Clinical NAGIM

Variant Counts Project

Version 0.3

Australian Genomics

Overview

This report describes a federated "variant counts" system for computing how often germline variants have been observed across all participating Australian pathology laboratories - for the purposes of aiding interpretation of unclassified variants. The primary expected use case is to assist with deprioritising variants that are unlikely to explain a person's condition.

This use case is one of several user stories that will be explored in the Australian Genomics Clinical NAGIM project 2024-25.

Version History

Version	Description of Changes
0.3	Incorporated AG feedback
0.2	Added solution design, incorporated lab feedback
0.1	Initial Google Doc shared with labs

User story

As a pathology laboratory, I want to be able to access counts of how often germline variants have been observed across all participating Australian pathology labs - for the purposes of deprioritising variants that occur more often within the Australian population than would be expected from global population databases. I need to be able to do this in bulk (not via a network API call) as these counts will be used with our laboratory in automated tooling for variant prioritisation - and hence will need to be accessed millions of times per sample at speed. I expect to submit counts of germline variation from my laboratory to contribute data to the Australia-wide count.

Input

Counts of variants split by zygosity (homozygous and heterozygous) computed across my lab.

e.g. (proposed data as submitted from one lab)

contig --- str	position --- u64	ref --- str	alt --- str	hom_count --- u32	het_count --- u32
NC_000015.9	20000041	T	A	0	1
NC_000015.9	20000075	C	T	1	0
NC_000015.10	20000114	G	T	0	1
NC_000015.10	20000123	T	C	2	3
NC_000015.10	20087835	T	A	0	1
NC_000015.10	20408548	CTT	CT	1	3
NC_000015.10	20408548	CTT	CTTT	0	4
NC_000015.10	20408548	CTT	C	0	4
NC_000015.10	20495041	G	GC	4	1
NC_000015.10	20495041	G	C	0	5

Output

Counts of variants split by zygosity (homozygous and heterozygous) summed across all participating labs. Columns with the number of labs that contributed non-zero values to each variant count will also be added.

e.g. (proposed return data to all labs)[1]

contig --- str	position --- u64	ref --- str	alt --- str	hom_count --- u32	het_count --- u32	hom_lab_num --- u8	het_lab_num --- u8
NC_000015.9	20000041	T	A	4	9	3	3
NC_000015.9	20000075	C	T	10	1	3	3
NC_000015.10	20000114	G	T	0	5	0	2
NC_000015.10	20000123	T	C	2	3	1	3
NC_000015.10	20087835	T	A	1	1	2	3
NC_000015.10	20408548	CTT	CT	1	3	2	3
NC_000015.10	20408548	CTT	CTTT	0	4	0	1
NC_000015.10	20408548	CTT	C	0	4	0	2
NC_000015.10	20495041	G	GC	4	1	2	2
NC_000015.10	20495041	G	C	0	7	0	3

[1] Whilst this is displayed as a simple table, it may also be possible to transform this data to formats compatible with existing tooling (e.g. a population VCF) as part of the central aggregation service.

Definitions

Variants

Throughout this document we will refer to variants being stored – but we need to define exactly what variants should be included if we are not to needlessly store variants that add no clinical value. Ideally the system will store variants that are not commonly encountered in a global population but noting also that identifying the differing variant frequencies between an "Australian rare disease cohort" and global population databases is in part the purpose of this system. Therefore, we need to establish a reasonably safe threshold above which the global frequency means that the variant is unlikely to be clinically different specifically within the Australian rare-disease context.[2]

We propose setting the threshold to exclude all variants that occur with AF > 5% in the gnomAD 4.1 joint sites data.[3]

At the other end of the spectrum, it would be possibly to exclude variants that are already known to be pathogenic or likely pathogenic - and therefore not really needing to play a part in further variant prioritisation. However, from feedback, it was determined that there was no point in excluding these variants. Whilst these variants will have limited clinical use as a variant count in a specific Australian frequency database, there is little storage/compute downsides to including them anyway.

Open questions

- Do we need to consider "founders" that are at a higher population % when considering the threshold to use - see https://clinicalgenome.org/site/assets/files/3460/ba1_exception_list_07_30_2018.pdf.

Variant identifier

The primary key of the data we are going to sum needs to be some sort of string identifier of the variant.

