Massively Parallel Processing Database for Sequence and Graph Data Structures Applied to Rapid-Response Drug Repurposing

Christopher D. Rickett
Hewlett Packard Enterprise
chris.rickett@hpe.com

Kristyn J. Maschhoff
Hewlett Packard Enterprise
kristyn.maschhoff@hpe.com

Sreenivas R. Sukumar
Hewlett Packard Enterprise
sreenivas.sukumar@hpe.com

2020 IEEE International Conference on Big Data (Big Data) | 978-1-7281-6251-5/20/$31.00 ©2020 IEEE | DOI: 10.1109/BigData50022.2020.9378331

***Abstract*—In this paper, we present the application of a massively parallel-processing graph database for rapid-response drug repurposing. The novelty of our approach is that the scalable graph database is able to host a knowledge graph of medically relevant facts integrated from multiple knowledge sources and also act as a computational engine capable of in-database protein sequence analytics. We demonstrate the performance of the graph database on a real-world use-case to hypothesize cures for COVID-19, leveraging its built-in accelerated protein-sequence matching capabilities at unprecedented scale (to simultaneously handle data size and query latency requirements for interactive research). Based on supporting evidence from medical literature, we show that results generated by computing similarity of COVID-19 virus proteins across 4 million other open-science sequences and intelligently traversing over a 150 billion facts from open-science medical knowledge produces biologically insightful results. By presenting sample queries and extending application to use-cases beyond COVID-19, we demonstrate the use and value of the novel database for hypotheses generation in reducing the time-to-insight and increasing researcher productivity with interactivity.**

***Index Terms*—graph database, graph analytics, in-database analytics, distributed processing, parallel processing, sequence analytics.**

I. Problem Statement

Amidst the pandemic caused by the novel coronavirus, drug repurposing – the investigation of existing drugs for new therapeutic purposes – emerged as the first ray of hope towards the discovery of a medical cure. The state-of-the-art drug repurposing pipeline involves understanding the protein structures of the disease-causing organism, interpreting the interactions of the organism's protein structures with the human body, mining through properties of potential drug molecules, connecting the dots across curated literature to explain the mechanism-of-action, searching for evidence in assay data and analyzing for potential safety and efficacy using data from prior trials, and more. This process is done manually and takes several months today. The cumbersome nature of the problem is attributed to the time required for a life-sciences researcher to: (a) understand the disease-causing organism by matching and comparing protein sequences to previously known or studied disease-causing organisms (over 4 million sequences), (b) handle and process multi-modal Big data (protein sequences, proteomic interactions, bio-chemical pathways, structured data from past clinical trials etc.), (c) integrate and search for patterns connecting across the multiple multi-modal multi-terabyte datasets, (d) install, configure and run a plethora of tools (genetics, proteomics, molecular dynamics, data science etc.) to generate insights, and finally (e) verify and validate the scientific rigor for pharmacological interpretation. This paper is motivated by a need for rapid response to accelerate the drug repurposing pipeline amidst a pandemic.

Our proposed technology solution is the design and implementation of a massively parallel database that (a) stores, handles, hosts and processes multi-modal data represented in the form of knowledge graphs, (b) provides interactive query and semantic-traversal capabilities for data-driven discovery, (c) accelerates domain-specific functions such as Smith-Waterman algorithm to conduct protein similarity analysis, vertex-centric and whole-graph algorithms such as PageRank for graph-theoretic connectivity and relevance analysis, and (d) runs/executes a workflow of queries across multiple datasets to generate drug hypotheses in the order of seconds as opposed to months. We demonstrate the application of this technology using an integrated knowledge graph of multiple multi-modal life-science databases, conduct protein-sequence matching in parallel and present a novel rapid drug-repurposing methodology that is able to query across 4+ million proteins, 155+ billion facts while handling approximately 30 terabytes of data.

Our approach extends the proven value of knowledge graphs for the drug repurposing problem described in [1]–[4] by offering a generalizable Big Data platform to other biomedical discovery problems beyond the COVID-19 pandemic. In doing so, we are able to claim the following contributions: (a) A scalable graph database that offers order-of-magnitude computational speed-up and interactivity required for knowledge traversal and discovery, (b) An integrated life-sciences knowledge graph that captures the open-science universe of available biomedical facts, (c) hypotheses of potential drug candidates for the ongoing pandemic, (d) reproducible code and results for future studies on the universe of biomedical facts (on viruses, proteins, drugs, bio-chemical pathways) as opposed to the state-of-the-practice limited to disease-specific knowledge graphs.

We have organized this paper as follows. Section II describes the graph database solution. Section III describes the application of the database to the COVID-19 pandemic by documenting datasets integrated for the experiments. Section IV presents results from sample queries run for the rapid-response drug repositioning experiments and showcases the performance characteristics of the graph database solution from ingest to hypothesis generation. We conclude with future directions in Section IV.

II. The Graph Engine

The proposed technology solution builds on prior work on the Cray Graph Engine [5]. The Cray Graph Engine (CGE) is an in-memory semantic graph database designed to scale to hundreds of nodes and tens of thousands of processes on the Cray XC supercomputer to support interactive querying of large data sets (∼100s of terabytes). CGE ingests datasets of N-Triples/N-Quads based on the standardized Resource Description Framework (RDF) format and enables queries using the SPARQL query language [6]. RDF data is expressed as a labeled, directed graph with the "quad" consisting of four fields: subject, predicate, object and graph. A triple is simply a quad that is stored in the "default graph". For example, the following is a simplified version of an example RDF triple from the Uniprot COVID-19 data [7] that could be loaded into CGE:

```
<urn:P0DTC2> <urn:mnemonic> "SPIKE_SARS2" .
```

A semantic graph is a collection of such triples with subjects and objects representing vertices and predicates representing edges between the vertices. Semantic graph databases differ from relational databases in that the underlying data structure is a graph, rather than a structured set of tables. The graph structure makes semantic databases ideal for analyzing multi-modal unstructured and structured data that is loosely connected or is schema-less – as is the case with social network interactions or interactions between proteins and genes in living organisms [8].

