

Spfy: an integrated graph database for real-time prediction of *Escherichia coli* phenotypes and downstream comparative analyses

Kevin K Le^{*1}, Matthew D Whiteside¹, James Hopkins¹, Victor PJ Gannon¹ and Chad R Laing^{†1}

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

April 18, 2018

Abstract

Public health laboratories are currently moving to whole-genome sequence (WGS) based analyses, and require the rapid prediction of standard reference laboratory methods based solely on genomic data. Currently, these predictive genomics tasks rely on workflows that chain together multiple programs for the requisite analyses. While useful, these systems do not store the analyses in a genome-centric way, meaning the same analyses are often re-computed for the same genomes.

To solve this problem, we created Spfy, which uses a graph database to store and retrieve results from computational workflows, and link data to individual genomes using standardized ontologies.

The Spfy platform facilitates rapid phenotype identification, as well as the efficient storage and downstream comparative analysis of tens of thousands of genome sequences. Though generally applicable to bacterial genome sequences, Spfy currently contains 10,243 *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. Additionally, the presence / absence of then entire *E. coli* pan-genome was computed and linked to each genome.

Owing to its vast database of pre-computed results, and the ability to easily incorporate user data, Spfy facilitates hypothesis testing in fields ranging from population genomics to epidemiology, while mitigating the re-computation of analyses.

The graph approach of Spfy is flexible, and can accommodate new analysis software modules as they are developed, easily linking new results to those already stored. Spfy provides a database and analyses approach for *E. coli* that is able to match the rapid accumulation of WGS in public databases.

Database URL: <https://lfz.corefacility.ca/superphy/spfy/>.

1 Introduction

Whole genome sequencing (WGS) can provide the entire genetic content of an organism. This unparalleled resolution and sensitivity has recently transformed public-health surveillance and outbreak response [1, 2]. Additionally, the identification of novel disease mechanisms [3, 4], and rapid clinical diagnoses and reference lab tests are now possible. [5, 6].

The rapid characterization based on WGS relies on the outputs from multiple software programs that are targeted for specific applications. Examples include the identification of known antimicrobial resistance (AMR) genes, through software such as the Resistance Gene Identifier (RGI) [7], [8], [9], and ARIBA [10]; or the identification of known virulence factor genes (VF) through software such as VirulenceFinder [8], SRST2 [11], and GeneSippr [12]. For clinical diagnoses and comparisons, individual species can be first divided into subtypes with complementing AMR and VF results. Software methods for subtyping rely on

^{*}kevin.le@canada.ca

[†]chad.laing@canada.ca

pre-selected intraspecies genes or genomic regions, and are targeted through software such as Phylotyper [13], SerotypeFinder [14], the EcOH dataset applied through SRST2 [15], and V-Typer [16]. These methods represent *in-silico* analogues of traditional wet-lab tests, 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 analyses programs into a cohesive whole also exist. These include free platforms such as the Bacterium Analysis Pipeline (BAP) [17], and the Pathosystems Resource Integration Center (PATRIC) [18]. 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 [19]. These platforms are designed to be applied to individual projects.

Many of the analyses that are currently used in the characterization and study of bacterial genomes are broadly useful, and therefore the same analyses are often computed multiple times for the same genome. An effective method to mitigate this re-computation, 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 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 [20], 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, which the original SuperPhy platform was incapable of.


Here, we present a new platform merging Superphy’s pre-computed results with a novel data storage and processing architecture, which we call Spfy; Spfy integrates a graph database with real-time analyses to avoid recomputing identical results. Graph-based result storage also allows retrospective comparisons across stored results 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-silico* 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 using ECTyper https://github.com/phac-nml/ecoli_serotyping as well as VF gene determination; Shiga-toxin 1 (Stx1), Shiga-toxin 2 (Stx2), and Intimin typing using Phylotyper [13]; and AMR annotation using the resistance gene identifier from the comprehensive antibiotic resistance database [7]. An example of the VF results is given in Figure 1. Spfy also performs pan-genome analyses using Panseq [21], with the entire pan-genome stored in the database and associated with source genomes.

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 the previously created ontologies GenEpiO [22], FALDO [23], and TypOn [24], where metadata including genotypes, biomarkers, host, source 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



For Contact, Email: chadi.laing@canada.ca
 v4.3.1 superphy/backend

Tasks

Results

[Export to CSV](#)

| Filename | Contig ID | Analysis | Hit | Orientation | Start | Stop | Cutoff |
|---|-------------------------------------|-------------------------|-------------------------------------|-------------------------|-------------------------------|-------------------------------|-----------------------|
| <small>Please enter a value</small> | <small>Please enter a value</small> | <small>Please e</small> | <small>Please enter a value</small> | <small>Please e</small> | <small>Please enter a</small> | <small>Please enter a</small> | <small>Please</small> |
| GCA_001911825.1_ASM191182v1_genomic.fna | n/a | Serotype | O88:H25 | n/a | n/a | n/a | n/a |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | cheA | - | 50899 | 50899 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | cheB | - | 44076 | 44076 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | cheR | - | 44939 | 44939 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | cheW | - | 48914 | 48914 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | cheY | - | 43012 | 43012 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | cheZ | - | 42612 | 42612 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flhA | - | 40625 | 40625 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flhB | - | 41766 | 41766 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flhC | - | 53419 | 53419 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flhD | - | 53781 | 53781 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flhE | - | 38547 | 38547 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flhA | - | 76587 | 76587 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flis | + | 80325 | 80325 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flIT | + | 80690 | 80690 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flIY | - | 75183 | 75183 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | flIZ | - | 75858 | 75858 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | motA | - | 52714 | 52714 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | motB | - | 51830 | 51830 | 90 |
| GCA_001911825.1_ASM191182v1_genomic.fna | LGMU01000001.1 | Virulence F... | tar/cheM | - | 48266 | 48266 | 90 |

Figure 1: Detailed results for the virulence factor subtyping task. While data storage in Spfy is graph-based, a familiar tabular structure is presented to users. The result can also be shared using the URL with the embedded token.

data, or provided metadata such as location or host-source. All results are displayed to users in real-time, usually within 2-3 minutes. For example, Spfy can determine if there's a statistically significant difference, using Fisher's exact test, in the presence of all known AMR genes in collected O157 *E. coli* genomes vs. O26 as shown in Figure 2. Different types can also be joined into a group through logical connectives AND, OR, or Negation. This approach can be used to compare any data regardless of source software.

