Spfy: an integrated graph-based sequence database for real-time prediction of Escherichia coli phenotypes using the SuperPhy platform

Kevin K Le*[1], Matthew D Whiteside[1], James Hopkins[1], Victor PJ Gannon[1] and Chad R Laing†[1]

[1]National Microbiology Laboratory at Lethbridge, Public Health Agency of Canada, Twp Rd 9-1, Lethbridge, AB, T1J 3Z4, Canada

February 22, 2018

Abstract

Current workflows in comparative computational genomics rely on chaining different analysis software together, but lack storage and retrieval methods for the generated results. Reference laboratories are currently moving to whole-genome sequencing based analyses, and require rapid analyses coupled to efficient storage and retrieval of data. To solve this problem, we have created Spfy, which uses a graph database to store and retrieve results from computational workflows. This facilitates rapid analyses, as well as the efficient management and downstream processing of tens of thousands of whole-genome sequences. Though generally applicable to bacterial genome sequences, Spfy currently contains X *Escherichia coli* genomes, for which *in-silico* serotype and Shiga-toxin subtype, as well as the presence of known virulence factors and antimicrobial resistance determinants have been computed. Spfy links the results and metadata to the genome sequences through a standardized ontology, which facilitates hypothesis testing in fields ranging from population genomics to epidemiology, while mitigating the recomputing of analyses. The graph approach is flexible, and can accommodate new analysis software modules as they are developed, and easily link new results to those already stored. Integrated data storage and analyses are currently necessary as the number of publicly available whole genome sequences is currently in the hundreds of thousands, with millions likely to be available within the next few years.

Database URL: `https://lfz.corefacility.ca/superphy/spfy/`.

1 Introduction

Whole genome sequencing (WGS) can in theory provide the entire genetic content of an organism. This unparalleled resolution and sensitivity has recently transformed public-health surveillance and outbreak response [**?**, **?**]. Additionally, the identification of novel disease mechanisms [**?**, **?**], and rapid clinical diagnoses and reference lab tests based on the specific mechanism of disease are now possible. [**?**, **?**].

The rapid characterization and comparison of bacterial pathogens relies principally on the combination of outputs from multiple software programs that are targeted for specific applications. Examples include the identification of known antimircrobial (AMR) resistance genes, through software such as the Resistance Gene Identifier (RGI) [**?**], ResFinder, ARG-ANNOT, and ARIBA; the identification of known virulence genes through software such as VirulenceFinder [**?**], SRST2, and an additional reference; the annotation of bacterial genomes, through software such as Prokka [**?**], another, and another. Add a section on serotyping and subtyping a la Phylotyper. These methods represent *in-silico* analogues of traditional wet-lab tests,

*kevin.le@canada.ca

†chad.laing@canada.ca

which allows new WGS results to be viewed in the context of historical tests, and greatly expedites the analyses of newly sequenced genomes.

Comprehensive platforms that combine individual programs into a cohesive whole also exist. These include free platforms such as the Bacterium Analysis Pipeline (BAP) [**?**], the Integrated Rapid Infectious Disease Analysis (IRIDA) project `http://www.irida.ca/`, and the Pathosystems Resource Integration Center (PATRIC) [**?**]. Commercial applications, such as Bionumerics, which is used by PulseNet International for the analyses of WGS data in outbreak situations also exist, and offer support as well as accredited, standardized tests [**?**].

Traditionally, these platforms have been applied to samples numbering only a few dozen [] while WGS of bacterial pathogens have recently accumulated in public databases in the hundreds of thousands, with millions set to be available within the next few years. For *Escherichia coli* alone, there are over sixty thousand publicly available genomes in EnteroBase `https://enterobase.warwick.ac.uk/` and three million sequenced genomes in GenBank [**?**].

Many of the comparative analyses that are currently used in the analyses of bacterial genomes are broadly useful, and therefore computed multiple times for the same genomes. An effective method to mitigate the recomputing of analyses, is to make the storage and retrieval of results part of the analyses platform, and effectively linked to the genomes of interest through a standardized ontology. Such measures can help ensure the rapid response times required for public health applications, and allow results to be integrated and progressively updated as new data becomes available.