A. Background

The first scalable semantic graph database developed at Cray (recently acquired by Hewlett Packard Enterprise) was the Urika-GD appliance, which was built upon the scalable shared memory model of the Cray XMT2. The XMT2 processor consisted of 128 hardware streams per processor, which could be used to greatly improve latency tolerance for memory operations across processors. The Urika-GD database appliance utilized extreme multi-threading to enable parallel search of basic graph patterns. The Urika-GD database was later ported to the Cray XC30 architecture by utilizing the Partitioned Global Address Space (PGAS) programming model and Coarray-C++ [5]. CGE is developed as a highly parallel distributed application based on Coarray-C++, which is built on top of Cray's PGAS library that enables processes distributed across compute nodes to share data and synchronize operations. In Coarray-C++, the distributed processes are referred to as images, and within this paper the two terms may be used interchangeably.

CGE consists of two main components: the dictionary and the query engine. The dictionary is responsible for building the database, which is the process of ingesting the raw N-Triples/N-Quads files from the high performance Lustre file system and converting them to the internal representation used by CGE. The dictionary stores all unique RDF strings from the N-Triples/N-Quads and provides a mapping between the unique strings and the integer identifiers used for the quads internally by the query engine. Much of the dictionary build time is dominated by the Lustre I/O time, which has been demonstrated in previous studies [8], and will be further discussed in Section IV.

The CGE query engine processes SPARQL queries and SPARUL update requests, provides a number of built-in graph algorithms (such as measures of centrality, PageRank, connectivity analysis that are used in [1], [3], [4]) that can be applied to query data and returns results to the user. The core work performed by the query engine is matching the basic graph pattern in the SPARQL queries as well as supporting operations on the query results, such as FILTER and ORDER, that allows users to remove and sort solutions, respectively.

Since the Urika-GD appliance, significant effort has allowed porting CGE to improve the performance and scalability on the Cray XC supercomputer products with general-purpose processors and using a high-performance interconnect. The success of the scalability work demonstrated the ability of CGE to ingest and query a trillion triples using the well known Lehigh University Benchmark (LUBM) [9]. Using the LUBM dataset, CGE was shown to scale efficiently to hundreds of nodes and tens of thousands of processes, enabling CGE to perform the standard LUBM queries an order of magnitude faster than the closest competitors [8]. Further details describing the design and performance of CGE can be found in previous publications for the Cray User Group [5], [8], [10]. We leverage CGE's distributed data processing architecture to enable execution of pattern-search queries that execute orders-of-magnitude faster than other state-of-the-art graph databases to offer iterative discovery required for data-driven discovery. Multiple features were added to CGE to specifically support rapid response drug repurposing. We detail the new capabilities in the sub-section below.

B. Improvements for Drug Repurposing

Two core CGE improvements were made to support drug repurposing: (a) the support for user-defined functions (UDFs) for in-database protein sequence analytics and (b) the ability to execute such domain-specific UDFs in parallel for speed-up and scale-out using the SPARQL front-end user interface described above.

1) User Defined Functions: The syntax for user-defined functions in CGE follows Apache Jena guidelines [11] since CGE leverages Jena's SPARQL query parser interface. The SPARQL interface allows custom functions inside query expressions to enable domain specific operations on data as

part of a query. This is a feature that allows users to define, express and execute domain-specific mathematical operations to evaluate and rank query results [11] that are not supported in SPARQL. Such graph operations can be implemented as custom functions that are defined by URI in expressions. In the past, we had leveraged this capability to define a few custom functions within CGE, such as *sqrt*, however we never extended this capability to allow users to define their own function. Calls to these user-defined functions take the form:

```
prefix arq: <http://jena.hpl.hp.com/ARQ/function#>
...
arq:sqrt(5)
...
```

For supporting custom user defined functions (UDF) for drug repurposing, a new URI, *arq:user_func*, is allowed in CGE to invoke a callout to a UDF that exists separately from CGE. A simple C interface is defined for a function named *cge_user_eval* that CGE can execute as part of an expression. The *cge_user_eval* function takes four arguments that provide the total number of arguments, a list of the arguments, the return value and the return type. This allows users to pass data from CGE to the UDF, evaluate the arguments, and return a primitive value (e.g., boolean, integer or double) that can be used to evaluate the SPARQL expression.

Since CGE executes in a massively parallel manner, with potentially tens of thousands of images running concurrently, any UDF invoked as part of a query is also executed in parallel on the query solution set. The parallel execution of the UDF enables it to scale to datasets that may otherwise be too large by dividing the data across the parallel images. A UDF can also be applied to the entire solution set in an embarrassingly parallel manner. Further, the parallel execution via UDFs enables distributed execution of computationally intensive algorithms by breaking down complex processing tasks into images that will have a considerably smaller input set to process.

2) Performance Improvements: Another improvement made to CGE focused on improving the performance of database operations such as FILTER or GROUP. These operations enable users to compare terms that are found as part of a query match to apply some order or ranking. The raw strings for these terms are stored in the CGE dictionary, which is a distributed hash table that spreads the strings across all processes. This distribution of the terms results in significant work pulling terms local to a process when they are needed as part of an operation, such as FILTER or GROUP.

To improve the overall performance and scalability of these operations, the strings required by images are now fetched as large blocks in a coordinated manner. Each image goes through the results and creates a list of strings it requires from each other image. All images then fetch the required strings from each other as a single block, rather than issue a remote fetch of each string individually. This increases the size of each message but significantly reduces the total number of messages required. This communication pattern matches what is done in CGE for the core graph operations, such as JOIN and MERGE, and has been shown in previous studies to significantly improve performance by reducing the number of outstanding messages at a time [8]. This CGE improvement is critical for parallel pairwise comparisons of a query protein sequence with millions of open-science sequences and rank-ordering the result set.

