VarGraph: A Decision Support Tool for Variant Classification Using Pathway Databases

By

David Ball

Presented to the Department of Medical Informatics and Clinical Epidemiology and the Oregon Health & Science University School of Medicine
in partial fulfillment of
the requirements for the degree of
Master of Science, Bioinformatics and Computational Biomedicine
August 2019

School of Medicine
Oregon Health & Science University

CERTIFICATE OF APPROVAL

This is to certify that the Master's Capstone Project of

David Ball

Has been approved

_____

Capstone Advisor

Guanming Wu, Ph.D.

Table of Contents

Acknowledgements

I would like to thank my advisor Dr. Guanming Wu for many lessons on networks and biology both in class and during this project. Many thanks as well to Diane Doctor for much guidance during this project.

I would like to thank Dr. Carlo Bifulco at the Providence St. Joseph's Healthcare Molecular Genomics lab for his vision and creation of the software paradigm in which I was able to add this component. And thank you to Dr. Brady Bernard and Dr. Christopher Dubay for many interesting insights.

Thank you to Robert Citek for help with docker and networking issues and with provisioning a powerful development environment.

Abstract

Background: Genomic screening is an increasingly important part of cancer care. Screening often detects somatic variants in the tumor sample of varied clinical significance: some are well understood, some are connected to only loose evidence, and some are unknown altogether.

Purpose: The purpose of this project was to construct a tool that could present state-of-the-art pathway information to genomics experts evaluating the clinical significance of variants.

Solution: A web service was created that runs queries against the Reactome pathway database in search of common pathway activity between variants of a clinical case that are known to be pathogenic and those that are of unknown significance. It was integrated into the software infrastructure of a high-volume genomics lab at Providence St. Joseph's Health. A very different approach to the same problem was attempted via the cloud database product Google BigQuery.

Conclusions: The project so far has failed to be of clinical utility. Two areas of improvement could remedy that situation in future iterations: a more stable network visualization technique, and higher resolution mapping of novel variants to pathway databases via accounting for the effect of alterations on particular protein subdomains. In its current form the project was not able to reap the benefits of a graph database in particular. A simpler focus on a small number of "canonical pathways" looks like a quicker path to a value-added user interface.

Chapter 1: Introduction

Genomic sequencing of somatic mutations in tumor and blood malignancies is becoming an increasingly important part of cancer care (1). As the cost of sequencing declines and clinically actionable information that can be learned from sequencing increases, it is more and more relevant. Clinical genomics labs must not only assess what variants are present in a patient's cancerous cells but filter down to those that are most clinically significant. "Clinical significance" might entail that there are known therapies that have been found to be successful in treating similar cancers, or that there is a reason to infer from the presence of that variant information about mechanism, prognosis, or resistance. Myriad efforts exist around the world to curate data from a variety of cancer datasets to aid in efforts to predict the functional role of variants (2,3,4).

It is common in clinical practice to find multiple somatic variants in a malignancy, and for some of them to be linked to known pathogenicity across databases and for some of them not to be (2,3). In other words, in the general field of personalized oncology, we have more data than we know what to do with. Clinical genomics labs all over the world are finding variants in tumors that may one day be actionable, while the current state of knowledge does not link them to known therapies. There is an "urgency"(3) to this problem and anything that can be done to help the discovery of the clinical significance of a novel unknown variant could potentially be of huge importance to a patient.

One way of assessing the clinical significance of variants is to look at what pathways are affected. Many large efforts are on-going to aggregate all known biochemical pathways into shareable datasets. This project aimed to make some of that knowledge in

pathways databases promptly accessible to genomics experts issuing clinical case reports.

The focus of this work was to create an annotation and visualization pipeline to enhance clinicians' abilities to assess the clinical significance of poorly characterized variants. The focus for this stage was on the Reactome database which is freely available on the internet as a Neo4j database. The aim was to add a pathway-based visualization component to an existing annotation and interpretation pipeline in a high-volume Molecular Genomics lab at a major medical center.

An application was constructed called VarGraph (for "Variant Graph"). VarGraph is web-application that queries the graph database from Reactome based on the set of variants found in a particular case and shows pathway information to the clinical interpreter in graphical form. I will also discuss some further analysis of historical cases that was done using a combination of VarGraph and the BigQuery engine component of the Google Cloud Platform.

Genesis of the Project

I have been working in the Providence St.-Joseph's Health Molecular Genomics Lab (PSJH-MGL) in Portland, OR for the last year on a suite of in-house software that is used for the review, annotation, and interpretation of genomics data results for oncology patients. I had some experience with Reactome in my studies at OHSU and was interested in exploring how data in Reactome could be used to enhance the information that our in-house users have available in making their decisions on clinical significance. We depend on third-party analysis for our pathway information and get it back in text format, rather than a more refined data structure that could be used for custom visualizations. Our

Medical Director, Dr. Bifulco, expressed an interest in seeing more pathway analysis in our clinical review process and even incorporating it into our reports if it met a quality threshold. My advisor Dr. Wu advised me early on that the level of resolution that I could get from Reactome was not likely to be relevant enough as it did not have 3-D structure information. I persisted with the idea to use Reactome in hopes that it would help me to create a good initial framework of pathway visibility in our software that could be further enhanced down the line with more precisely targeted analyses.

Chapter 2: Background

Variant Classification

The landscape of cancer genomics has changed so rapidly that it is reasonable to expect that many of the final consumers of clinical genomics reports will not have been formally trained in the field (5). They will rely on the reporting of the genomics analysis to connect them with the best possible treatment options for their patients. There are many avenues to predicting the functional consequences of novel variants, and quite a variety of projects and databases that aim to curate and organize some of these. Private companies combine the use of databases and expert researchers to provide annotation services, such as N-Of-One, with whom we have a working relationship in the PSJH-MGL. Recent efforts include crowdsourcing the functional annotations of molecular pathologists world-wide and making the resulting dataset available, such as CivicDB (6). Any given uncharacterized mutation could be an important driver of the tumor process, or an incidental artifact along for the ride, a so-called passenger mutation (7). In the end, variant classification by clinical significance comes down to: does the presence of this variant tell us something about the mechanism of disease, the availability of FDA approved therapies, available clinical trials, drug resistance risks, and/or prognosis. Often associations are filtered by "Levels of Evidence" criteria. The goal is to give the physician closest to the patient the most actionable information possible without drowning them in a deluge of low-probability signals. Overby et al. (5) defined a taxonomy of stages of variant analysis consisting of preanalytic, analytic, and postanalytic phases. The work of my capstone was to create a tool to aide users in the postanalytic phase, the "reporting and interpreting of

genetic test results."

Pathway Analysis

Each cancer is not only unique in its exact genomic makeup but itself changes over time. Many cancers evolve resistance to the therapies which are initially helpful against them and become increasingly difficult to treat. When multiple therapeutic targets can be found in the same pathway, it can sometimes not only improve outcomes but lower side effects (8). It is therefore vital to find any valid inferential information about novel variants, while not overburdening the oncologist with low-probability and low-quality information. A study of 10,000 tumors from The Cancer Genome Atlas (TCGA) focused on 10 key pathways and found that "Eighty-nine percent of tumors had at least one driver alteration in these pathways, and 57% percent of tumors had at least one alteration potentially targetable by currently available drugs." (9) As I learned this during the course of this work it both inspired me to persevere with bringing pathway analysis into the PSJH-MGL pipeline but also led to some questions about my approach. That analysis (9) brings some interesting nuance to the question of "driver" and "passenger" mutations, in that they find that there are some cases where the presence of a damaged pathway by one variant makes it *less* likely to find another mutation in the same pathway. They mapped the patterns of "co-occurence and mutual exclusivity" in the TCGA dataset. I became interested in performing a similar analysis on our dataset at the PSJH-MGL. Such correlations could be used, perhaps, as part of a chain of inference of predicting if an unknown variant is related to a pathway that another known variant in a given tumor affects. An important concern in any assessment of significance is the "level of evidence" available behind any assertion. Blucher et al. created a comprehensive set of drug-target information categorized by levels

of evidence (10). Applying evidence-level-based filtering to Reactome pathways, they found that the portion of Reactome pathways targeted by approved antineoplastic drugs ranges from 39%-60% as level of evidence filtering moves from "strong evidence" to "any type of supporting evidence." (10). In a clinical setting, such as PSJH-MJL, there is a need to employ only strong levels of evidence in making assessments.