We have previously developed Superphy [**?**], an online predictive genomics platform targeting *E. coli*. Superphy integrates pre-computed results with domain-specific knowledge to provide real-time exploration of publicly available genomes. While this tool has been useful for the thousands of pre-computed genomes in its database, the current pace of genome sequencing requires real-time predictive genomic analyses of tens-, and soon hundreds-of-thousands of genomes, and the long term storage and referencing of these results, something that the original SuperPhy platform was incapable of.

In this study, we present the Spfy update to the SuperPhy platform, which integrates a graph database with real-time analyses; this integration avoids recomputing identical analyses. Graph-based result storage also allows retrospective comparisons as more genomes are sequenced or populations change, and is flexible, accommodating new analysis modules as they are developed. The database is available at `https://lfz.corefacility.ca/superphy/spfy/`.

2 FUNCTIONALITY

Spfy provides rapid in-silco versions of common reference laboratory tests for the analyses of *E. coli*. It supports the following *in-silico* subtyping options: serotyping, through both O- and H-antigen identification ectyper ref; Shiga-toxin 1 (Stx1), Shiga-toxin 2 (Stx2), and Intimin typing using Phylotyper [**?**], VF gene determination using ECTyper `https://github.com/phac-nml/ecoli_serotyping`, and AMR annotation using the RGI program [**?**].

Spfy also performs pangenome analyses using Panseq [**?**], and provides machine learning modules for biomarker discovery among groups using Scikit-learn [**?**].

Spfy handles all of the analyses tasks by dividing them into subtasks, which are subsequently distributed across a built-in task queue. Results are converted into individual graphs and stored within a larger graph

database according to standard ontologies list, where metadata including genotypes, biomarkers, host, source, and statistical significance testing of genome markers for user-defined groups are stored.

By integrating task distribution with graph storage, Spfy enables large-scale analyses, such as epidemiological association studies. Any data type or relation in the graph is a valid option for analysis. This means that genomes can be compared on the basis of the presence or absence of pan-genome regions, serotype, subtyping data, or provided metadata such as location or host-source.

3 IMPLEMENTATION

The server-side code for Spfy, graph generation, and analysis modules, are developed in Python, with the front-end website developed using the React JavaScript library `https://facebook.github.io/react/`. When new data is added to the database, the following steps are taken:

i) The upload begins through the website, where user-defined analyses options are selected. The results of these analyses are immediately reported to the user following their completion, while all other non-selected analyses are subsequently completed in the background and stored in the database without interaction from the user. The public web service accepts uploads of up to 200 MB (approximately 50 *E. coli* genomes uncompressed, or 120 genomes compressed) at a time, though an unlimited amount of data can be submitted to a local instance.

ii) User-selected analyses are enqueued into the Redis Queue `http://python-rq.org/` task queue. Redis Queue consists of a Redis Database `https://redis.io/` and task queue workers which run as Python processes.

iii) The workers dequeue the analyses, run them in parallel, and temporarily store results in the Redis database.

iv) Python functions parse the results and permanently store them in Blazegraph `https://www.blazegraph.com/`, the graph database used for Superphy.

3.1 Data Storage

Semantic web technologies describe the relationships between data [**?**], and graph databases for the storage of this information has been proposed as a open standard for sharing public information [**?**]. For biological data, individual data points can be a genome, contiguous DNA sequence, or gene, and these are linked together in a searchable graph structure using existing ontologies, including annotations from GenEpiO [**?**], FALDO [**?**], and TypOn [**?**]. This system is flexible and allows novel data to be incorporated into the existing graph.

The permanent storage of results is as a one-time cost to avoid recomputation when the same analysis is re-run. For analyses, Spfy searches the graph for all data points annotated with the queried ontology term. This graph data is then converted into the required structure, usually numerical arrays, or as required for the given analysis module. In a graph database, a search can begin at any node or attribute. This is in contrast to a SQL database which requires a predefined schema, or a NoSQL database which treats data as documents with varying structure. For example, the addition of a new analysis module would typically require a new table definition in a SQL database, or the addition of a new document type in a NoSQL database. With a graph database, new nodes or attributes are added and then connected to existing data, removing the need for explicit joins or data conversions. Additionally, data can be added to Spfy in parts, and the database will infer the correct connections between the data.