III. Application to COVID-19

A. The knowledge graph

To demonstrate the power and value of the graph engine described in the previous section, we integrated a Life Sciences knowledge graph. We selected a set of open/publicly available biomedical data resources commonly used in life sciences and systems biology research. The typical workflow for researchers is to perform searches in one of the databases, then construct queries for another database, and iterate. The effort of manually mapping between the ontologies of various data sources and piecing together results from multiple query end points (or using yet another database to perform this translation), is a cumbersome process. The scalability of CGE enables all of the relevant databases to be loaded in one environment, enabling seamless cross-database queries. Although federated queries could be used to query across multiple databases, this approach has been shown to be problematic for complex queries, due to challenges such as network access from firewalled systems, query rate limits or simply performance problems of complex federated queries. The scalable load time of CGE also enables frequent reloads of the data set, including integration of in-house data on top of a background of the public databases during the workflow. Further, the performance and scalability of CGE for the database build process enables updated data to be quickly pulled in and the database fully rebuilt in less than an hour. A discussion of scalability and performance for CGE applied to a Life Sciences knowledge graph was previously discussed in [8].

The integrated Life Sciences knowledge graph assembled to study potential drug repurposing candidates for COVID-19 was generated from a collection of publicly available databases. Descriptions of the larger databases in the collection, and those explicitly mentioned in this paper, are described in more detail below.

1) Uniprot: The UniProt [7] database is the authoritative collection of functional information on proteins, and includes annotations, interrelationships, and in some cases, the amino acid sequences of the proteins themselves. Proteins are the building blocks used for studying drug protein structure and interactions. The interactions between proteins are complex and widely linked, so a graph representation is particularly useful. Uniprot concentrates on Human proteins, though other widely studied organisms are also well represented.

The UniProt Consortium is a collaboration between the European Bioinformatics Institute (EBI), the Swiss Institute of Bioinformatics (SIB) and the Protein Information Resource (PIR). It has been a pioneer in Semantic Web technology, and Uniprot has been distributed in RDF format since 2008.

Uniprot is continually growing as more scientific data is added. New Uniprot releases are distributed every four weeks.

For this study, the majority of the Uniprot RDF data is from the March 19, 2020 release. There is a new UniProt portal for providing the latest information for COVID-19 coronavirus protein entries and receptors which is updated independently of the general UniProt release cycle. For the COVID-19 study this allowed us to more quickly pull in updated COVID data. The COVID-19 Uniprot data for the knowledge graph was updated May 22, 2020. The Uniprot database contains approximately 87.6 Billion triples. In the form of an N-Triples (.nt) file on disk it is roughly 12.7 Terabytes. To simplify querying across multiple databases, we merged all named graphs into a single default graph.

2) PubChem: PubChem [12] is an open chemistry database maintained by the National Institutes of Health (NIH). The PubChemRDF project provides RDF formatted information for the PubChem Compound, Substance, and Bioassay databases. The knowledge graph for this study used the V1.6.3 beta version of PubChemRDF download from ftp://ftp.ncbi.nlm.nih.gov/pubchem/RDF on March 30, 2020. The PubChemRDF database contains approximately 80 Billion RDF triples. In the form of N-triples this amounts to about 13 Terabytes on disk.

3) ChEMBL: ChEMBL [13] is a manually curated database of bioactive molecules with drug-like properties. It brings together chemical, bioactivity and genomic data to aid the translation of genomic information into effective new drugs. The data is updated regularly, with releases approximately every 3-4 months. ChEMBL-RDF Release 27.0 (May 18, 2020) was integrated into the knowledge graph for this study. The ChEMBL database contains approximately 539M triples. In the form of N-triples this amounts to about 81 Gigabytes on disk.

4) Bio2RDF datasets: Bio2RDF [14] is an open-source project that uses Semantic Web technologies to pull together a diverse set of datasets from multiple data providers. In addition to providing an online Virtuoso-based SPARQL endpoint [15] for querying across the collection of heterogeneous datasets, Bio2RDF also provides a portal to download the converted RDF data files for datasets included in the Bio2RDF database. The Bio2RDF datasets included in the knowledge graph were downloaded from [16].

The full Bio2RDF collection consists of approximately 11 Billion triples across 35 datasets and includes the DrugBank, PubMed, and MESH datasets.

5) OrthoDB: OrthoDB (https://www.orthodb.org) provides evolutionary and functional annotations of orthologs, i.e. genes inherited by extant species from their last common ancestor. Since orthologs are the most likely candidates to retain functions of their ancestor gene, OrthoDB is aimed at narrowing down hypotheses about gene functions and enabling comparative evolutionary studies.

The OrthoDB [17], [18] database contains approximately 2.2 Billion RDF triples describing evolutionary and functional properties of 40 Million genes from 15 thousand organisms. In the form of N-triples this amounts to about 275 Gigabytes on disk.

TABLE I
KNOWLEDGE GRAPH DATASET CHARACTERISTICS. RAW SIZES BEFORE DUPLICATE REMOVAL.

Dataset	Size (on disk)	Size (triples)	source
UniProt (Mar 2020)	12.7 TeraBytes	87.6 Billion	[7]
PubChemRDF (v1.6.3 beta)	13.0 TeraBytes	80.0 Billion	[12]
ChEMBL-RDF (27.0)	81 GigaBytes	539 Million	[13]
Bio2RDF (Release 4)	2.4 TeraBytes	11.5 Billion	[14]
OrthoDB (v10)	275 GigaBytes	2.2 Billion	[17]
Biomodels (r31)	5.2 GigaBytes	28 Million	[20]
Biosamples (v20191125)	112.8 GigaBytes	1.1 Billion	[21]
Reactome (r71)	3.2 GigaBytes	19 Million	[22]

6) BioModels: The BioModels [19] database is a repository of mathematical models representing biological systems. It currently hosts a range of models describing processes like signaling, protein-drug interaction interactions, metabolic pathways, epidemic models and many more. The models that BioModels hosts are usually described in peer-reviewed scientific literature and in some cases, they are generated automatically from pathway resources (Path2Models). These models are manually curated and semantically enriched with cross-references to external data resources (such as publications, databases of compounds and pathways, ontologies, etc.)