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, 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 with 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 Spfy.

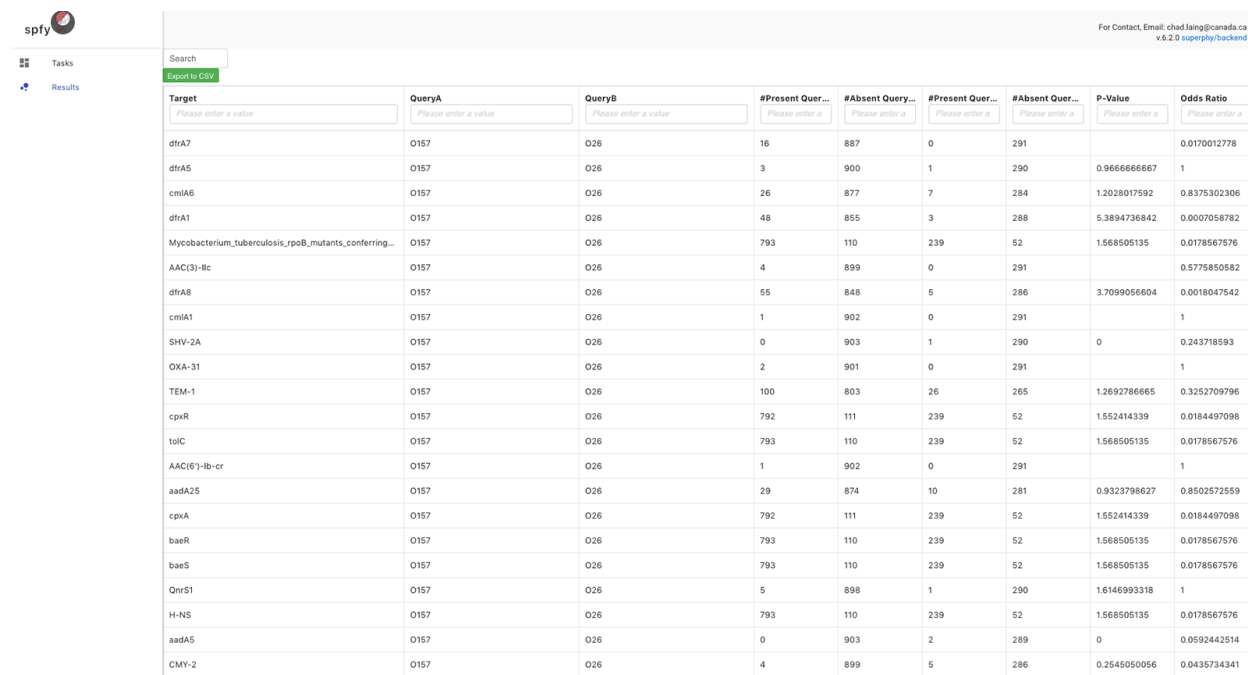
3.1 Data Storage

Semantic web technologies describe the relationships between data, and have been proposed as an open standard for sharing public information [25], while graph databases are a flexible means of storing this information [26]. Biological data can be a genome, contiguous DNA sequence, or gene, and these are linked together in a searchable graph structure using existing ontologies. 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, which avoids re-computation when the same analysis is re-run. During 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, 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.

Spfy primarily uses Blazegraph <https://github.com/blazegraph/database> for storage along with MongoDB to cache a hash table for duplicate checking. The cache allows Spfy to more efficiently check for duplicate files in Blazegraph than would be possible through a search of the graph structure.