Database Technologies

As this project entailed usage of two different non-traditional database technologies, a few words of background about them might be helpful.

Graph Databases

In graph databases, data is represented as a collection of nodes and edges rather than the structured tables of rows and strictly defined columns that are employed in traditional relational databases. A particular database engine called Neo4J has become popular in recent years and implements a data structure called a property graph. This property graph contains nodes and edges and each node and edge can contain a user-defined set of key-value pairs to indicate properties. Neo4j provides a query language called Cypher, which has SQL-like syntax but many differences. Graph databases, including Neo4j, particularly distinguish themselves from relational database in queries involving path searches across sequences of nodes. This proved to be highly useful to the Reactome team and they reported that, in some circumstances, it reduced their average query time by 93% (11, 14). I was eager to explore the query possibilities in such a graph database for this project in hopes that it could allow the asking of questions that are not tractable to state in traditional relational databases.

Cloud Databases

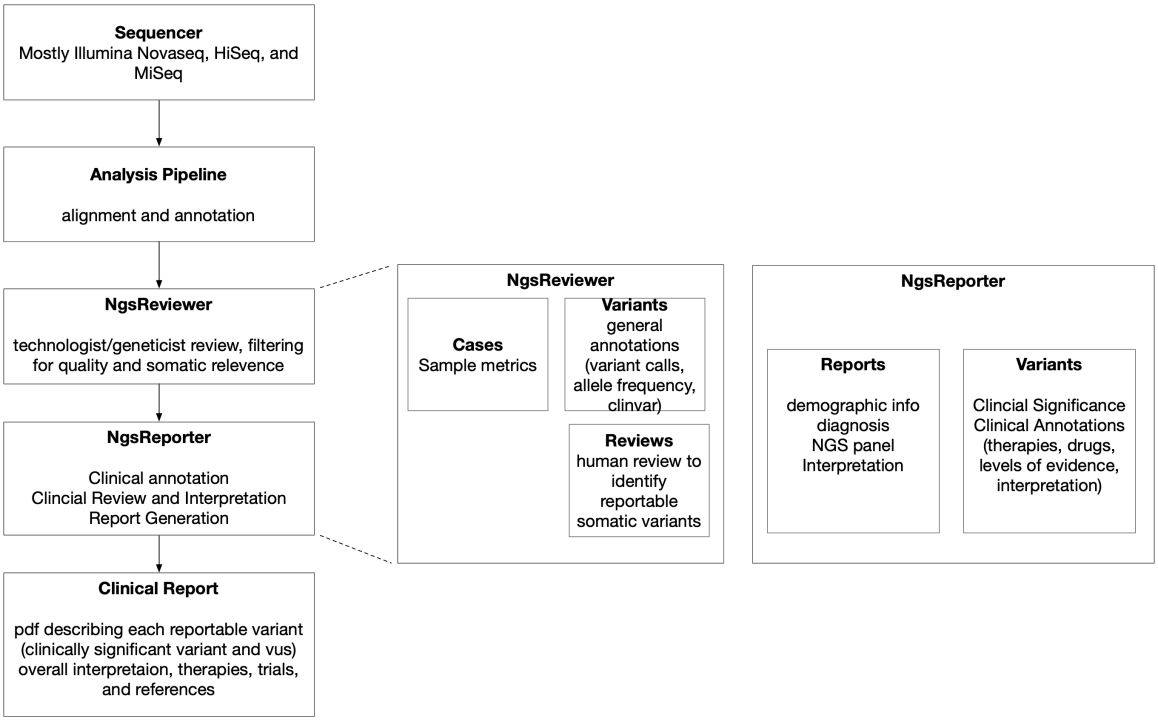
Another non-traditional database technology that was used in this project is a Google service known as BigQuery. It is a columnar database (12) that, to their users, is "serverless," and "infinitely scalable". Within the security of your own projects you can upload vast datasets but query them much like a traditional SQL database. BigQuery is compatible with SQL and adds a number of additional useful operators. Data is broadly distributed in ways that are not transparent to the end user, but can very efficiently perform queries on vast sizes of data. It has a billing model that is worth consideration for some cases: billing is in proportion to how much data each individual query touches. The capability of performing queries on petabyte-sized datasets without having to maintain internal hardware capacity for scale can make BigQuery a compelling solution for some use cases. Many datasets, such as The Cancer Genome Atlas data (12), are publicly available. The Health Information Portability and Accountability Act (HIPAA) requires that a Business Associate Agreement (BAA) be in place between a healthcare organization and a cloud provider before it can be used with any potential Patient Health Information (PHI) (13). Fortunately for this project, such an agreement was already in place at the time of initiation and Google Cloud, including BigQuery, was already a vital part of the computational infrastructure in our operations in the PSJH-MGL.

Molecular Genomics Lab at Providence-St. Joseph's Health

PSJH-MGL performs genomic sequencing for patients throughout Oregon and increasingly throughout the wider 7-state Providence system. Data is tracked through a variety of in-house applications, especially two titled NgsReporter and NgsReviewer. Over the last few years of operation these applications and their databases have accumulated a

store of sequencing results from approximately 4,500 solid tumors and hematological malignancies.

The PSJH-MGL dataset contains approximately 4,500 cases of sequenced DNA and/or RNA samples from solid tumor and hematological malignancies. Data goes through many internal pipelines and workflow steps during which variants are filtered for quality and relevance. A human reviewer evaluates each high-quality variant call and makes further effort to identify sequencing noise. Common germline polymorphisms are excluded, primarily by referencing EXAC and GNOMAD scores (14). After all this review a set of "reportable" variants for each case is determined.



1.a. High-level bioinformatics workflow in the MGL 1.b. Detail on the variant review and annotation pipeline

Fig. 1. Workflows and Software in the Providence St. Joseph's Health Molecular Genomics Lab (PSJH-MGL)

Aims

The aim of this project was to provide information to help molecular pathologists classify variants of unknown significance in clinical reports. The existing pipeline includes an abundance of annotations, including textual descriptions of implicated pathways, but did not have any visualization of pathway relationships nor direct links to pathway databases. The aim was to construct a tool that could put knowledge embedded in existing pathway databases into a readily reviewable form for the busy molecular pathologist, and to build such a tool in a way that it could be easily integrated into the existing human review and curation workflow. In particular the aim was to attempt to find links between the variants of unknown significance (VUS) and the clinically significant variants by means of using the graph database provided by Reactome.

Chapter 3: Materials and Methods

Datasets

PSJH-MGL Datasets

The datasets of the applications NgsReporter and NgsReviewer contain records from several years of sequencing of tumors and hematological malignancies for both clinical and research purposes. The genomic sequencing of RNA and DNA libraries outputs files in BCL format that are converted into FASTQ files and processed by alignment tools provided by the sequencing vendors. Most of our samples were sequenced on the Illumina platform. The datasets relevant to this project come from the NgsReviewer and NgsReporter applications, which consume the output of the alignment pipelines. Within NgsReviewer we have approximately 41 million raw records of variant information. The majority of this output is noisy and not of immediate relevance, but of those 195,000 rows were filtered for review by technologists to validate the call and make distinctions between somatic variants and common germline polymorphisms. Of these, approximately 28,000 variants were flagged as "reportable" and sent to molecular pathologists for clinical significance determination. For this analysis I focused on 2,800 cases that had at least one reportable variant, the dataset includes a total of 11,389 reportable variants, of which 5,306 (46.6%) were classified as clinically significant. Those that are not found to be significant are reported to be "Variants of Unknown Significance" (VUS).