B. Query Mechanism

With the capability of CGE to support UDFs, queries could now be written that combine information from the knowledge graph and apply domain specific UDFs to the data in order to better refine results. For drug repurposing, our approach utilizes a UDF that performs protein sequence similarity to infer connections between proteins. This enables connections to be inferred between proteins that little may be known about, such as COVID-19, and proteins that are well documented in open datasets such as Uniprot and ChEMBL.

We chose the Smith-Waterman (SW) protein similarity algorithm for aligning pairs of sequences and computing similarity scores for the alignment. For two sequences of length m and n, the SW algorithm returns the optimal local alignment and similarity score with a computational time complexity of $O(mn)$ [23]. The local alignment is useful for providing alignments describing the most similar regions within sequences, rather than end-to-end alignments of sequences returned by global alignments [23]. Since SW returns an optimal local alignment it is an essential component of many aligners. However, the computational complexity limits the extent the algorithm is utilized for comparing large sequence sets [24].

The SW algorithm was chosen for our knowledge graph because of the preference to score similarity using optimal local alignment and the availability of a highly optimized open source implementation as a standalone C/C++ library that could be loaded by CGE [24]. Given the highly parallel implementation of CGE, a user can query the knowledge graph and perform millions of protein similarity computations in a

matter of seconds, enabling the solutions to easily be filtered and ranked by similarity score.

To normalize the scores, each sequence is compared to itself. The product of the square root of those scores is used as the denominator [25] as outlined in listing 1.

```
int64_t score = prot_cmp(ref, read);
int64_t max_ref_score = prot_cmp(ref, ref);
int64_t max_read_score = prot_cmp(read, read);
double norm_score = score /
    (sqrt(max_ref_score) * sqrt(max_read_score));
```

Listing 1. Normalization method of Smith-Waterman scores

IV. Results

A. The COVID-19 Use Case

As mentioned earlier, this work was motivated by the need to identify drug candidates for the novel COVID-19 virus. Our approach begins by first allowing researchers to understand how similar or different the novel virus is to other known viruses. If parts of the protein sequence that make up the novel virus have sequence and functional overlaps with other known viruses, the information in the integrated knowledge graph helps us extrapolate the search to identify potential drug candidates that are known to inhibit disease-causing activity on the known viruses. We share sample queries that implement the similarity-based extrapolation in the following paragraphs.

1) COVID-19 Similarity: The COVID-19 protein sequence is made up of several non-structural proteins, envelope proteins, Spike protein, etc. To hypothesize potential drugs that bind or interact with the different parts of the COVID-19 viral protein, we first have to identify open-science proteins that have similar structure to the novel COVID-19 mutation. The query in listing 2 is an example to find proteins most similar to the COVID-19 Spike protein sequence.

```
select ?protB ?name ?mnem ?sciName ?sim
where {
  # Look up the info for our protein of interest
  ?protein a core:Protein ;
    core:mnemonic 'SPIKE_SARS2' ;
    core:sequence ?isoform .
  ?isoform rdf:value ?seq .

  # Look up all other proteins with sequence info
  ?protB a core:Protein ;
    core:sequence ?isoformB ;
    core:mnemonic ?mnem ;
    up:recommendedName ?recommended .
  ?isoformB rdf:value ?seqB .
  ?recommended up:fullName ?name .

  # Optionally, look for the scientific name of the organism.  This
  # may not exist if the protein data is too new, such as for covid-19, so
  # make it optional so we still get the match.
  optional {
    ?protB up:organism ?taxon .
    ?taxon core:scientificName ?sciName .
  }

  # Compare the protein of interest to each protein to get a sim value
  bind(arq:user_func(?seq, ?seqB) as ?sim)
  filter(?sim >= 0.1)
}
# List the proteins with the highest sim score first
order by desc(?sim)
```

Listing 2. SPARQL query to rank similar proteins to reference protein

The similarity query in listing 2 first looks up the protein sequence for SPIKE_SARS2 using the Uniprot mnemonic. Next, the sequences for all proteins that have sequences and names are retrieved. Finally, each of these sequences is compared to the sequence for the SPIKE_SARS2 and the similarity score is saved in the variable *sim*. The *bind* clause saves all of the *sim* values in a temporary table so that they can be used for other operations and returned to the user. In this query, any result with a similarity score less than 0.1 is removed and the results are returned in descending order by the similarity score.

The protein returned with the highest similarity score was A0A2D1PX97, which is "Bat SARS-like coronavirus", with a similarity score of 0.817. Several of the top results are bat coronaviruses or coronaviruses in other species, as shown in the top 10 results listed in table II. The similarity scores quickly drop from 0.79 to 0.37, which is the point where Middle East Respiratory Syndrome (MERS) first appears in the results with a similarity score of 0.368 for protein A0A2I6PIX8, which is "Middle East respiratory syndrome-related coronavirus". Following several proteins for MERS are a number of coronaviruses in other species, including bovine, human, rabbit and murine, and several non-coronavirus proteins begin appearing such as A0A1B2RX89 for "Infectious bronchitis virus" with a similarity score of 0.322. These scores match up well with research that has suggested that COVID-19 likely originated from bats and has a close similarity to MERS [26].