Currently proposing using `contig`, `pos`, `ref`, `alt` essentially passed through directly from input VCF files (with a simple map from chromosome to contig) - though this essentially is punting on concerns about normalisation. The use of `contig` rather than `chromosome` allows the database to hold both 37 and 38 data as entirely separate counts and avoids ambiguity about the prefixes of chromosomes in VCF files.

[2] Stakeholder feedback included "I would think that setting a conservative threshold (maybe 5%) for inclusion would maximise utility while reducing the storage/compute." and "BA1, which by default in ACMG 2025 gets you to Benign, is applied over 5%. I think nowadays, 1% or lower can be used for gene-specific rules where the max pop frequency is calculated."
[3] e.g. https://gnomad-public-us-east-1.s3.amazonaws.com/release/4.1/vcf/joint/gnomad.joint.v4.1.sites.chr1.vcf.bgz

Open questions

- Is there a better identifier to use?
    - VRS (GA4GH)
    - SPDI (NCBI)
    - gnomAD (chromosome-position-reference-alternate)
- Should we agree upon a normalisation system so that variant identifiers are more likely to be consistent between labs (e.g. VOCA)?
- Could lift-over achieve a merging of the 37 and 38 tables and would this be desirable?

Threat analysis

Someone with a large list of the variants of an individual could look up enough of these variants to establish the presence of a similar individual in the aggregate data of Australia pathology laboratories - thereby abstractly confirming the *existence* of the individual. This threat however only allows an attacker to replay back information which they already possess (an individual's variants). They do not gain any non-variant information, and they cannot use the aggregate data to extend their knowledge of variants of a single individual.

Where non-variant information might be included (see extensions), a common mitigation technique for preventing the information leakage would be to set a minimum count below which totals would only be reported as "under threshold". This could be used as a mitigation technique by the centralised counting service when sending aggregates back to labs. This mitigation however needs to be balanced against the use of this data for "rare" variation - and suppressing low count data may work against the underlying reasons for the system.

There is no suggestion that the count information is to be made available outside the participating labs, so it is possible there is a mitigation via governance of the data (contractual agreement to not distribute or attempt re-identification).

Solution design

A solution is proposed that uses AWS native sharing mechanisms to allow a federation of laboratories to participate in this rare disease counts user story. A central system (in a designated centrally managed AWS account) will read laboratory counts and then immediately relay back an Australia-wide count computed from every laboratory it has access to.

The central system will not retain any count information i.e. labs do not need to leave their data in a central system – they can withdraw from the system at any point merely by removing bucket permissions. Whilst engaged in the system however, the central system account will need to be trusted enough that it can perform the limited read and write operations needed to calculate aggregates.

The solution breaks down into two fundamental activities

- Internal lab activity to create variant count files across their lab population. There are a variety of techniques for doing this, dependent on how labs store variants internally. A separate document/repo will be prepared for helping labs with this process.
- A central system to read/collate these variant count files and write back aggregate count files.

As the variant count files are the key interface point for sharing, they will need to be defined in an agreed upon format, schema and location.

Format

Parquet is an open-source table structure file format that forms the basis for many high performance analytics tooling. Parquet can be read natively (and at speed) by a variety of tooling that may support lab needs, from `Apache Spark` to `polars` to `DuckDB`. Parquet internally compresses data so there is no need to also zip objects.

Parquet objects have contained schema definitions, which means the system can evolve relatively safely (it can handle detecting count files from two labs where one is “normal” and the other is “extended” for instance).

Obviously, the size of the objects will to some extent depend on the number of variants a laboratory has seen, which in turns depends on the number of rare disease cases sequenced – but from representative testing the objects are expected to be tens to hundreds of megabytes.

Schema

Presented is an example schema definition, shown using Polars (a Python data science library) – but this is similar to the schema definition that will be used by any Parquet library.

```
s = pl.Schema({
    "contig": pl.String,
    "position": pl.UInt64,
    "ref": pl.String,
    "alt": pl.String,
    "hom_count": pl.UInt32,
    "het_count": pl.UInt32,
})
```

Location

Each lab will provide a named S3 bucket with read/write permissions granted to the central account. The bucket keys will be divided by two prefixes – one is a "readable" folder that provides count data of the lab. The other is a "writeable" folder in which resulting aggregate "Australia-wide" count data is written.