Classification	Count	Average Per Case
Clinically Significant	5306	1.895
Variant of Unknown Significance (VUS)	6083	2.1725
Likely Significant	217	0.0775

Table 1. Classifications of Reported Variants

After reviewing all the aggregated data, an assessment of clinical significance is made into one of two categories: variants with clinical significance, and variants of unknown significance (vus). If a variant has been identified by the prior information pipeline as having good diagnostic, prognostic, explanatory role, and/or there is high quality evidence to directly link a variant with recommended therapies or trials, then it is deemed clinically significant. But the opposite claim of "insignificant" is not made. There is simply not enough known about the implications of any possible variant to make the affirmative negation of significance. VUS is used when the best available evidence can lead to no positive conclusion of significance. For the purposes of this project, I extracted a highly simplified dataset consisting of a case id, diagnosis, and list of variants in each of the two categories.

Data Exploration and Reactome Query Design

Reactome Queries

The first query design challenge was how to relate the PSJH-MGL datasets to the

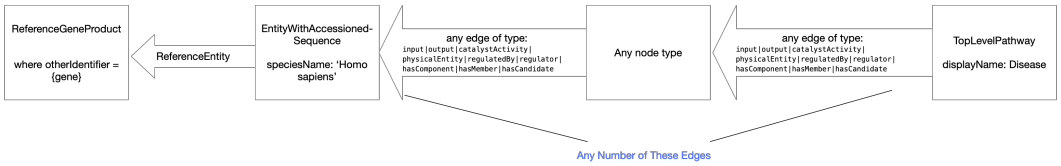
Reactome database in order to perform the desired pathway queries. The first stage of the project was to create queries of Reactome. Two aspects of the query construction challenge at every step were: (a) how to accurately map to the correct part of the Reactome graph and (b) how to make tractable queries that return in a reasonable amount of time.

Ultimately what is most significant about every variant will be a function of the precise three-dimensional structure of the protein produced by it and in the precise way that that altered protein interacts with every other molecule that it encounters. However, that level of resolution remained out of scope for what I was able to accomplish in this project. The analysis was limited to "gene" level pathways under alteration.

The Reactome database contains a rich interconnected database of nodes of various kinds and how they relate to each other. Though Neo4j is designed for high-performance querying of graphs, the large size of the Reactome graph means that caution must still be used to avoid computationally expensive queries. Neo4J queries are composed in a language called Cypher, which has been described as a version of the widely known Structured Query Language (SQL) for graph structures (15). Like SQL, the language is "declarative," such that the user is telling the query engine what it wants to see rather than explicitly how to retrieve that info. A key to efficient query design in this project was to use the WITH clause, which "allows query parts to be chained together, piping the results from one to be used as starting points or criteria in the next." (15) Using the WITH clause improved both the readability and execution times of my queries. Filtering Reactome queries by species early in the chain of WITH clauses was another important consideration for performance.

Query 1

Initially my aim was to construct a single query that would find all the pathways that involve any of the protein products of any of the reportable variants of a clinical case. I encountered problems with this approach both with query performance and with the complexity of the Cypher expression. Furthermore, it became interesting to see a pairwise comparison of the common pathways between each pair of variants. I took the approach of making a separate Reactome query for each variant in a case and aggregating the results into a matrix of variants by pathway outside of the database query. Figure 2 illustrates the query, which could be loosely summarized in natural language as: Find all paths of any length that begin with nodes representing the gene of a variant, end with a "top level pathway" named "Disease", and are composed of edges that indicate participation in a reaction or pathway.




```
MATCH (n:ReferenceGeneProduct) where (('BRAF' in n.otherIdentifier))
WITH n
MATCH (n)<-[r1:referenceEntity]-(re:EntityWithAccessionedSequence{speciesName:'Homo sapiens'})
WITH n,r1,re
MATCH path1=(re:EntityWithAccessionedSequence{speciesName:'Homo sapiens'})<-[r:input|output|catalystActivity|physicalEntity|
regulatedBy|regulator|hasComponent|hasMember|hasCandidate*]-(c)
WITH re,r,c,r1,n,path1
MATCH path2=(p:TopLevelPathway{displayName:'Disease'})-[e:hasEvent*]->(c)
WITH nodes(path2) as pns
UNWIND (pns) as pn
WITH pn
MATCH (pn:Pathway)
RETURN distinct pn
```

Fig. 2. Cypher Query with a diagram of the logic

With the precise identify of a ReferenceGeneProduct in mind, it is possible to find genetic variants of that gene by finding those nodes of type EntityWithAccessionedSequence that both reference the gene of interest and have an edge of type :hasModifiedResidue.

```
match (braf:ReferenceGeneProduct)-[:referenceEntity]-
(v:EntityWithAccessionedSequence)-[:hasModifiedResidue]-
```

```
(r:GeneticallyModifiedResidue) where id(braf)=238230 return
v,r
```

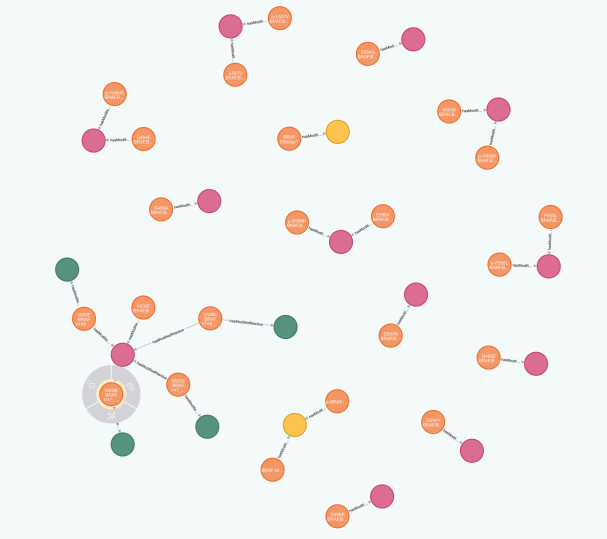


Fig. 3. Neo4J's native visualization of a Reactome query for gene BRAF

In future work this may allow a greater specificity of matching some of our well-characterized variants to appropriate nodes in the Reactome database. However, our VUS are, by definition, generally not well characterized in the literature and thus also unlikely to be annotated in Reactome.

The VarGraph Application

A web application called VarGraph was constructed with the goal of performing useful pathway queries in the Reactome database, assembling the results, and presenting them in a User Interface. VarGraph is a web application written in Python and utilizing the Flask micro web framework (16).

VarGraph accepts as input any one of the following: a list of genes, a list of transcript ids, or an order identification number of a clinical case in the PSJH-MGL. The primary output of the user interface is a force-directed layout of a graph structure in which

the nodes are Genes and Pathways. It is important to note that VarGraph is only representing pathway interactions of normal products of the gene of the variant, and, in its current form, cannot show specific functional differences of the particular variant in these pathways. Genes are color coded by the current PSJH-MGL clinical significance classification of the variant. The core feature set of VarGraph is oriented around a single clinical case at a time. For each clinical case, the purpose of VarGraph is to:

1. Retrieve the latest data on reportable variants for the case. It is intended to be usable on cases that are actively undergoing interpretation, so this data could change frequently. Generally, these variants would be in a preliminary state of classification by clinical significance already, and the interpreter would be looking to VarGraph to access additional information to complete the classification process.
2. Identify pathways in common between the genes of the clinically significant variants and the genes of the variants of unknown significance, or pathways common among the variants of unknown significance.
    2.1. Query to a local Reactome Neo4j database.
    2.2. Prepare a matrix of pathways by gene.
    2.3. Use the matrix to identify pathways that are common.
3. Present the findings to the user
    3.1. Render a network forced-layout visualization in the user interface.
    3.2. Display the matrix of genes by pathway.
    3.3. Provide links to third party sites for additional information.