TABLE II
Top 10 protein sequences most similar to COVID-19 spike

Protein	Scientific Name	Score
A0A2D1PX97	"Bat SARS-like coronavirus"	0.817
A0A0U2IWM2	"SARS-like coronavirus WIV16"	0.817
A0A2D1PXA9	"Bat SARS-like coronavirus"	0.816
U5WLK5	"Bat SARS-like coronavirus RsSHC014"	0.814
A0A2D1PX29	"Bat SARS-like coronavirus"	0.814
U5WHZ7	"Bat SARS-like coronavirus Rs3367"	0.813
U5WI05	"Bat SARS-like coronavirus WIV1"	0.813
A0A2D1PXC0	"Bat SARS-like coronavirus"	0.813
A0A2D1PXD5	"Bat SARS-like coronavirus"	0.812
A0A4Y6GL47	"Coronavirus BtRs-BetaCoV/YN2018B"	0.812

2) COVID-19 Drug Repurposing: Our next step was to leverage the knowledge graph to find potential drugs that could be repurposed for COVID-19 based upon the similarity score rankings. To do this a SPARQL query was needed that would work in reverse – rather than looking for all known targets of a given compound the query starts with an unknown protein and searchs for potential compounds that could target it. In this case we focused on compounds that would have an inhibitory action on the proteins. The query in listing 3 was used to do this search.

```
select distinct ?protB ?sciName ?label ?sim
  where {
  {
    # Look up activities with an inhibitory effect that have small
    # molecules and have gone through a certain development phase
    select ?activity ?assay ?label
    where {
      { ?activity cco:type 'Inhibition' } union { ?activity cco:type
      'IC50' }
      ?activity a cco:Activity ;
        cco:hasMolecule ?molecule ;
        cco:hasAssay ?assay .
      ?molecule rdf:type cco:SmallMolecule ;
        cco:highestDevelopmentPhase ?phase ;
        skos:prefLabel ?label .
      filter(?phase >= 3)
    }
  }

  # Look up proteins that have a sequence, compare to our sars2
  # protein, and select only the top X proteins based similarity
  {
    select distinct ?protB ?sim
    where {
```

```
      # Look up the sequence for the sars2 spike protein
      ?protein a core:Protein ;
        core:mnemonic 'SPIKE_SARS2' ;
        core:sequence ?isoform .
      ?isoform rdf:value ?seq .

      # Look up activities that target proteins. We do this here to
      # make sure we only select proteins that are known targets
      ?targetcmpt cco:targetCmptXref ?protB .
      ?target cco:hasTargetComponent ?targetcmpt .
      ?assay cco:hasTarget ?target .

      # Look up known proteins and get sequence values
      ?protB a core:Protein ;
        core:sequence ?isoformB .
      ?isoformB rdf:value ?seqB .
      bind(arq:user_func(?seq, ?seqB) as ?sim)
    }
    order by desc(?sim)
    limit 150
  }

  # Look up our compounds that target our proteins of interest, if any
  # do. This is a repeat from the inner-protein query because we do
  # not want the inner query to generate combinations for
  # proteins/compounds when finding the most similar proteins.
  # Redoing the joins here is quick.
  ?targetcmpt cco:targetCmptXref ?protB .
  ?target cco:hasTargetComponent ?targetcmpt .
  ?assay cco:hasTarget ?target .

  # Optionally, look for the scientific name of the organism.
  optional {
    ?protB up:organism ?taxon .
    ?taxon core:scientificName ?sciName .
  }
}
order by desc(?sim)
```

Listing 3. SPARQL query to find potential drugs that could be repurposed

There are three main components to the query. First, the top inner-query searches ChEMBL for information about compounds that have an inhibitory activity that have been through a certain development phase. Since the intent is to repurpose existing drugs, the compounds were limited to only those that are in phase 3 development or higher for clinical trials [27]. In the second inner query, all proteins that are known targets of a given compound are compared to the COVID-19 Spike protein. The results are put into descending order by the similarity to the SPIKE_SARS2 protein and only the proteins for the top 150 isoforms are returned. There are often multiple sequences for a given protein and in this case, the top 150 isoforms are associated with approximately the top 50 most similar proteins. The final part of the query again matches the selected proteins to the compounds that target them as well as the activity information from the first inner query. The final results are returned in descending order by the similarity score to highlight compounds that could potentially be repurposed based on similarity to the COVID-19 spike.

The reverse query returns compounds targeting proteins with similarity scores ranging from 0.2 down to 0.183. Several of these compounds are for drugs that have already been put into clinical trials because of their potential to be repurposed against COVID-19 [28]. Some of the top scoring protein sequences against the COVID-19 spike found by the reverse query that are also in clinical trials are shown in table III.

TABLE III
EXAMPLE DRUGS CURRENTLY IN CLINICAL TRIALS FOR COVID-19 THAT APPEAR IN REVERSE QUERY RESULTS

Protein	Compound Name	Score
P52333	BARICITINIB	0.194
P17948	RIBAVIRIN	0.189
P17948	RITONAVIR	0.189
P17948	DEXAMETHASONE	0.189
P17948	AZITHROMYCIN	0.189
P08183	LOPINAVIR	0.187

One method we have used to validate the results returned by the reverse query is to compare the overlap between the compounds returned with those currently in clinical trials for COVID-19. Based on the drugs currently part of clinical trials in early June, we created a list of 196 unique drugs to compare our results against [28]. For the above query considering the top 150 isoform sequences most similar to the SPIKE_SARS2 protein, the results returned by the knowledge graph include 91 of the 196 compounds (46%). The significant overlap between compounds found by the reverse query with the clinical trials list also helps define the range of scores that could be considered interesting. Since the proteins found by the reverse query all have similarity scores between 0.183 and 0.20, it seems reasonable that compounds targeting other proteins with scores in the same range could have a beneficial impact against COVID-19 as well.