3.2 Web design

The front-end website is written as a single-page application. To ensure a familiar user interface, we followed the Material Design specification `https://material.io/`, published by Google, surrounding a card-based design. (see Figure 2) Both the task selection and result displays follow the same design pattern: while data storage is graph-based, the results of various analysis modules are presented to users in a familiar tabular structure and available for download as .csv spreadsheet files. (see Figure 3)

No account creation is required to use the platform. A sharable token is automatically created for users upon entering the website, and is embedded into the website address. Users can share results by copying their URL, and files submitted from different computers using the same token will be visible to anyone with the same link.

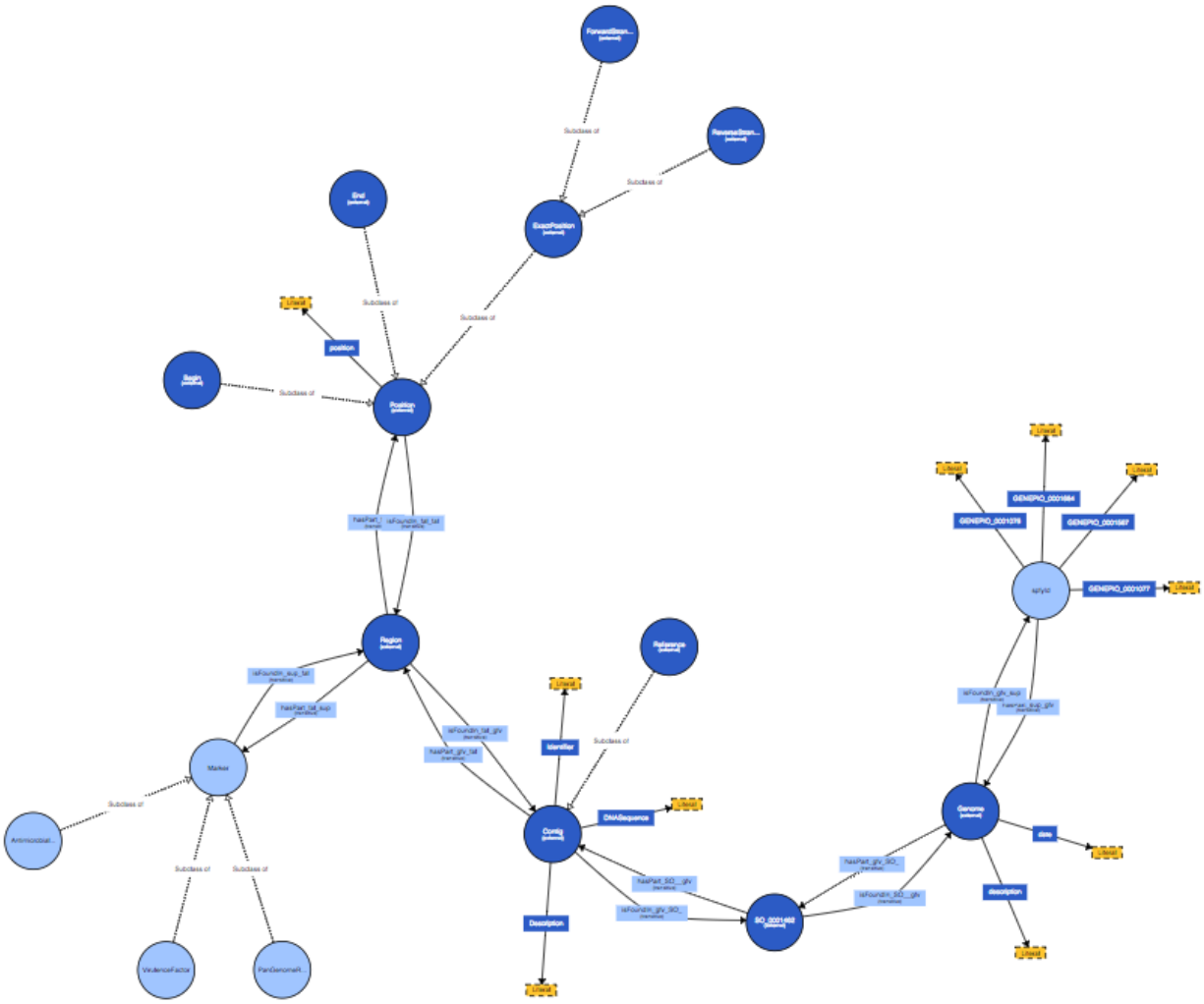

Figure 1: An example of how data is stored in Spfy. Brackets highlight the source of different data points and the software it was generated from. These parts are added in as the analyis modules complete, at varying times, and the overall connections are inferred by the database.

