

---

9. Appendix

9.1 Sample Configuration Files

- Examples of nextflow.config and other critical files.

9.2 References

- Links to official documentation.
- Relevant community forums or technical resources.