3) A New Hypothesis - Tetanus: One potentially interesting result returned by the reverse query using the SPIKE_SARS2 is tetanus toxin, which has the Uniprot identifier P04958 and mnemonic TETX_CLOTE. The reverse query against the spike returns TETX_CLOTE as the highest match with a similarity score of 0.20. Given the large rate of asymptomatic positive COVID-19 cases, which the Centers for Disease Control and Prevention (CDC) currently estimates to be 40% [29], the TETX_CLOTE result caused one unexpected, but interesting hypothesis - that the tetanus vaccine could be contributing to the asymptomatic rate by enabling the immune system to generate a reasonable response to the virus and reduce the severity of symptoms. According to the CDC, in 2017 approximately 63.4% of adults 19 and older in the US had received some form of the tetanus vaccine within the last 10 years as recommended [30], with a notable decline in individuals greater than 65 years old. While tetanus is caused by a bacteria and COVID-19 is a virus, there are multiple examples of heterologous immunity between bacteria and viruses. This heterologous immunity has at least initially been attributed to amino acid sequence similarities of T and B cell epitopes for antigens of different pathogens [31].

The high level of asymptomatic and mild cases in children in particular has lead many to consider if a childhood vaccine is a contributing factor due to the immune response stimulated by the vaccine [32]. The severity of COVID-19 in children is significantly less than adults and some experts have reported that over 90% of children who test positive could be asymptomatic or have only mild symtpoms [33]. The World Health Organization (WHO) reports that 85% of infants worldwide receive the recommended 3 doses of diphtheria-tetanus-pertusis (DTP3) vaccine [34], which correlates well with the symptom rates of COVID-19 observed in children. Countries such as Brazil have noted poor vaccination coverage for DTP3 in a significant percentage of their municipalities [35], which also correlates with the higher mortality rate seen in Brazil [36].

Other possibly interesting populations are pregnant women and prison inmates. A recent study of pregnant women being universally tested for COVID-19 upon admission for delivery

found that of the 215 women admitted, 33 were positive and 29 of those 33 (87.9%) were asymptomatic [37], which is approximately double the rate of the general population [29]. There are two vaccines that pregnant women are recommended to receive with each pregnancy, which are the influenza vaccine and the Tetanus toxoid, reduced diphtheria toxoid and acellular pertussis (TDaP) vaccine [38]. Interestingly, the only childhood vaccine recommended for pregnant women is TDaP [38]. Additionally, prison inmates in Arkansas, Ohio, North Carolina and Virginia also had an asymptomatic rate of 96% [29]. The vaccination policies for all of these states is not known. However, North Carolina has a published policy of keeping inmates up to date on certain vaccines, including TDaP, further supporting the potential correlation with the asymptomatic rates [39].

A recent study [40] also utilized sequence similarity between the COVID-19 Spike protein and proteins involved in viruses targeted by childhood vaccines to determine if a childhood vaccine could help against COVID-19. Based on the sequence similarities, the authors hypothesize that the MMR vaccine could provide protection against COVID-19 and may explain why children are less susceptible. The Uniprot identifiers for the proteins in that study are Q786F3 (Measles virus strain Ichinose-B95a) and P08563 (Rubella virus strain M33) [7]. Using the Clustal Omega alignment program available from Uniprot [7], the Q786F3 protein has a sequence identity and similarity percentage with the SPIKE_SARS2 protein of 6.87% and 31%, respectively. The P08563 protein has a sequence identity and similarity percentage of 10.68% and 24%, respectively. By comparison, the TETX_CLOTE protein using the same tool has a sequence identity and percentage similarity with the SPIKE_SARS2 protein of 12.78% and 30%, respectively.

To further test the potential similarity between the TETX_CLOTE and SPIKE_SARS2 proteins, the similarity query was used with TETX_CLOTE supplied as the protein of interest rather than SPIKE_SARS2. The same threshold of 0.2 was used in the query. As expected, the SPIKE_SARS2 was found with the same 0.20 similarity. Additionally, there are three other coronavirus proteins that show up in the tetanus results for the given threshold, which are shown in Table IV:

TABLE IV
ADDITIONAL SPIKE GLYCOPROTEINS WHICH SHOW HIGH SIMILARITY TO TETANUS

Protein	Mnemonic	Scientific Name	Score
A0A023PTS3	A0A023PTS3_CVHSA	Rhinolophus affinis coronavirus	0.203
A0A023PUW9	A0A023PUW9_CVHSA	Rhinolophus affinis coronavirus	0.203
A0A0U2IWM2	A0A0U2IWM2_9BETC	SARS-like coronavirus WIV16	0.2

The first two for "Rhinolophus affinis coronavirus" are important because a recent study found a SARS strain from 2013, SARSr-Ra-BatCoV-RaTG13, with a 96.1% genomic identity to COVID-19 that was reported in Rhinolophus affinis bats [41]. The authors use this identity and other similar identities to bat SARS strains to illustrate the possible bat origin of COVID-19. These results further support the possible bat origin and validate the SW method utilized to find similar proteins as part of a query.

B. Database Performance

To facilitate COVID-19 research, the Life Science knowledge graph was hosted on a small number of the larger internal Cray XC-40 development systems. These systems primarily contained a mix of Intel Broadwell, Skylake and Cascade Lake processors. The files containing the N-Triples used to build the database as well as the built database are stored on the attached Lustre filesystem and striped to match the available number of object server targets (OSTs) in the file system.

The performance results in this section were run on an internal 370 node XC-40 development system (336 compute nodes, 34 service nodes) with a mix of dual-socket 48-core Skylake nodes, and 48-core and 56-core Skylake nodes, ranging in frequency from 2.1-2.4GHz. The majority of these nodes have 192 GB DDR-2666 memory but 63 of the Cascade Lake nodes have the larger 384 GB DDR4-2933 memory. The attached Lustre file system is a Sonexion CS-L300N system with 8 OSTs providing 655 TB of storage. Database build and load times are dominated by I/O performance to/from the Lustre filesystem so I/O system performance is an important consideration. Query execution time reported is the strict query time and does not include the time required for writing the results to the Lustre file system which is common practice.