| Target | QueryA | QueryB | #Present Quer... | #Absent Query... | #Present Quer... | #Absent Quer... | P-Value | Odds Ratio |
|---|--------|--------|------------------|------------------|------------------|-----------------|--------------|--------------|
| dfrA7 | O157 | O26 | 16 | 887 | 0 | 291 | | 0.0170012778 |
| dfrA5 | O157 | O26 | 3 | 900 | 1 | 290 | 0.9666666667 | 1 |
| cmIA6 | O157 | O26 | 26 | 877 | 7 | 284 | 1.2028017592 | 0.8375302306 |
| dfrA1 | O157 | O26 | 48 | 855 | 3 | 288 | 5.3894736842 | 0.0007056782 |
| Mycobacterium.tuberculosis_rpoB_mutants_conferring... | O157 | O26 | 793 | 110 | 239 | 52 | 1.568505135 | 0.0178567576 |
| AAC(3)-IIC | O157 | O26 | 4 | 899 | 0 | 291 | | 0.5775850582 |
| dfrA8 | O157 | O26 | 55 | 848 | 5 | 286 | 3.7099056604 | 0.0018047542 |
| cmIA1 | O157 | O26 | 1 | 902 | 0 | 291 | | 1 |
| SHV-2A | O157 | O26 | 0 | 903 | 1 | 290 | 0 | 0.243718593 |
| OXA-31 | O157 | O26 | 2 | 901 | 0 | 291 | | 1 |
| TEM-1 | O157 | O26 | 100 | 803 | 26 | 265 | 1.2692786665 | 0.3252709796 |
| cpvR | O157 | O26 | 792 | 111 | 239 | 52 | 1.552414339 | 0.0184497098 |
| tolC | O157 | O26 | 793 | 110 | 239 | 52 | 1.568505135 | 0.0178567576 |
| AAC(6)-Ib-cr | O157 | O26 | 1 | 902 | 0 | 291 | | 1 |
| aadA25 | O157 | O26 | 29 | 874 | 10 | 281 | 0.9323798627 | 0.8502572559 |
| cpvA | O157 | O26 | 792 | 111 | 239 | 52 | 1.552414339 | 0.0184497098 |
| baeR | O157 | O26 | 793 | 110 | 239 | 52 | 1.568505135 | 0.0178567576 |
| baeS | O157 | O26 | 793 | 110 | 239 | 52 | 1.568505135 | 0.0178567576 |
| QnrS1 | O157 | O26 | 5 | 898 | 1 | 290 | 1.6146993318 | 1 |
| H-NS | O157 | O26 | 793 | 110 | 239 | 52 | 1.568505135 | 0.0178567576 |
| aadA5 | O157 | O26 | 0 | 903 | 2 | 289 | 0 | 0.0592442514 |
| CMY-2 | O157 | O26 | 4 | 899 | 5 | 286 | 0.2545050056 | 0.0435734341 |

Figure 2: Genomes with the predicted subtype O157 are compared against the subtype O26. In this example, there are 903 O157 genomes and 291 O26 genomes which are compared for the presence of 129 AMR genes. Fisher's Exact Test is used for the comparison.

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, built around a card-based design (Figure 4).

Both the task selection and result displays follow the same design pattern, where data storage is graph-based, but the results of analysis modules are presented to users in a familiar tabular structure and available for download as .csv spreadsheet files, as shown in Figure 1.

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.

Ontology

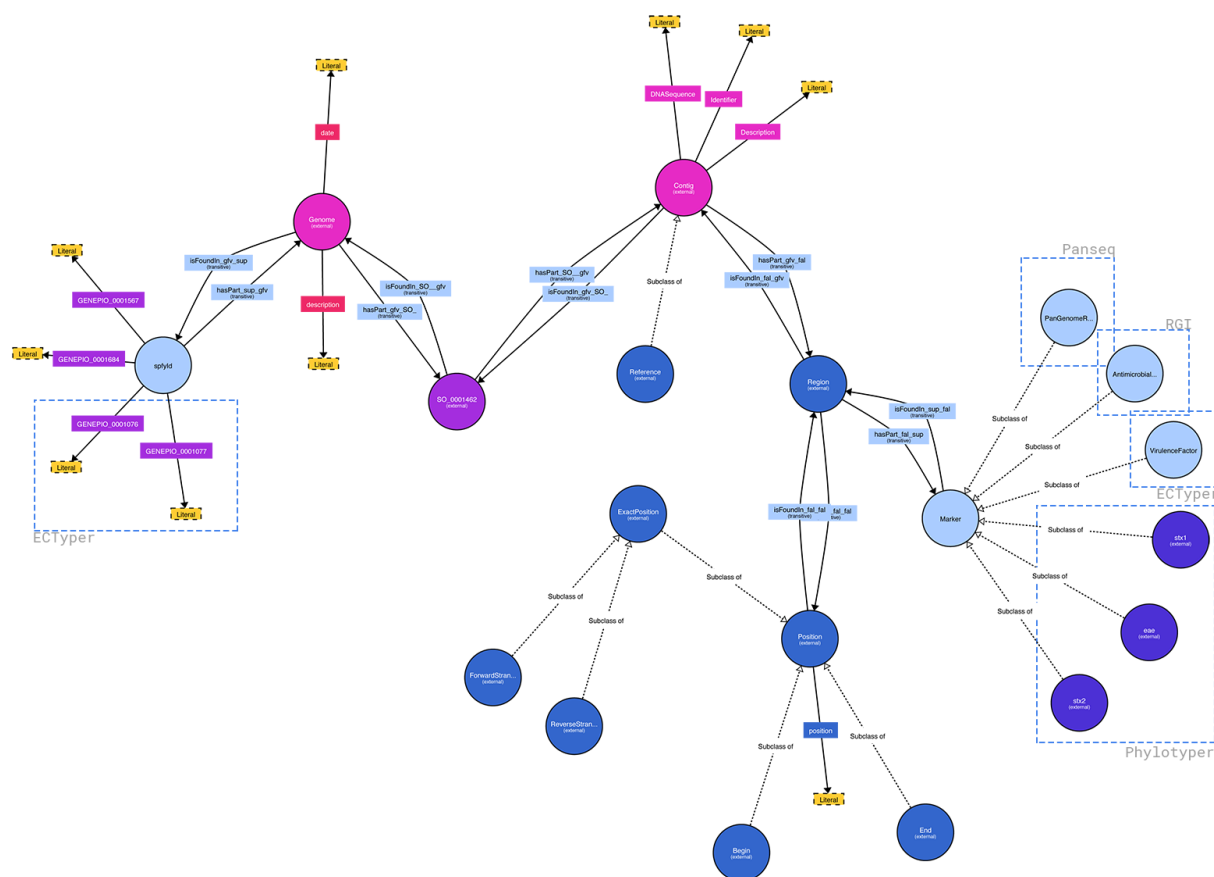


Figure 3: Structure of the Spfy graph database. Brackets highlight the source of different data points and the software it was generated from. Data are added in as the analysis modules complete, at varying times, and the overall connections are inferred by the database. Non-bracketed sections are sourced from the uploaded genome files or user-supplied metadata.