VarGraph User Interface

The VarGraph is a browser-based application. Analysis for a clinical case is

accessed via an accession number in the URL. In the resulting screen rendered in the user's browse, the list of genes to be analyzed is shown at the top of the page. A link to open the same gene list for analysis in PathwayCommon.org is also shown. Next, the primary network visualization is displayed (Figure 4), showing nodes for each gene of a reportable variant and any associated pathways. Gene nodes are color coded to indicate the current clinical significance classification. Pathway nodes are color coded to indicate whether the pathway was found to be in "common" between the gene of a VUS and the gene of another variant. The purpose of the use of color is to allow a busy reviewer to look quickly at the visualization and see if this visualization is offering them information of interest.

I used the JavasScript-based visualization library D3.js as the foundation of the network visualization and made use of the particular technique of Mike Bostock (17) for dealing with Disjoint Graphs. Rendering of the force-directed layout is done within the browser using JavaScript that operates on a JSON payload retrieved from the VarGraph server. Pathway nodes and gene nodes are linked by an edge if the Reactome queries above found an association with that pathway. This definition of "edge" is a major area for expansion of the VarGraph platform in future work. The user can hover over to see more information about a node.

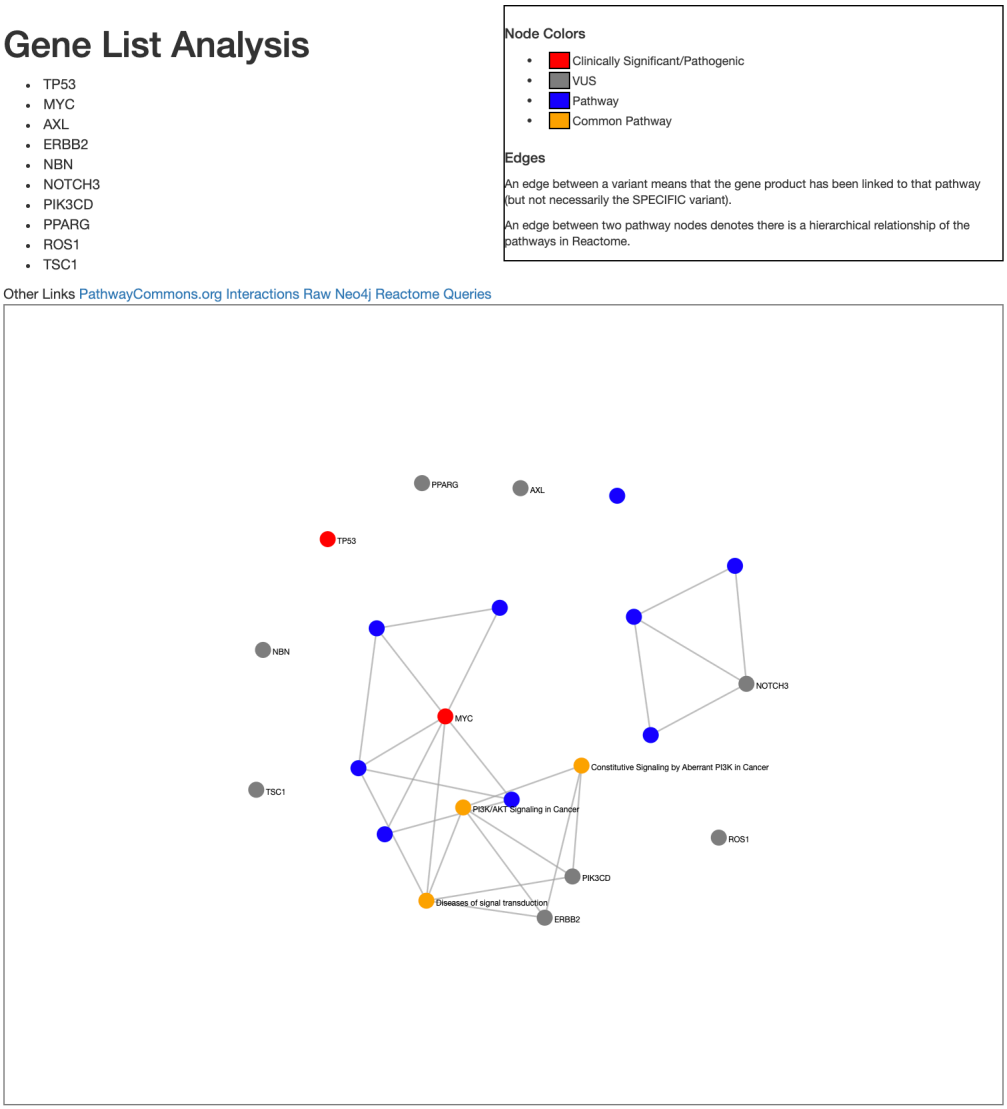
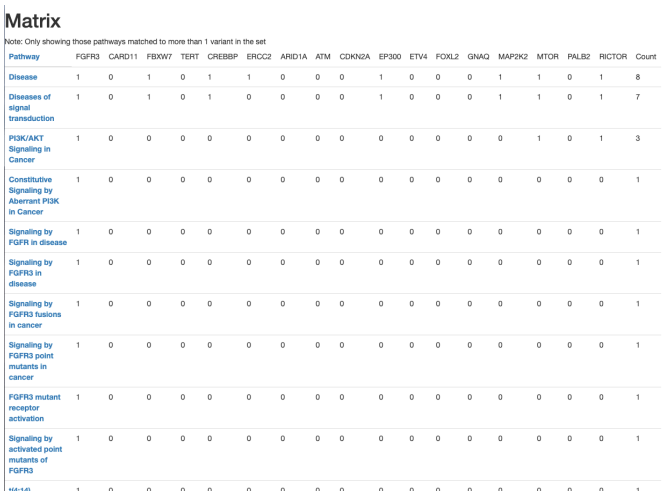


Fig. 4. VarGraph network visualization. The Orange color nodes highlight pathways found among the VUS

Below the network visualization, a matrix of genes by pathway is displayed (Figure 5). A "1" in a matrix cell indicates that the Reactome queries that VarGraph performed found an association between the gene and that pathway. The last column shows the count of genes that were mapped to a pathway in that row.