1) Database Build: As previously mentioned in the CGE background section, the first step done by CGE is to build a database from a set of input N-Triple/N-Quad files to produce the compiled database in the representation used by the query engine. The raw N-Triples input files used for the life sciences database are 28.29 TB on Lustre. The build process is handled by the dictionary component of CGE and consists of several steps, which are outlined in Table V along with the times (in seconds) for each step.

TABLE V
TIMES OF BUILD STEPS FOR LIFE SCIENCES DATABASE

Build Step	128 nodes x 16 images per node Times (seconds)	256 nodes x 16 images per node Times (seconds)
Read	1937.04	1613.16
Ingest	385.10	180.61
Sync	54.88	27.34
Update	380.87	154.35
Total Build	2757.89	1975.46
Checkpoint	422.09	375.35

As the numbers show, the build time for the database is dominated by the time to read the raw N-Triples files from Lustre, which is expected. The times for the remaining build steps (i.e., Ingest, Sync and Update) scale well from 128 (2048 images) to 256 nodes (4096 images). A checkpoint of the built database is written to Lustre so subsequent restarts of CGE with the same database can load the compiled database rather than having to ingest the raw N-Triples again. The built database is only $\sim$5.4 TB on disk, versus the $\sim$28.29 TB of raw N-Triples, and CGE can be restarted with the built database in approximately 568 seconds on 256 nodes with 16 images per node.

2) Spike Similarity Query: The first query used to test the performance of CGE with the life sciences knowledge graph was the similarity query from listing 2. This query searches the known proteins from Uniprot that met certain conditions, such as having a sequence value and a recommended name, and compares them to the sequence value for the SPIKE_SARS2. The similarity query finds 49,299,877 protein sequences to compare against the SPIKE_SARS2 sequence. Table VI shows the times for computing the SW calculations for the 49.3 million protein sequences as well as the total query time when executing on 128 and 256 nodes using either 16 or 32 images per node.

Looking at the times for the SW calculations we observe that the time to compute similarity scales well by both image count (i.e., cores) as well as node count. This is attributed to the fact that calculation is independent of the others so all images can compute the calculations for a subset of the protein sequences in parallel. The scaling also highlights the advantage of utilizing the SW calculation within the massively parallel context of CGE. If a knowledge graph executing on a single process performed the same SW calculations it would take ∼21,709 seconds (i.e., 10.6 x 2048), essentially making the query impossible in a serial context. The strict query times do show reasonable scaling from 128 to 256 nodes, at least at 16 images per node, but the scaling of the query is limited by performance limitations of the GROUP operator. Since the similarity scores are rounded to three decimal places there are a large number of repeat values, which need to be removed when storing the values as new variables within CGE (i.e., *?sim* query variable). Due to how CGE distributes these new variables across images the large number of repeats can result in several images waiting for a small number of images to finish processing the values they will store. Further, the query scaling is also related to the recent performance improvement made to enable operations such as GROUP and FILTER to fetch the required strings as blocks, rather than individually. The protein sequences can be quite long, ranging from hundreds to several thousand amino acids, so the fetching of these long strings as blocks from remote processes is crucial to prevent communication overhead from dominating the query performance.

3) Spike Reverse Query: The next query used to test the performance of CGE on the life sciences knowledge graph was the reverse query from list 3. This query starts at the protein sequence for SPIKE_SARS2 and searches for similar proteins that are targets of acceptable compounds that have the desired affect (i.e., inhibitory). The reverse query is considerably more complex than the similarity query because of the number of joins that must be done on the large intermediate results to find only the proteins or compounds that are desired. While the complex joins impacts the overall query time, the extra conditions imposed by the joins significantly reduces the number of protein sequences that must be compared. CGE has been optimized to filter out solutions early in the query during the scan and join phases by reusing information from previous portions of the query [8]. This optimization is critical for reducing the size of the intermediate results that must be joined, which is essential for not only performance but also memory requirements for queries as complex as the reverse query. For the case of the SPIKE_SARS2 reverse query, the number of proteins compared is only 1,165,914, which is much smaller than the 49.3 million proteins compared in the similarity query.

As the numbers show in Table VII, the SW times are a very small portion of the query time due to the abilities of CGE to leverage information within the knowledge graph to significantly reduce the number of proteins considered. The strict query time for the reverse query is more than double the time for the similarity query, which is expected because of the larger number of complex joins in the reverse query. The majority of the strict query time is dominated by the join. For example, on 256 nodes with 16 images per node the strict query time is 49.0 seconds and 34.52 seconds of that is spent doing the joins. Even with the complex joins, the performance from 128 to 256 nodes scales reasonably well when using 16 images per node (1.82x speedup), but while the query is faster with 32 images per node the scaling is not as efficient. The limited scaling with more cores per node is related to memory access bottlenecks caused by images accessing memory on the same node but on a different socket.

Since there are no other known large semantic graph engines capable of loading a real world life science dataset of this magnitude the performance of CGE cannot be easily compared with other database engines. However, previous benchmarks have clearly demonstrated with the standard LUBM trillion triples dataset that CGE is at least an order of magnitude faster than any competitor, especially when performing complex queries [8]. For the case of LUBM the typical benchmark query is number 9, which does multiple complex joins to search the dataset for a certain triangular relationship amongst entities [9].

TABLE VI
SCALING RESULTS FOR SPIKE SIMILARITY QUERY

Nodes	Images/Node	Total Images (Threads)	Time for Protein Similarity calculation (seconds)	Strict Query Time (seconds)
128	16	2048	10.6	27.0
	32	4096	5.6	21.6
256	16	4096	5.2	18.2
	32	8192	2.7	17.6

TABLE VII
SCALING RESULTS FOR REVERSE QUERY

Nodes	Images/Node	Total Images (Threads)	Time for Protein Similarity calculation (seconds)	Strict Query Time (seconds)
128	16	2048	0.58	89.4
	32	4096	0.33	60.0
256	16	4096	0.21	49.0
	32	8192	0.08	37.3

4) Scalability Advantages: The performance and scalability of CGE has been demonstrated in previous studies [8], [10] as well as in the results discussed above. This performance at scale is critical when trying to search such large datasets interactively. While smaller graph engines could be leveraged to analyze the same data, the time required to perform even

simple queries would almost certainly be too high to make the system useful. The ability of CGE to scale to hundreds of nodes and tens of thousands of images enables very large datasets to be ingested quickly and searched in seconds, even when performing the most complex queries across the graph. This scalability is also very advantageous for UDFs because it enables domain specific functions, some of which may be computationally complex, to easily be applied for improved refinement of the query results. These unmatched capabilities of CGE [8] give researchers the ability to search very large, real world datasets in order to find compounds that could be effectively repurposed in real time.