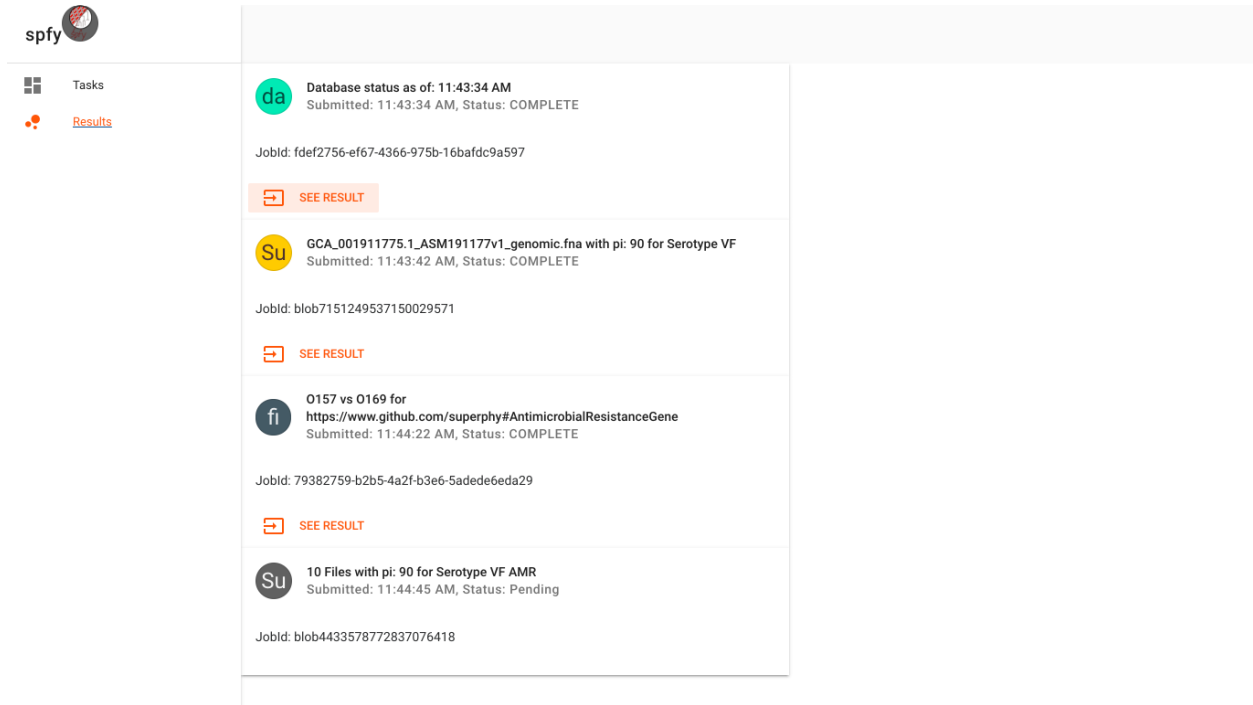


Figure 4: The results interface for submitted tasks. Cards represent individual tasks and the entire collection can be accessed by sharing the URL with the embedded token.

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 [27]. The Spfy platform depends on a series of web servers, databases, and task workers, and uses Docker to compartmentalize these services, which are then networked together using Docker-Compose <https://docs.docker.com/compose/>. (see Figure 5) Docker integration ensures that software dependencies, which are typically manually installed [28, 21, 11, 29], are instead handled automatically.

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

Tests for functionality and backwards compatibility are 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 on GitHub 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: serotyping (O-antigen, H-antigen), toxin subtyping (Shiga-toxin 1, Shiga-toxin 2, and Intimin), the identification of VF and AMR determinants, and determination of the pan-genome content of *E. coli*. The resulting database has 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.

Spfy is running on a virtual machine (VM) with 8 vCPUs emulating single-core Intel Xeon E3-series processors, and with 32 GB of RAM. The VM is running CentOS version 7.3, with Docker version 17.06.1-ce and Docker-Compose version 1.12.0. On comparison tasks, Spfy can retrieve and compare 1 million nodes/attributes in the graph database in approximately 70 seconds. As shown in Figure 6, performance scales linearly with the number of genomes involved in the comparison, or the number of target nodes retrieved per genome; 1.5 million nodes/attributes can be compared in approximately 90 seconds, and 2 million nodes/attributes in approximately 110 seconds.

On analysis tasks, Spfy runs all included modules in an average of 130 seconds per genome, as shown in Figure 7. Spfy can also queue batches of genomes for analysis, decreasing the average runtime to an average of 54 seconds per file due to parallelization of analysis runs 8. In total, the platform analyzed 50 genomes in 45 minutes, and 100 genomes in 89 minutes.