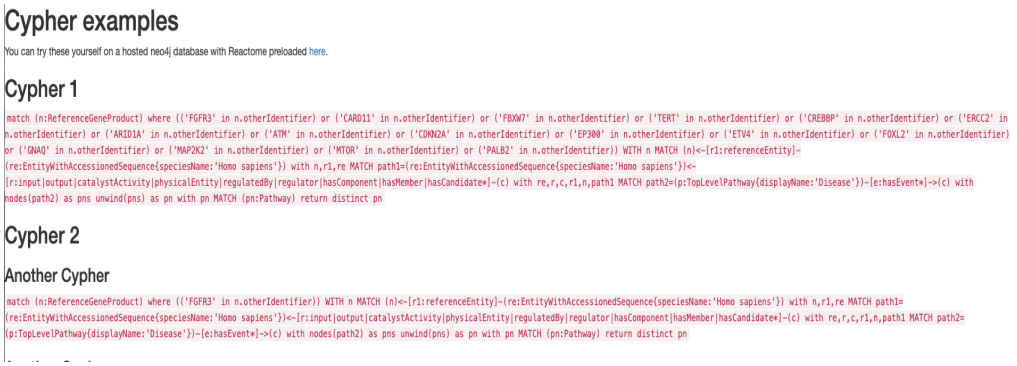
Matrix

Note: Only showing those pathways matched to more than 1 variant in the set

Pathway	FGFR3	CARD11	FBXW7	TERT	CREBBP	ERCC2	ARID1A	ATM	CDKN2A	EP300	ETV4	FOXL2	GNAQ	MAP2K2	MTOR	PALB2	RICTOR	Count
Disease	1	0	1	0	1	1	0	0	0	1	0	0	0	1	1	0	1	8
Diseases of signal transduction	1	0	1	0	1	0	0	0	0	1	0	0	0	1	1	0	1	7
PI3K/AKT Signaling in Cancer	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	3
Constitutive Signaling by Aberrant PI3K in Cancer	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Signaling by FGFR in disease	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Signaling by FGFR3 in disease	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Signaling by FGFR3 fusions in cancer	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Signaling by FGFR3 point mutants in cancer	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
FGFR3 mutant receptor activation	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Signaling by activated point mutants of FGFR3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fig. 5. A Matrix of Gene By Pathway for a single tumor case as displayed in VarGraph. A 1 indicates the inclusion of some gene product in the given pathway.

While Neo4J was not previously a part of our operational software infrastructure at PSJH-MGL, many of the users of our variant review software have a variety of data manipulation and programming skills. In consideration that some might want to learn Neo4J's Cypher query language I provided in VarGraph links directly to the hosted Neo4j version of Reactome that is running for VarGraph. The VarGraph user interface for an analysis of a gene set provides the raw Cypher that users could use as a starting point for refining the particular query that interests them.



Cypher examples

You can try these yourself on a hosted neo4j database with Reactome preloaded here.

Cypher 1

```
match (n:ReferenceGeneProduct) where (('FGFR3' in n.otherIdentifier) or ('CARD11' in n.otherIdentifier) or ('FBXW7' in n.otherIdentifier) or ('TERT' in n.otherIdentifier) or ('CREBBP' in n.otherIdentifier) or ('ERCC2' in n.otherIdentifier) or ('ARID1A' in n.otherIdentifier) or ('ATM' in n.otherIdentifier) or ('CDKN2A' in n.otherIdentifier) or ('EP300' in n.otherIdentifier) or ('ETV4' in n.otherIdentifier) or ('FOXL2' in n.otherIdentifier) or ('GNAQ' in n.otherIdentifier) or ('MAP2K2' in n.otherIdentifier) or ('MTOR' in n.otherIdentifier) or ('PALB2' in n.otherIdentifier)) WITH n MATCH (n)<-[r1:referenceEntity]-(re:EntityWithAccessionedSequence{speciesName:'Homo sapiens'}) with n,r1,re MATCH path1=(re:EntityWithAccessionedSequence{speciesName:'Homo sapiens'})<-[r:input|output|catalystActivity|physicalEntity|regulatedBy|regulator|hasComponent|hasMember|hasCandidate*]-(c) with re,r,c,r1,n,path1 MATCH path2=(p:TopLevelPathway{displayName:'Disease'})-[e:hasEvent*]->(c) with nodes(path2) as pns unwind(pns) as pn with pn MATCH (pn:Pathway) return distinct pn
```

Cypher 2

Another Cypher

```
match (n:ReferenceGeneProduct) where (('FGFR3' in n.otherIdentifier)) WITH n MATCH (n)<-[r1:referenceEntity]-(re:EntityWithAccessionedSequence{speciesName:'Homo sapiens'}) with n,r1,re MATCH path1=(re:EntityWithAccessionedSequence{speciesName:'Homo sapiens'})<-[r:input|output|catalystActivity|physicalEntity|regulatedBy|regulator|hasComponent|hasMember|hasCandidate*]-(c) with re,r,c,r1,n,path1 MATCH path2=(p:TopLevelPathway{displayName:'Disease'})-[e:hasEvent*]->(c) with nodes(path2) as pns unwind(pns) as pn with pn MATCH (pn:Pathway) return distinct pn
```

Fig. 6. Cypher examples as displayed in the VarGraph User Interface.

Integration into Existing PSJH-MGL Software Stack

To be useful, of course, VarGraph needed access to actual sets of variants reported for a particular case. Some possible ways that data could have been exchanged between the two applications include:

- Data push from NgsReporter to VarGraph
- Data pull from VarGraph to NgsReporter
- NgsReporter and VarGraph each connect to a common data source
- Periodic synchronization of data from NgsReporter to VarGraph

There are ample tradeoffs involved in each approach. The twin goals of secure communication, and requiring minimal impact on existing software, argued for keeping VarGraph separate from any raw clinical data source. Thus, I gave VarGraph the ability to make secure encrypted API calls to NgsReporter using the existing authentication and interface mechanisms of NgsReporter, minimizing the need to make changes to NgsReporter for VarGraph's needs. Though it could be desirable in some circumstances to also have VarGraph capable of receiving an HTTP post of variants upon which to do its analysis, in our present circumstances in the lab it proved to be a lower impact on existing systems to have VarGraph do the pulling when it needs data.

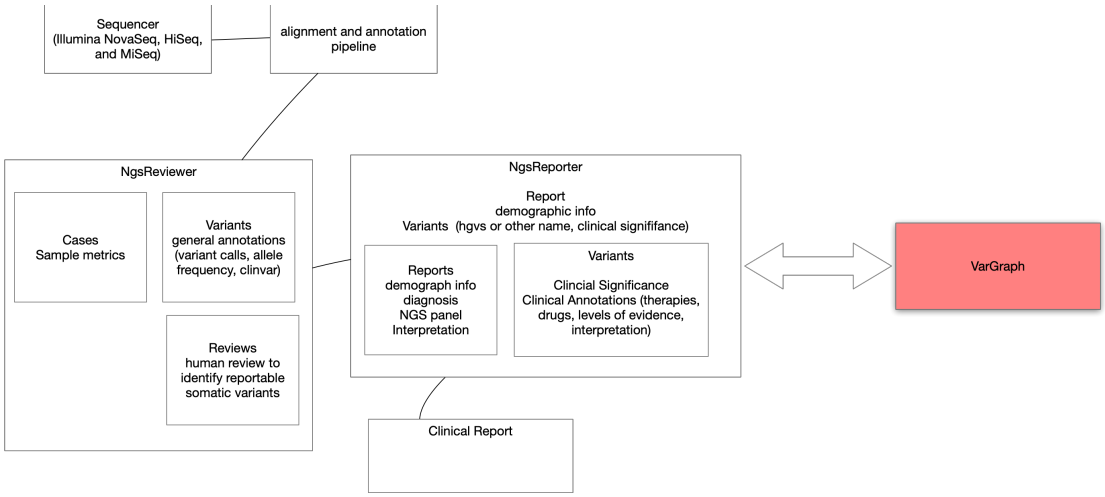


Fig. 7. Where VarGraph fits in the PSJH-MGL stack

The user interface of VarGraph could be integrated into clinical operations with very minimal change to the core clinical application NgsReporter. A hyperlink added to the HTML page for clinical interpretation takes the user to the VarGraph application ("Preview Pathway Search") (Figure 8).