Currently the two prefixes are:

`lab-counts/` (data created by lab i.e. readable by the central account)

`aggregate-counts/` (data returned to the lab i.e. writeable by the central account)

Variant count data may be computed by each lab at differing times and with different frequencies – therefore we propose an ISO 8601 dated folder structure where it is up to each lab to choose where/when to place the data. The central system will cope with sourcing the data from the "latest" date available from each lab.

For example (central system would choose to source data from 2024-07-15):

`lab-counts /2024-07-01/`

`lab-counts /2024-07-15/`

Within each of these dated folders must be the complete variant count data for a laboratory as of that date. The data can be specified in any number of parquet files with the suffix `.parquet`.

For example:

`lab-counts /2024-07-15/chr1.parquet`

`lab-counts /2024-07-15/chr2.parquet`

…

`lab-counts /2024-07-15/chr23.parquet`

Data will be returned into the `aggregate/` folder prefix in a variety of formats (still under design). One will be a parquet file with summed counts and written to this fixed location.

`aggregate-counts/latest.parquet`

The buckets should be “Bucket owner enforced” and with “SSE-S3” encryption settings. All other S3 bucket settings are up to the owning laboratory.

See appendix A for an example S3 resource policy.

Extensions

Data extension

The user story could be extended to include other breakdowns of interesting variables, with each lab opting in to this extended data as it becomes available to them.

For instance, further splitting variant zygosity by sex, would refine the homozygous column to also include two other sub-columns (male, female). In Parquet format, an example of `{4,{3,1}}` - meaning 4 homozygous samples consisting of 3 males and 1 female.

e.g. (proposed input with sex extended data)

contig --- str	position --- u64	ref --- str	alt --- str	hom_count --- struct[2]	het_count --- struct[2]
NC_000015.9	20000041	T	A	{0,{0,0}}	1,{0,1}}
NC_000015.9	20000075	C	T	{1,{1,0}}	0,{0,0}}
NC_000015.10	20000114	G	T	{0,{0,0}}	1,{0,1}}
NC_000015.10	20000123	T	C	{2,{2,0}}	3,{2,1}}
NC_000015.10	20087835	T	A	{0,{0,0}}	1,{1,0}}
NC_000015.10	20408548	CTT	CT	{1,{0,1}}	3,{0,3}}
NC_000015.10	20408548	CTT	CTTT	{0,{0,0}}	4,{2,2}}
NC_000015.10	20408548	CTT	C	{0,{0,0}}	4,{2,2}}
NC_000015.10	20495041	G	GC	{4,{3,1}}	1,{0,1}}
NC_000015.10	20495041	G	C	{0,{0,0}}	5,{0,5}}

Extended data types could include:

- Sex
- Broad phenotypes (HPO top-level terms)
- Broad ethnicity (super populations)
- Assay type

Open questions

- Is there a high priority extension data (sex?) that should be included in initial implementations?
- Should individual labs (i.e. not all at the same time) be able to submit extension data and if whilst this is done by a subset of labs, what should be returned?

Frequency extension

Change the user story “count” to a “frequency”. A column with the population count over which this particular variant was able to be observed will also be added - allowing frequencies to be calculated.

This obviously has the problem that distinguishing between a site that was not called and a site that was outside coverage requires information external to the input VCFs.

Also, the population number (which is needed to compute accurate frequencies) reveals the number of samples sequenced and some aspects of the assay type (exome v genome). This could reveal information that some labs consider to be commercial in confidence.

Variant type extension

Somatic variation would need a different schema and hence would not use identical mechanisms for sharing (noting that this is a rare disease project and this is probably not important).

Structural variation is not really supported in the schema.

Appendix A

Example bucket policy. The bucket name will be selected by the participating lab, and the central account number will be provided as part of the solution roll out.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowObjectListingFromCentral",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<central account>:root"
            },
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::<bucket name>"
        },
        {
            "Sid": "AllowLabObjectReadingFromCentral",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<central account>:root"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::<bucket name>/lab-counts/*"
        },
        {
            "Sid": "AllowAggregateObjectWritingFromCentral",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::<central account>:root"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<bucket name>/aggregate-counts/*"
        }
    ]
}
```

Appendix B

The full solution architecture is shown below for reference – noting that most of the workings of the central system are irrelevant from the perspective of the lab integrations.