3.3 Service Virtualization

Docker `https://www.docker.com/` is a virtualization technology to simulate self-contained operating systems on the same host computer, without the overhead of full hardware virtualization [**?**]. The Spfy platform depends on a series of webservers, databases, and task workers, and uses Docker to compartmentalize these

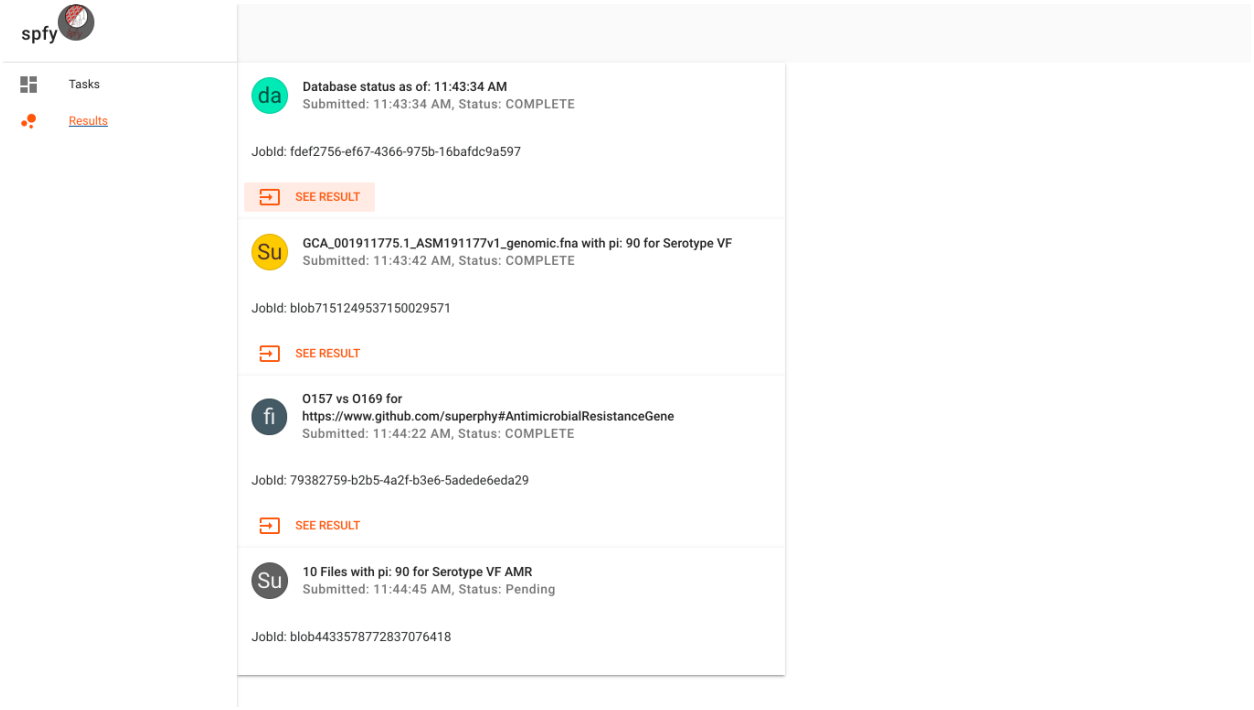

Figure 2: Caption for figure within column.