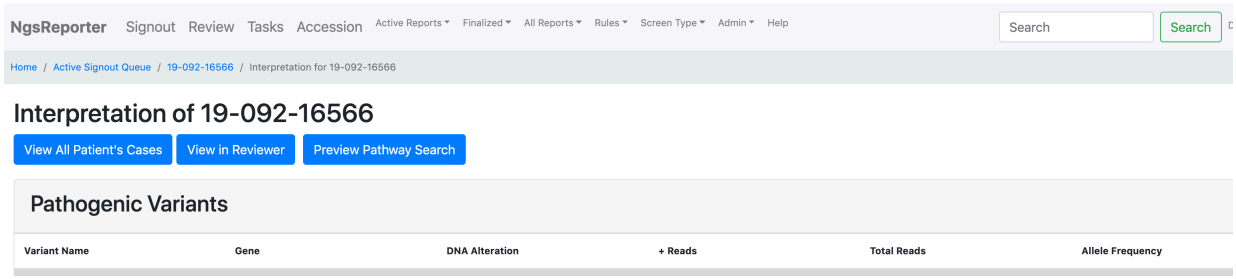


Fig. 8. VarGraph integration in the PSJH-MGL's in-house Interpretation and Reporting software, NgsReporter

VarGraph needed an authentication system for deployment in a production environment, to prevent unauthorized access to the application. A session-based authentication mechanism was built into VarGraph with authentication governed by

environment variables. Users must login with a username and password in order to use it.

VarGraph was deployed to a virtual machine in a development environment. To ease the burden of deploying VarGraph in new environments, a Docker container was created that bundles both the Neo4j database engine itself along with the Neo4j database from Reactome. Docker is a containerization technology that is in widespread use across many domains. It has been very useful in the life sciences realm, aiding with performance, isolation of concerns, reproducibility, and scalability (18). Reactome generously provides a downloadable Neo4j database edition of their data (19). Instead of requiring the downloading of Reactome data and configuration of a Neo4j server to connect to the downloaded data with each and every deployment, the reusable Docker container was stored in a container registry for easy access from any new deployment environments.

The entirety of the configuration required to specify the dockerization is shown in Figure 9. File Dockerfile_neo4j_Reactome uses a base Docker image provided by Neo4j that has their server installed and ready to go. For VarGraph, the only necessary changes required to the baseline Neo4j Docker image were the importing of the Reactome data and the exposing of appropriate ports (7474, 7473, 7687). The Dockerfile for VarGraph is similarly quite simple, beginning with a base Python container and adding the necessary code directories from VarGraph and installing Python dependencies.

Docker Compose is a technology provided by Docker used to orchestrate clusters of containers. For VarGraph, a small cluster of two docker containers was created: one container was the Neo4j_Reactome service and the other was the VarGraph application service (the Python Flask application). The cluster can be brought up with a simple command “docker-compose up”, run from within the application’s working directory.

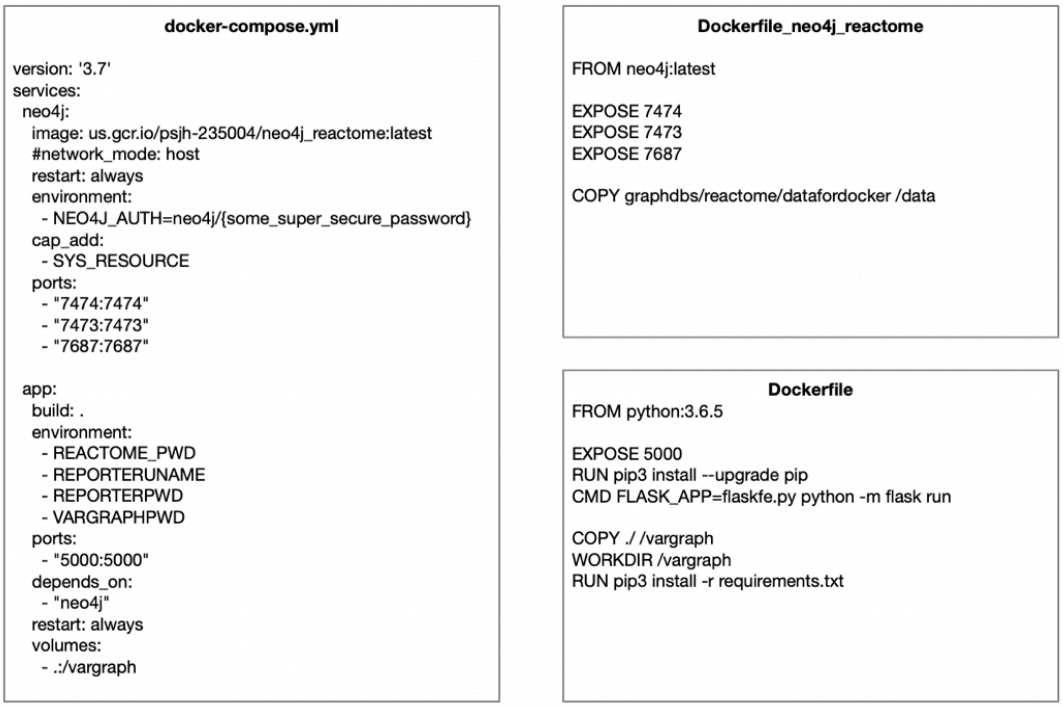


Fig. 9. Docker configuration.

Execution and rendering times for the preparation of each VarGraph analysis depend on the number of variants and the particular variants searched but are often in the range of 10-30 seconds in an early version. This is an unacceptable latency period for a good user interface. For that reason, a caching mechanism was added. Via the caching mechanism, the VarGraph application persists its matrix and JSON description to a private caching directory on its host machine. Upon receiving a new request to analyze a case, VarGraph first inspects its cache and only commences a fresh analysis if no cached content is available. In a production clinical system, this workflow would lead to a significant cache invalidation problem that was not addressed yet at the present time. Subsequent optimizations made the caching mechanism less necessary for application usage, but the cached output nonetheless proved useful for aggregating the results of running VarGraph

across all available cases.

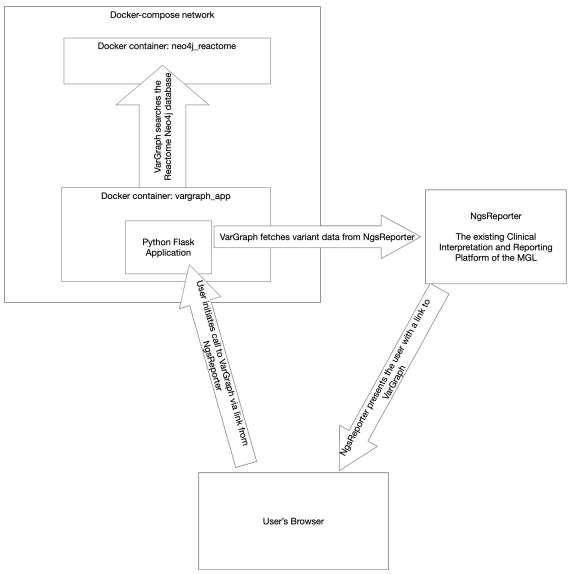


Fig. 10. VarGraph Deployment topology.

Static Reactome File-based Analysis

The initial approach of VarGraph was to perform queries of Neo4j in real-time. This proved to be less efficient and effective than working directly with files provided by Reactome that map NCBI gene ids to Reactome nodes of type "PhysicalEntity". An alternate workflow in VarGraph was created that works directly with these static files and does not require dynamic querying via Neo4j. Human gene symbols from the MGL records were mapped to NCBI gene ids and from there to the Reactome Physical entities. This method both executes much more quickly and has the advantage of being based on a well-vetted canonical mapping of genes to Reactome pathways.

Analysis Using VarGraph

VarGraph was run offline to process and cache list of 4,597 PSJH-MGL accession

numbers. A python script[1] read through all the cached JSON to generate a CSV file that was uploaded into BigQuery. This resulted in 1,611 accession numbers in which more than one gene was used in the analysis. The cached summary of each case was stored in the file system in JSON and CSV formats. I parsed these results with a python script and uploaded them to BigQuery for analysis via SQL queries.