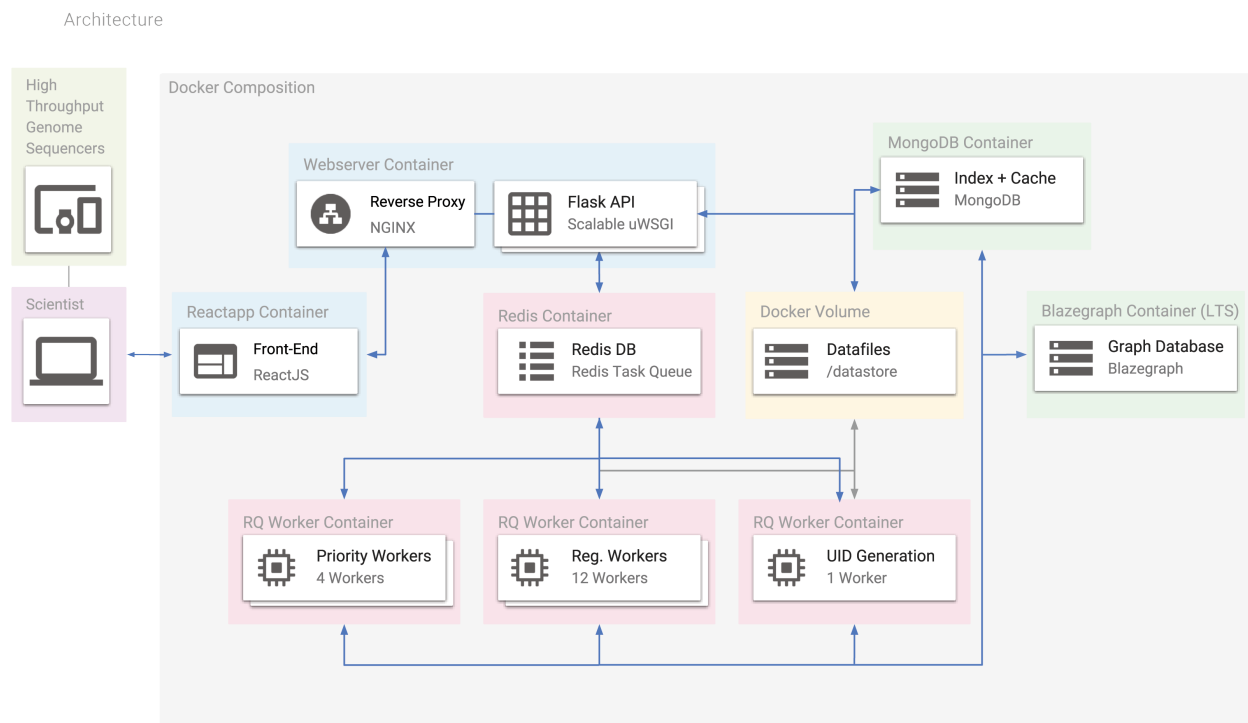


Figure 5: The Docker containers used in Spfy. Arrows represent the connections between different containers, and the entire platform can be recreated with a single command using its Docker-Compose definition.



Figure 6: Runtimes of Fisher’s Exact Test depending on the number of nodes/attributes involved in the comparison. Left: Runtimes as the number of genomes involved increases for a fixed (107) number of targets per genome. Center: Runtimes as the number of targets involved increased for a fixed (116) number of genomes per target. Right: Runtimes as the total number of targets retrieved increases; the total number of targets is calculated as follows: (Number Genomes Group A + Number Genomes Group B) x Number Targets per Genome. In all cases, we see a linear increase in runtime as the number of targets or the number of genomes involved increases.

5 DISCUSSION

Many bioinformatics software programs have been developed *ad hoc*, with individual researchers and laboratories developing software specific to their environment [30]. Such tools were often script-based, with custom data formats, and only suitable for small collections of data [30]. Recent efforts [31, 17] 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 [32]. The ability to integrate data from different source technologies, merge submissions from other labs, and distribute computations over fault-tolerant systems are now required [32].

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 from 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, for the characterization of bacterial isolates in outbreak, surveillance, and reference laboratory settings [1]. 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 tests.

Spfy provides rapid and easy predictive genomic analyses of *E. coli* genomes while also addressing the problem of large scale comparisons. With the larger datasets involved in population genomics, it is no longer viable for individual researchers to download data to perform comparisons. Instead, efforts have focused

on storing biological data online and enabling analyses of those data [32]. By using a graph database, Spfy integrates results from different technologies, as well as laboratory results and user-submitted metadata. In addition, datasets can be built and submitted from multiple labs for joint analyses.

5.2 Comparison with other bioinformatics pipeline technologies

The automated analyses of WGS is currently facilitated by existing scientific workflow technologies such as Galaxy [31]. Galaxy aims to provide a reproducible, computational interface which is accessible to