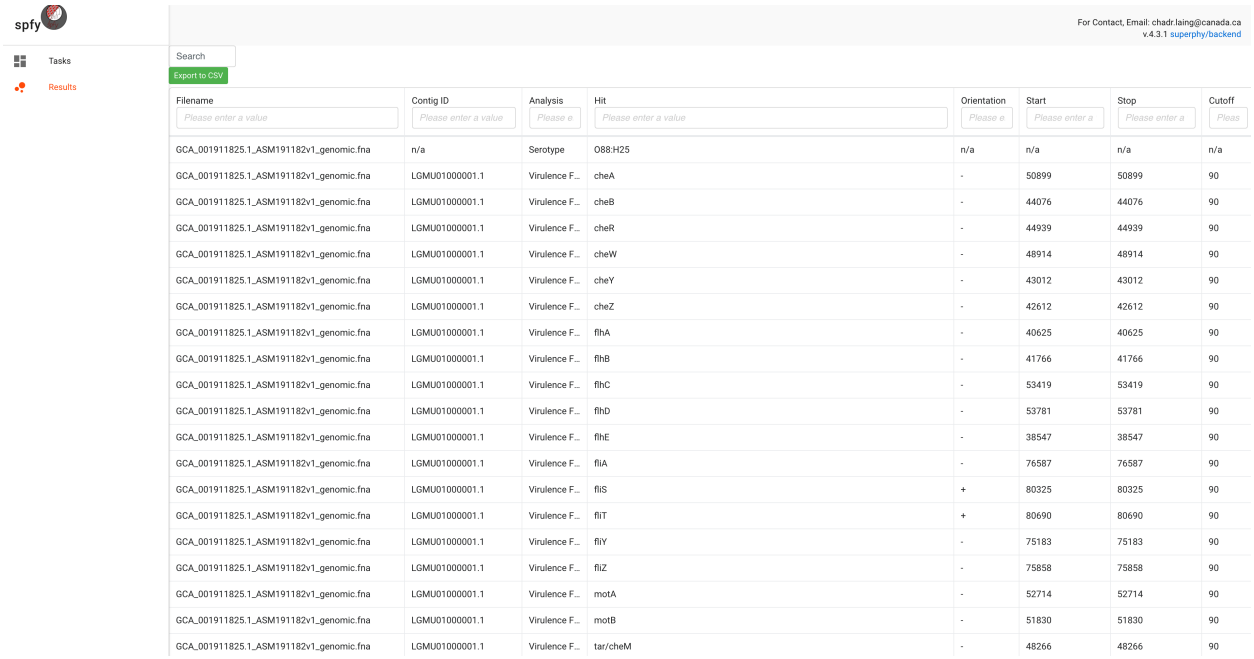

Figure 3: Caption for figure within column.

services, which are then networked together using Docker-Compose `https://docs.docker.com/compose/`. (see Figure 4) Docker integration ensures that software dependencies, which are typically manually installed [?, ?, ?, ?], are instead handled automatically.