BigQuery based pathway analysis

While it was not an initial aim, at this point in the project I became interested to see how our dataset would compare to the previously mentioned analysis of TCGA data along 10 canonical oncogenic signaling pathways (9). How would our dataset compare to their findings of 89% of tumors having at least one driver alteration in one of ten key pathways? As our datasets were already available in BigQuery, there was a simple solution: I transcribed the gene lists from the paper (9), uploaded it into BigQuery, and created some SQL views that tied the relevant data structures together. The query structure and results will be described in the Results section.

[1] accession_json_cache/find_cache_stats.py

Chapter 4: Results

VarGraph Usability

VarGraph was implemented and deployed in production such that it can optionally be used by case interpreters. I did not deem it useful enough at this stage to subject the user base to a significant usability test. There is a problem of visual noise. The force-directed layout is not ideal for highlighting the most important information and the inherent inconsistencies in a force-directed layout are not ideal but improved visualization techniques could improve the utility.

VarGraph Analysis

VarGraph with Live Neo4j

The mean time for processing a case analysis, excluding user interface rendering time, was 10.7 seconds though the longest was nearly 6 minutes. The mean number of "common pathways" found connecting the VUS to clinically significant variants or other VUS was 1.5, with a maximum of 55 "common pathways" found.

VarGraph with Static Reactome Files

With the VarGraph analysis based on static files from Reactome, using files pre-parsed in RAM, the mean time to analyze a case was only 3 milliseconds with the slowest completing in only 80 milliseconds. Figure 11 shows the most frequent "common pathways" identified by the analysis.

Pathway found in Common	# identified	Pathway Hierarchical Context
Constitutive Signaling by Aberrant PI3K in Cancer	201	Disease > Diseases of signal transduction > PI3K/AKT Signaling in Cancer > Constitutive Signaling by Aberrant PI3K in Cancer
Regulation of TP53 Activity through Phosphorylation	130	Gene expression (Transcription) > RNA Polymerase II Transcription > Generic Transcription Pathway > Transcriptional Regulation by TP53 > Regulation of TP53 Activity > Regulation of TP53 Activity through Phosphorylation
TP53 Regulates Transcription of DNA Repair Genes	121	Gene expression (Transcription) > RNA Polymerase II Transcription > Generic Transcription Pathway > Transcriptional Regulation by TP53 > TP53 Regulates Transcription of DNA Repair Genes
Pre-NOTCH Transcription and Translation	117	Signal Transduction > Signaling by NOTCH > Pre-NOTCH Expression and Processing > Pre-NOTCH Transcription and Translation
Regulation of TP53 Degradation	104	Gene expression (Transcription) > RNA Polymerase II Transcription > Generic Transcription Pathway > Transcriptional Regulation by TP53 > Regulation of TP53 Activity > Regulation of TP53 Expression and Degradation > Regulation of TP53 Degradation
Recruitment and ATM-mediated phosphorylation of repair and signaling proteins at DNA double strand breaks	94	DNA Repair > DNA Double-Strand Break Repair > DNA Double Strand Break Response > Recruitment and ATM-mediated phosphorylation of repair and signaling proteins at DNA double strand breaks
Resolution of D-loop Structures through Holliday Junction Intermediates	71	DNA Repair > DNA Double-Strand Break Repair > Homology Directed Repair > HDR through Homologous Recombination (HRR) or Single Strand Annealing (SSA) > HDR through Homologous Recombination (HRR) > Resolution of D-Loop Structures > Resolution of D-loop Structures through Holliday Junction Intermediates
TP53 Regulates Metabolic Genes	71	Gene expression (Transcription) > RNA Polymerase II Transcription > Generic Transcription Pathway > Transcriptional Regulation by TP53 > TP53 Regulates Metabolic Genes
Regulation of TP53 Activity through Methylation	65	Gene expression (Transcription) > RNA Polymerase II Transcription > Generic Transcription Pathway > Transcriptional Regulation by TP53 > Regulation of TP53 Activity > Regulation of TP53 Activity through Methylation
Interleukin-4 and Interleukin-13 signaling	64	Immune System > Cytokine Signaling in Immune system > Signaling by Interleukins > Interleukin-4 and Interleukin-13 signaling

Fig. 11. Top 10 Common Pathways identified by VarGraph (Static files version). The context of the pathway in the overall Reactome hierarchy of pathways in shown in the right column.

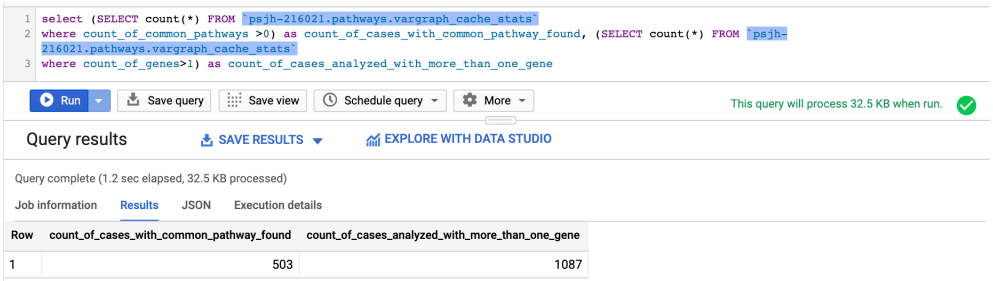


Fig. 12. Using BigQuery to analyze VarGraph results.

BigQuery Pathway Queries

In a side analysis, not directly relevant to VarGraph, but directly relevant to the *aim* of VarGraph, I did some analysis of our historical data against the gene lists from the Oncogenic Signaling Pathways (9) paper. The efficiency with which analysis can be done in BigQuery is sometimes remarkable. In this case, the analysis can be presented in the form of a single SQL (Figure 13) that returns in 1.0 second (Figure 14).

proportion_all_casesin_common_pathways

With a single SQL View

```
with
  accessions_with_common_pathway as (select count(*) as
count_in_common_pathway from (SELECT distinct
accession_number FROM `psjh-
216021.pathways.common_pathways_by_accn`)),
  all_accessions as (select count(*) as
count_all_accessions from (SELECT distinct accession_number
FROM `psjh-216021.pathways.variant_classifications`)),
  all_reportings as (select count(*) as
count_all_reportings from (SELECT distinct accession_number
FROM `psjh-216021.smr_scratch.reportings` where
deleted=false))

select
  a.count_in_common_pathway,
  b.count_all_accessions,
  c.count_all_reportings,
  a.count_in_common_pathway/b.count_all_accessions
proportion_all_with_any_variant,
  a.count_in_common_pathway/c.count_all_reportings as
proportion_all_reportings
```

```
from
  accessions_with_common_pathway a, all_accessions
b,all_reportings c
```

Fig. 13. SQL to generate pathway counts.

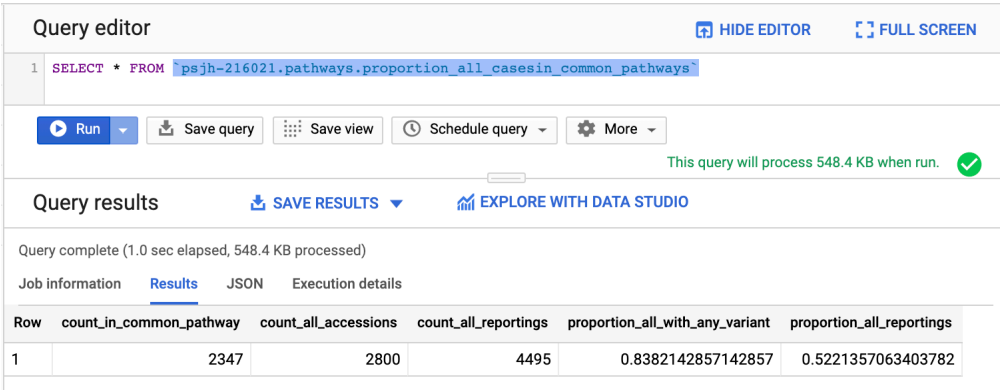


Fig. 14. Very fast but simple pathway analysis in BigQuery

As shown in Figure 14, approximately 84% of cases for which we reported at least one variant were found to have a variant of a gene in at least one of the 10 pathways covered by (9), using the same gene list for each pathway that the original paper did. This is a number that is of a similar magnitude to the 93% finding that (9) found in the TCGA dataset. This is an interesting finding and led to me and some of my colleagues at PSJH-MGL wanting to explore this further in future work.