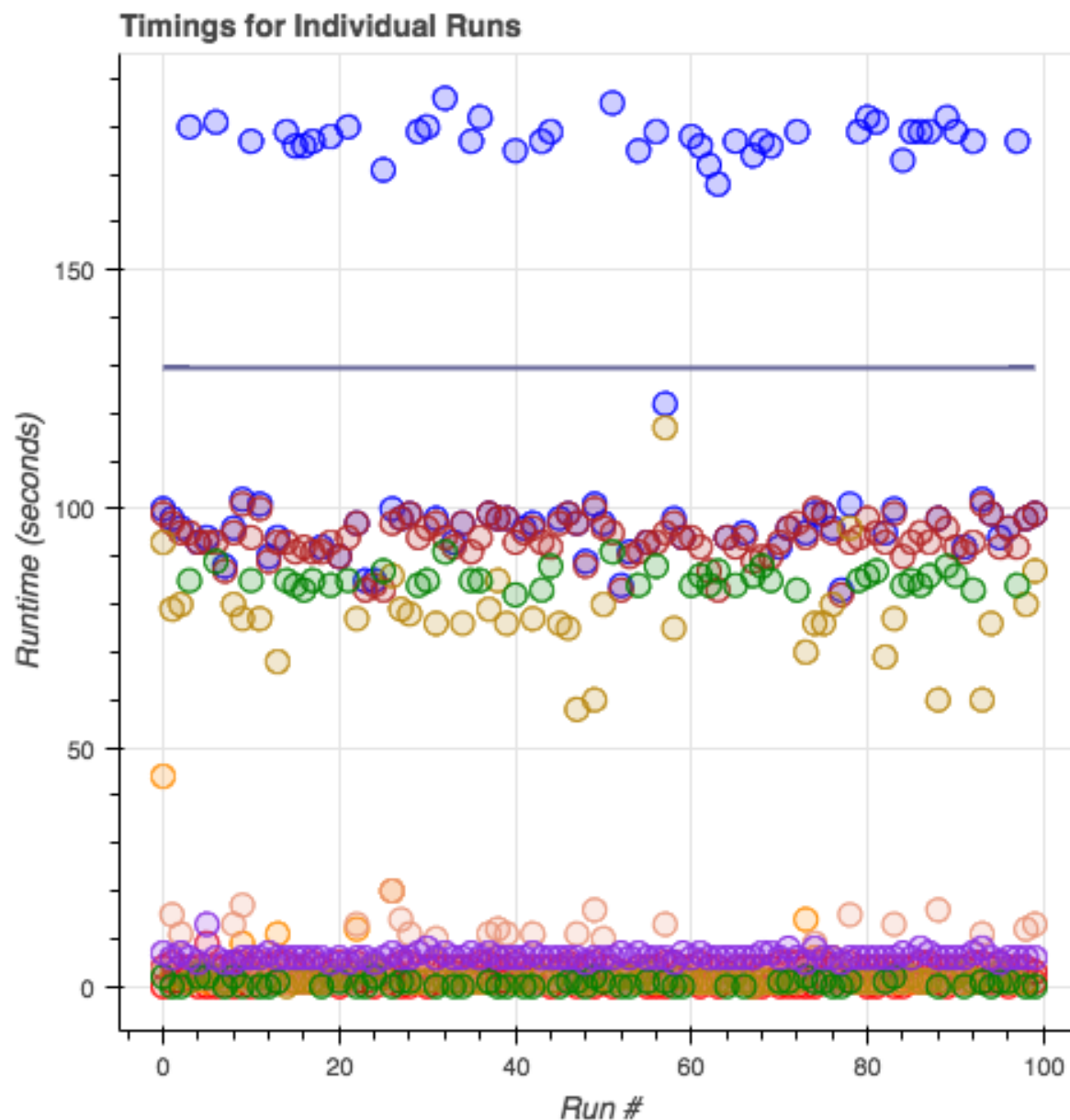


Figure 7: Runtimes of Spfy’s analysis modules. 100 genome files were run one-by-one with the different colors representing the runtimes of the different analysis module; The blue line indicates the average actual runtime per file after accounting for parallelization.

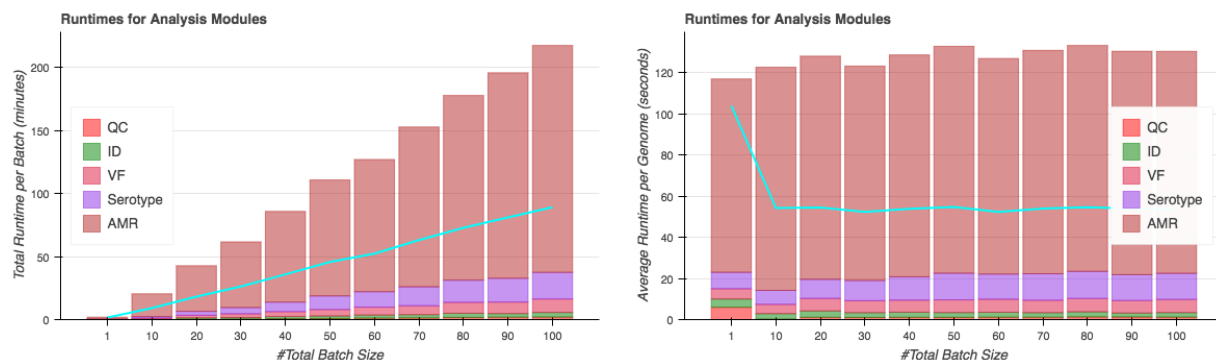


Figure 8: Runtimes of Spfy’s analysis modules for batches of files. Left: Total runtime for analyzing all files in the given batch. The blue line indicates the actual time to completion after accounting for parallelization; 50 files are analyzed in 45 minutes and 100 files in 89 minutes. Right: Average runtime per file in the batch. As indicated by the line, the average runtime per genome is constant when running more than one file. This is due to the degree of parallelization reaching the capacity of the host computer.

individuals without programming knowledge. Galaxy defines a formal schema for linking different analysis software together, so the entire pipeline can be replicated and also extended as new tools are developed. The Galaxy workflow focuses on running an individual analysis pipeline. It does not include functionality to store and collate analysis results for large-scale comparative studies.

The Bacterium Analysis Pipeline (BAP) [17] provides an integrated analysis pipeline for bacterial WGS data as a web service. It provides an individual per-genome report of the determined species, multilocus sequence type, VF and AMR genes [17].

Spfy is similar to these technologies in that it automates workflows for users and uses task queues to distribute selected analyses. 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. After accounting for assembly services, BAP reported [17] an average runtime of 8-9 minutes per genome over 476 runs which is in the same scale as Spfy’s average of over 2 minutes. Note that the two platforms are now directly comparable, due to the differences in analysis tasks involved. In similar tasks such as AMR determination, the ResFinder program included in BAP took an average of 3-4 minutes [17] and is similar to the RGI program included in Spfy, which took an average of 1 minute 30 seconds.

However, unlike these workflow managers, Spfy is designed to help solve the needless re-computation of analyses by storing results in a graph database for downstream comparative studies 1. This allows Spfy to perform population-wide analyses, regardless of the individual software used to generate the results for a given genome.