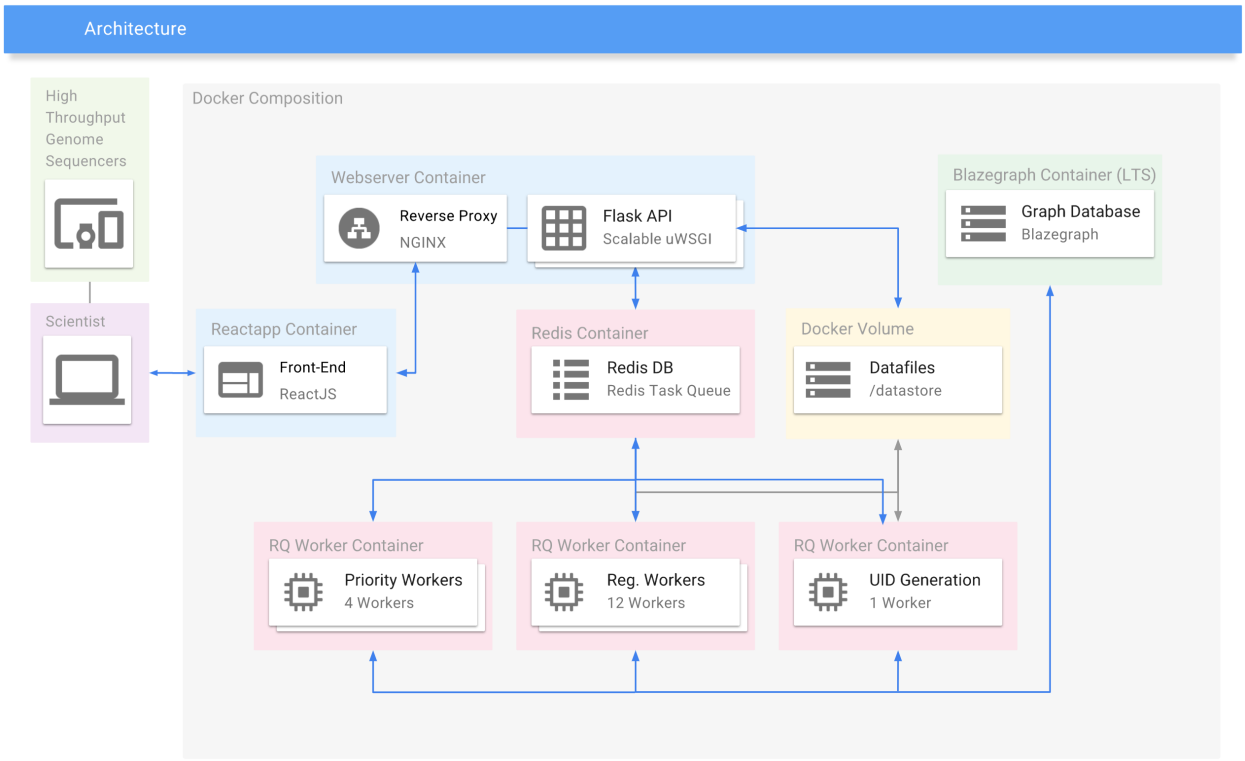

Figure 4: Caption for figure within column.

One of the key benefits of using common-place technologies is the compatibility with other infrastructure resources. Docker containers are widely supported by cloud computing services: Amazon Web Services (AWS) `https://aws.amazon.com/docker/`, Google Cloud Platform (GCloud) `https://cloud.google.com/container-engine/`, and Microsoft Azure `https://azure.microsoft.com/en-us/services/container-service/`, and self-hosted cloud computing technologies such as OpenStack `https://wiki.openstack.org/wiki/Docker`. Spfy packages compute nodes as reproducible Docker containers, and allows the platform to easily scale to demand.

3.4 Continuous integration

Our tests for functionality and backwards compatibility run on TravisCI `https://travisci.io`, a continuous integration (CI) platform. The individual tests use PyTest `https://doc.pytest.org/`, and the current build status can be checked either on our GitHub repository or at `https://travis-ci.org/superphy/backend`. TravisCI also builds the core Docker images for Spfy, and uploads them to Docker Hub `https://hub.docker.com/u/superphy/`.

4 RESULTS

Spfy was tested with 10,243 public *E. coli* assembled genomes from Enterobase, storing every sequence and the results for all included analysis modules. This included: serotypeing (O-antigen, H-antigen), toxin subtyping (Shiga-toxin 1, Shiga-toxin 2, and Intimin), the identification of VF and AMR determinants, and

determination of pan-genome content *E. coli*. The resulting database had 17,820 nodes and 3,811,473 leaves, with 1,125,909,074 object properties. Spfy has been up since May 2017. The server accepts assembled *E. coli* genomes with the *.fasta* or *.fna* extensions. Submissions are subjected to quality control, ensuring the submitted genomes are *E. coli* sequences before subsequent analyses are run.

Table 1: My caption

name	indexType	m	height	nnodes	nleaves	nentries	nodeBytes	leafBytes	totalBytes
__globalRowStore	BTree	32	1	1	6	102	193	8537	8730
kb.lex.BLOBS	BTree	692	2	17	6727	3227686	90596	60187016	89331537662
kb.lex.ID2TERM	BTree	905	2	88	39912	18080577	515355	293436043	2058798455
kb.lex.TERM2ID	BTree	193	3	1153	147978	18080577	5596557	1152764154	1158360711
kb.spo.JUST	BTree	284	3	13213	2042532	299426518	77979362	15527483178	15605462540
kb.spo.OSP	BTree	708	3	1649	639325	262364538	11448125	3760927987	3772376112
kb.spo.POS	BTree	990	2	864	463188	262364538	8347594	2246478879	2254826473
kb.spo.SPO	BTree	1024	2	835	471805	262364538	10308603	2661857997	2672166600