Chapter 5: Discussion

Future

One possible way to facilitate the queries between clinical data and Reactome would be to merge it all together. It would be an interesting project to work with a secure private copy of the Reactome database and ingest clinical data into it. By creating edges from each variant to the most relevant place in the Reactome network once upon ingest, that part of the mapping would not need to be repeated with each query. Native cypher queries could be used to answer a variety of questions.

The analysis conducted so far was primarily focused on the gene level. It is not at a level of resolution where a difference between 2 mutations that affect different functional subdomains of a protein could be discerned. Obviously, the latter is what is actually most biologically relevant. The current system gives a probabilistic signal and a guide for users on when to look further. It would be good to extend the computational pipeline of VarGraph itself to account for these subtleties. One such avenue of extension to explore could be integrating data from dbFNSP, which provides "a list of all potential nsSNVs and ssSNVs based on the human reference sequence were created and functional predictions and annotations were curated and compiled for each SNV" (20).

As a user interface modality, the force-directed graph layout can be overly crowded and the continued moving of nodes adds strain for some users. Visualization via a hierarchical layout could be explored as a possible means of making a more predictable and explicit image for users. Viz.js (21) is a tool to render graphs described in the GraphViz format within a web browser. A library such as Viz.js could be used to render such a

hierarchical layout. Such a static layout would give a more consistent layout experience for users, and may also help render a repeatable static visualization that could be included in reporting, if the analysis quality of future versions of VarGraph grew to an acceptable level.

Chapter 6: Summary and Conclusions

VarGraph is useful as a proof of concept but is not ready for clinical use. I would like to build upon it at both ends, improving the underlying computational pathway alignment techniques, as well as the visualization techniques. There is an abundance of data available on many variants of unknown significance, but this data is of varying quality. I did not here solve the problem of using pathway information to present a fast, responsive and clear user interface for interpreters of genomic reports, but there is enough infrastructure in place to continue building upon it. VarGraph would benefit from adding additional data resources to its computational pipeline in addition to Reactome. And there is much more valuable data in Reactome that VarGraph could still make better use of, from laying out more reaction level data to directly presenting references to the reviewer. The problem of mapping variants in all their specificity to pathways will likely entail incorporation of other data sources.

Source code for VarGraph is available at https://github.com/davidball/VarGraph

References

1. El-Deiry WS, Goldberg RM, Lenz H-J, Shields AF, Gibney GT, Tan AR, et al. The current state of molecular testing in the treatment of patients with solid tumors, 2019. CA: A Cancer Journal for Clinicians. 2019;69(4):305–43.

2. Stockley TL, Oza AM, Berman HK, Leighl NB, Knox JJ, Shepherd FA, et al. Molecular profiling of advanced solid tumors and patient outcomes with genotype-matched clinical trials: the Princess Margaret IMPACT/COMPACT trial. Genome Medicine. 2016 Oct 25;8(1):109.

3. Chang MT, Bhattarai TS, Schram AM, Bielski CM, Donoghue MTA, Jonsson P, et al. Accelerating Discovery of Functional Mutant Alleles in Cancer. Cancer Discov. 2018 Feb 1;8(2):174–83.

4. Thorsson V, Gibbs DL, Brown SD, Wolf D, Bortone DS, Yang T-HO, et al. The Immune Landscape of Cancer. Immunity. 2018 Apr 17;48(4):812-830.e14.

5. Overby CL, Kohane I, Kannry JL, Williams MS, Starren J, Bottinger E, et al. Opportunities for genomic clinical decision support interventions. Genetics in Medicine. 2013 Oct;15(10):817–23.

6. Griffith M, Spies NC, Krysiak K, McMichael JF, Coffman AC, Danos AM, et al. CIViC is a community knowledgebase for expert crowdsourcing the clinical interpretation of variants in cancer. Nature Genetics. 2017 Jan 31;49:170–4.

7. Gao J, Chang MT, Johnsen HC, Gao SP, Sylvester BE, Sumer SO, et al. 3D clusters of somatic mutations in cancer reveal numerous rare mutations as functional targets. Genome Medicine. 2017 Jan 23;9(1):4.

8. Smalley KSM, Sondak VK. Skin cancer: Targeted therapy for melanoma: is double hitting a home run? Nature Reviews Clinical Oncology. 2013 Jan;10(1):5–

9. Sanchez-Vega F, Mina M, Armenia J, Chatila WK, Luna A, La KC, et al. Oncogenic Signaling Pathways in The Cancer Genome Atlas. Cell. 2018 Apr;173(2):321-337.e10.

10. Blucher AS, Choonoo G, Kulesz-Martin M, Wu G, McWeeney SK. Evidence-Based Precision Oncology with the Cancer Targetome. Trends in Pharmacological Sciences. 2017 Dec 1;38(12):1085–99.

11. Fabregat A, Korninger F, Viteri G, Sidiropoulos K, Marin-Garcia P, Ping P, et al. Reactome graph database: Efficient access to complex pathway data. PLOS Computational Biology. 2018 Jan 29;14(1):e1005968.

12. Inside Capacitor, BigQuery's next-generation columnar storage format [Internet].

Google Cloud Blog. [cited 2019 Aug 10]. Available from: https://cloud.google.com/blog/products/gcp/inside-capacitor-bigquerys-next-generation-columnar-storage-format/

13. Rights (OCR) O for C. Covered Entities and Business Associates [Internet]. HHS.gov. 2015 [cited 2019 Aug 17]. Available from: https://www.hhs.gov/hipaa/for-professionals/covered-entities/index.html

14. Karczewski KJ, Francioli LC, Tiao G, Cummings BB, Alföldi J, Wang Q, et al. Variation across 141,456 human exomes and genomes reveals the spectrum of loss-of-function intolerance across human protein-coding genes. bioRxiv. 2019 Jan 30;531210.

15. The Neo4j Cypher Manual v3.5 [Internet]. [cited 2019 Aug 17]. Available from: https://neo4j.com/docs/cypher-manual/current/introduction/

16. The Python micro framework for building web applications.: pallets/flask [Internet]. The Pallets Projects; 2019 [cited 2019 Aug 12]. Available from: https://github.com/pallets/flask

17. Disjoint Force-Directed Graph [Internet]. 2018 [cited 2019 Aug 12]. Available from: https://observablehq.com/@d3/disjoint-force-directed-graph

18. Grüning B, Chilton J, Köster J, Dale R, Soranzo N, van den Beek M, et al. Practical

Computational Reproducibility in the Life Sciences. Cell Systems. 2018 Jun 27;6(6):631–5.

19. Fabregat A, Jupe S, Matthews L, Sidiropoulos K, Gillespie M, Garapati P, et al. The Reactome Pathway Knowledgebase. Nucleic Acids Res. 2018 04;46(D1):D649–55.

20. Liu X, Wu C, Li C, Boerwinkle E. dbNSFP v3.0: A One-Stop Database of Functional Predictions and Annotations for Human Non-synonymous and Splice Site SNVs. Hum Mutat. 2016 Mar;37(3):235–41.

21. Daines M. A hack to put Graphviz on the web. Contribute to mdaines/viz.js development by creating an account on GitHub [Internet]. 2019 [cited 2019 Aug 19]. Available from: https://github.com/mdaines/viz.js