Table 1: Comparison of four bioinformatic pipelines and their underlying database. Functionally, Spfy integrates different analysis modules as in BAP while also merging large datasets as in PATRIC.

| | Spfy | Galaxy | BAP | PATRIC |
|----------|---------------------|---------------------|---------------------|---------------------|
| Database | Blazegraph | PostgreSQL | MySQL + File System | MongoDB + Shock |
| Type | Graph | SQL | SQL + File System | NoSQL |
| Focus | Integrated Analyses | Workflow Technology | Batch Analysis | Integrated Analyses |

PATRIC [18] and Spfy share the same goal of integrated analyses. PATRIC has support for comparing up to nine user-submitted genomes against a reference genome, based on gene annotations; the platform indexes a NoSQL document store to compare similar document types. Unlike PATRIC, Spfy provides the ability to perform statistical comparisons of any permutation of a population group based on the chosen data types, and Spfy’s graph database has no limit on the number of genomes grouped for comparison.

6 CONCLUSIONS

The integrated approach taken in the creation of Spfy, where the storage and retrieval of results provides enormous benefits for the large-scale analyses of *E. coli*. The developed analyses modules are also self-contained and be transferred to existing platforms such as Galaxy. Future work will focus on adding machine learning modules to improve genotype / phenotype predictions, and supporting bacterial species such as *Salmonella*, and *Campylobacter*. 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

References

- [1] J Ronholm, Neda Nasheri, Nicholas Petronella, and Franco Pagotto. Navigating microbiological food safety in the era of whole-genome sequencing. *Clinical Microbiology Reviews*, 29(4):837–857, 2016.
- [2] Birgitta Lytsy, Lars Engstrand, Åke Gustafsson, and Rene Kaden. Time to review the gold standard for genotyping vancomycin-resistant enterococci in epidemiology: Comparing whole-genome sequencing with pfge and mlst in three suspected outbreaks in sweden during 2013–2015. *Infection, Genetics and Evolution*, 2017.
- [3] Kai Wang, Siu Tsan Yuen, Jiangchun Xu, Siu Po Lee, Helen HN Yan, Stephanie T Shi, Hoi Cheong Siu, Shibing Deng, Kent Man Chu, Simon Law, et al. Whole-genome sequencing and comprehensive molecular profiling identify new driver mutations in gastric cancer. *Nature genetics*, 46(6):573–582, 2014.
- [4] Ryan KC Yuen, Bhooma Thiruvahindrapuram, Daniele Merico, Susan Walker, Kristiina Tammimies, Ny Hoang, Christina Chrysler, Thomas Nalpathamkalam, Giovanna Pellecchia, Yi Liu, et al. Whole-genome sequencing of quartet families with autism spectrum disorder. *Nature medicine*, 21(2):185–191, 2015.
- [5] Laurel K Willig, Josh E Petrikin, Laurie D Smith, Carol J Saunders, Isabelle Thiffault, Neil A Miller, Sarah E Soden, Julie A Cakici, Suzanne M Herd, Greyson Twist, et al. Whole-genome sequencing for identification of mendelian disorders in critically ill infants: a retrospective analysis of diagnostic and clinical findings. *The Lancet Respiratory Medicine*, 3(5):377–387, 2015.
- [6] Frederick E Dewey, Megan E Grove, Cuiping Pan, Benjamin A Goldstein, Jonathan A Bernstein, Hassan Chaib, Jason D Merker, Rachel L Goldfeder, Gregory M Enns, Sean P David, et al. Clinical interpretation and implications of whole-genome sequencing. *Jama*, 311(10):1035–1045, 2014.
- [7] Andrew G McArthur, Nicholas Waglechner, Fazmin Nizam, Austin Yan, Marisa A Azad, Alison J Baylay, Kirandeep Bhullar, Marc J Canova, Gianfranco De Pascale, Linda Ejim, et al. The comprehensive antibiotic resistance database. *Antimicrobial agents and chemotherapy*, 57(7):3348–3357, 2013.
- [8] Kortine Annina Kleinheinz, Katrine Grimstrup Joensen, and Mette Voldby Larsen. Applying the resfinder and virulencefinder web-services for easy identification of acquired antibiotic resistance and e. coli virulence genes in bacteriophage and prophage nucleotide sequences. *Bacteriophage*, 4(2):e27943, 2014.
- [9] Sushim Kumar Gupta, Babu Roshan Padmanabhan, Seydina M Diene, Rafael Lopez-Rojas, Marie Kempf, Luce Landraud, and Jean-Marc Rolain. Arg-annot, a new bioinformatic tool to discover antibiotic resistance genes in bacterial genomes. *Antimicrobial agents and chemotherapy*, 58(1):212–220, 2014.
- [10] Martin Hunt, Alison E Mather, Leonor Sánchez-Busó, Andrew J Page, Julian Parkhill, Jacqueline A Keane, and Simon R Harris. Ariba: rapid antimicrobial resistance genotyping directly from sequencing reads. *Microbial genomics*, 3(10), 2017.
- [11] Michael Inouye, Harriet Dashnow, Lesley-Ann Raven, Mark B Schultz, Bernard J Pope, Takehiro Tomita, Justin Zobel, and Kathryn E Holt. Srst2: Rapid genomic surveillance for public health and hospital microbiology labs. *Genome medicine*, 6(11):90, 2014.
- [12] Dominic Lambert, Catherine D Carrillo, Adam G Koziol, Paul Manninger, and Burton W Blais. Genesippr: a rapid whole-genome approach for the identification and characterization of foodborne pathogens such as priority shiga toxigenic escherichia coli. *PLoS One*, 10(4):e0122928, 2015.
- [13] Matthew D Whiteside, Chad R Laing, and Victor PJ Gannon. Phylotyper: in silico predictor of gene subtypes. *Bioinformatics*, 2017.
- [14] Katrine G Joensen, Anna MM Tetzschner, Atsushi Iguchi, Frank M Aarestrup, and Flemming Scheutz. Rapid and easy in silico serotyping of escherichia coli using whole genome sequencing (wgs) data. *Journal of clinical microbiology*, pages JCM–00008, 2015.
- [15] Danielle J Ingle, Mary Valcanis, Alex Kuzevski, Marija Tauschek, Michael Inouye, Tim Stinear, Myron M Levine, Roy M Robins-Browne, and Kathryn E Holt. In silico serotyping of e. coli from short read data identifies limited novel o-loci but extensive diversity of o: H serotype combinations within and between pathogenic lineages. *Microbial genomics*, 2(7), 2016.
- [16] Catherine D Carrillo, Adam G Koziol, Amit Mathews, Noriko Goji, Dominic Lambert, George Huszcynski, Martine Gauthier, Kingsley Amoako, and Burton W Blais. Comparative evaluation of genomic and laboratory approaches for determination of shiga toxin subtypes in escherichia coli. *Journal of food protection*, 79(12):2078–2085, 2016.
- [17] Martin Christen Frølund Thomsen, Johanne Ahrenfeldt, Jose Luis Bellod Cisneros, Vanessa Jurtz, Mette Voldby Larsen, Henrik Hasman, Frank Møller Aarestrup, and Ole Lund. A bacterial analysis platform: an integrated system for analysing bacterial whole genome sequencing data for clinical diagnostics and surveillance. *PloS one*, 11(6):e0157718, 2016.