5 DISCUSSION

Many bioinformatics software programs have been developed *ad hoc*, with individual researchers and laboratories developing software specific to their environment [**?**]. Such tools were often script-based, with custom data formats, and only suitable for small collections of data [**?**]. Recent efforts [**?**, **?**] have focused on providing a common web interface for these programs, while still returning the same result files. However, many subsets of biology now require the analyses of big-data, where inputs are taken from a variety of analysis programs, and involve large-scale data warehousing [**?**]. The ability to perform computations in real-time, store data in flexible databases, and utilize a common application programming interface (API) linking resources are now required for types of analyses that need to be performed [**?**].

One of the key goals in developing Spfy was to accommodate and store a variety of result formats, and then to make the data from these results retrievable and usable as inputs for downstream analyses, such as predictive biomarker discovery. We have shown how a graph database can accommodate the results given by a variety of bioinformatics programs, and how Spfy is performant for data retrieval on the results for multiple analyses among over 10,000 genomes, providing results from big-data comparisons in the same efficiency as old analyses on single files.

5.1 Impact on Public Health Efforts

The isolation and characterization of bacterial pathogens are critical for Public Health laboratories to rapidly respond to outbreaks, and to effectively monitor known and emerging pathogens through surveillance programs. Until recently, public-health agencies relied on laboratory tests such as serotyping, pulsed-field gel electrophoresis (PFGE), PCR-based amplification of known VFs, and disc-diffusion assays to identify AMR, for the characterization of bacterial isolates in outbreak, surveillance, and reference laboratory settings [**?**].

Current efforts are focused on predictive genomics, where the relevant phenotypic information can be determined through examination of the whole-genome sequence without need for the traditional laboratory test. Spfy provides rapid and easy predictive genomic analyses of *E. coli* genomes.

5.2 Comparison with other bioinformatic pipeline technologies

The automated analyses of WGS is currently facilitated by existing scientific workflow technologies such as Galaxy [**?**], the Integrated Rapid Infectious Disease Analysis (IRIDA) platform `http://www.irida.ca/`, and the Bacterium Analysis Pipeline (BAP) [**?**]. Galaxy aims to provide a reproducible, computation-based research environment which is accessible to individuals without programming knowledge. Galaxy defines a formal schema for linking different analysis software together, so the entire analysis pipeline can be replicated. IRIDA is built on top of Galaxy and adds prebuilt pipelines for specific analyses, as well as sequence and result storage. IRIDA takes a project-based approach, with sequences stored per project, and results stored per project. Similarly, BAP provides an integrated analysis

pipeline for bacterial WGS data as a web service. These workflow technologies store their results using relational tables [**?**], or as files on disk [**?**].

Spfy is similar to these technologies in that it automates workflows for users, and like Galaxy, uses task queues to distribute selected analysis. On a per file basis, Spfy performs at a similar speed to BAP on predictive genomics tasks, though Spfy does not provide genome assembly services. Spfy processes XXXX files over XX tasks in XXXX time, distributing computations over a task queue and multiple Docker compartmentalized containers. However, unlike these workflow managers, Spfy is designed to help solve the needless recomputation of analyses by storing results in a graph database for future use. This allows Spfy to perform population-wide analyses, regardless of the individual analysis software used to generate the results for an individual genome.

6 CONCLUSIONS

Future work will focus on adding additional analyses modules to aid genotype to phenotype predictions using machine learning modules, and supporting bacterial species such as *Salmonella*, and *Campylobater*. While the integrated approach of storing and retrieving results provides enormous benefits, the developed analyses modules are self-contained and be transfered to existing platforms such as Galaxy, and IRIDA. The source code for Spfy is hosted at `https://github.com/superphy/backend`, and is available for free under the open-source Apache 2.0 license. A developer guide is provided at `https://superphy.readthedocs.io/en/latest/`.

Conflict of interest. None declared.

7 ACKNOWLEDGEMENTS

8 APPENDIX