- [18] Alice R Wattam, James J Davis, Rida Assaf, Sébastien Boisvert, Thomas Brettin, Christopher Bun, Neal Conrad, Emily M Dietrich, Terry Disz, Joseph L Gabbard, et al. Improvements to patric, the all-bacterial bioinformatics database and analysis resource center. *Nucleic acids research*, 45(D1):D535–D542, 2016.
- [19] Bala Swaminathan, Timothy J Barrett, Susan B Hunter, Robert V Tauxe, and CDC PulseNet Task Force. Pulsenet: the molecular subtyping network for foodborne bacterial disease surveillance, united states. *Emerging infectious diseases*, 7(3):382, 2001.
- [20] Matthew D Whiteside, Chad R Laing, Akiff Manji, Peter Kruczkiewicz, Eduardo N Taboada, and Victor PJ Gannon. Superphy: predictive genomics for the bacterial pathogen escherichia coli. *BMC microbiology*, 16(1):65, 2016.
- [21] Chad Laing, Cody Buchanan, Eduardo N Taboada, Yongxiang Zhang, Andrew Kropinski, Andre Villegas, James E Thomas, and Victor PJ Gannon. Pan-genome sequence analysis using panseq: an online tool for the rapid analysis of core and accessory genomic regions. *BMC bioinformatics*, 11(1):461, 2010.
- [22] Emma Griffiths, Damion Dooley, Morag Graham, Gary Van Domselaar, Fiona SL Brinkman, and William WL Hsiao. Context is everything: Harmonization of critical food microbiology descriptors and metadata for improved food safety and surveillance. *Frontiers in Microbiology*, 8:1068, 2017.
- [23] Jerven T Bolleman, Christopher J Mungall, Francesco Strozzi, Joachim Baran, Michel Dumontier, Raoul JP Bonnal, Robert Buels, Robert Hoehndorf, Takatomo Fujisawa, Toshiaki Katayama, et al. Faldo: a semantic standard for describing the location of nucleotide and protein feature annotation. *Journal of biomedical semantics*, 7(1):39, 2016.
- [24] Cátia Vaz, Alexandre P Francisco, Mickael Silva, Keith A Jolley, James E Bray, Hannes Pouseele, Joerg Rothganger, Mário Ramirez, and João A Carriço. Typon: the microbial typing ontology. *Journal of biomedical semantics*, 5(1):43, 2014.
- [25] Tim Berners-Lee, James Hendler, Ora Lassila, et al. The semantic web. *Scientific american*, 284(5):28–37, 2001.
- [26] Ian Horrocks, Bijan Parsia, Peter Patel-Schneider, and James Hendler. Semantic web architecture: Stack or two towers? In *International Workshop on Principles and Practice of Semantic Web Reasoning*, pages 37–41. Springer, 2005.
- [27] Wes Felter, Alexandre Ferreira, Ram Rajamony, and Juan Rubio. An updated performance comparison of virtual machines and linux containers. In *Performance Analysis of Systems and Software (ISPASS), 2015 IEEE International Symposium On*, pages 171–172. IEEE, 2015.
- [28] Torsten Seemann. Prokka: rapid prokaryotic genome annotation. *Bioinformatics*, 30(14):2068–2069, 2014.
- [29] Samia N Naccache, Scot Federman, Narayanan Veeraraghavan, Matei Zaharia, Deanna Lee, Erik Samayoa, Jerome Bouquet, Alexander L Greninger, Ka-Cheung Luk, Barryett Enge, et al. A cloud-compatible bioinformatics pipeline for ultrarapid pathogen identification from next-generation sequencing of clinical samples. *Genome research*, 24(7):1180–1192, 2014.
- [30] Alexandre G de Brevern, Jean-Philippe Meyniel, Cécile Fairhead, Cécile Neuveglise, and Alain Malpertuy. Trends in it innovation to build a next generation bioinformatics solution to manage and analyse biological big data produced by ngs technologies. *BioMed research international*, 2015, 2015.
- [31] Jeremy Goecks, Anton Nekrutenko, and James Taylor. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome biology*, 11(8):R86, 2010.
- [32] Michael C Schatz. Biological data sciences in genome research. *Genome research*, 25(10):1417–1422, 2015.

8 APPENDIX