C. Insights on the Drug Repurposing Problem

While this study focused primarily on the application of the knowledge graph and SW sequence comparison to COVID-19, the changes and queries could be applied to any number of diseases of interest. For example, if the SPIKE_SARS2 mnemonic from the similarity query in listing 2 were changed to U5TGX1_COWPX, which is a Uniprot mnemonic for cowpox virus, the similarity query could be used to find the proteins most similar to the given cowpox. In this case, running that similarity query returns multiple cowpox viruses as the top three results, followed by a taterapox and then three camelpox viruses. More interesting results start showing up at number seven on the similarity list, which is Uniprot mnemonic V5QZD2_9POXV for the protein V5QZD2 for "Vaccinia virus WAU86/88-1", with a similarity score of 0.892. Vaccinia virus has been used more for human immunizations than any other vaccine due to its similarity to variola virus, which is the causative agent of smallpox [42].

The reverse query could also be used for other diseases to search for potential drugs that could be repurposed. For example, replacing the SPIKE_SARS2 mnemonic with the Uniprot mnemonic KITH_HHV11, which is for "Human herpesvirus 1" (HHV1), returns Brivudine as the top compound for inhibiting protein P06479 with a similarity score of 0.984. Brivudine is known to have a strong antiviral activity against varicella-zoster virus and herpes simplex virus type 1 [43]. The top 10 query results are all in fact for various HHV1 proteins with compounds such as Penciclovir and Acyclovir, which are known antivirals that target herpes simplex virus type 1 [43], [44].

These results clearly demonstrate the ability of CGE to enable the knowledge graph with the SW sequence similarity UDF to quickly and effectively find potential drugs that could be repurposed to target a different disease. These abilities of CGE can allow researchers to interactively search for potential candidates and apply subject matter expertise to further refine the query results to possibly increase the effectiveness of re-targeted drugs.

V. Conclusions and Future Work

In this paper, we discussed the creation of a real world life science knowledge graph that brought together multiple terabytes of data from different data sets. We demonstrated how CGE, a massively parallel semantic graph engine, can easily ingest and search a graph database of this scale and enable researchers to perform complex queries across the datasets in seconds. Further, we showed how the performance of CGE gives researchers the ability to interactively query the given database of 155 billion triples to search for potentially hidden connections between nodes of the graph that would otherwise be impossible to find.

This paper also discussed recent changes made to CGE to enable user defined functions that allow users to apply domain specific expertise to operations such as FILTER, GROUP and ORDER. Our work focused on leveraging an open source implementation of the Smith-Waterman sequence similarity algorithm as a UDF within a query to apply ranking to proteins targeted by a known compound based on the similarity to a given reference sequence. Using the SPIKE_SARS2 protein as a reference we showed how a query could be written that enabled CGE to find potential drugs that could be repurposed for COVID-19 in a matter of seconds. Additionally, we demonstrated that these abilities of CGE are not specific to COVID-19 and could easily be used to find potential drugs to repurpose for other known or new diseases of interest.

Using our two main queries of interest we demonstrated the strong scaling of the Smith-Waterman function within a query and showed good overall scaling for the queries themselves. The scaling tests showed some future areas to focus on for improving scaling. First, the GROUP operator has performance bottlenecks caused by too many duplicates being generated and the queries did not scale as well as desired with increased core counts. However, even with these limitations we were able to show good scaling for the complex queries, further demonstrating the unique capabilities of CGE.

Finally, we have shown that the unique capabilities of CGE, including massive parallelism, complex query performance and the scale of data that can ingested, combined with a protein sequence similarity can enable researchers to quickly and effectively repurpose existing drugs to target new diseases.

References

[1] S. Sadegh, J. O. Matschinske, D. B. Blumenthal, G. Galindez, T. Kacprowski, M. List, R. Nasirigerdeh, M. Oubounyt, A. Pichlmair, T. Rose, M. Salgado-Albarrán, J. Späth, A. Stukalov, N. K. Wenke, K. Yuan, J. Pauling, and J. Baumbach, "Exploring the sars-cov-2 virus-host-drug interactome for drug repurposing," *Nature Communications*, vol. 11, 2020.

[2] Y. Zhou, Y. Hou, J. Shen, Y. Huang, W. Martin, and F. Cheng, "Network-based drug repurposing for novel coronavirus 2019-ncov/sars-cov-2," *Cell Discovery*, vol. 6, 2020.

[3] D. M. Gysi, Í. D. Valle, M. Zitnik, A. Ameli, X. Gan, O. Varol, H. Sanchez, R. Baron, D. Ghiassian, J. Loscalzo, and A.-L. Barabási, "Network medicine framework for identifying drug repurposing opportunities for covid-19," *ArXiv*, 2020.

[4] "Knowledgegraphhub," https://github.com/Knowledge-Graph-Hub/kg-covid-19, 2020.

[5] K. Maschhoff, R. Vesse, and J. Maltby, "Porting the Urika-GD graph analytic database to the XC30/40 platform," in *Cray User Group Conference (CUG '15)*, Chicago, IL, 2015.

[6] A. Seaborne and S. Harris, "SPARQL 1.1 query language," W3C, W3C Recommendation, Mar. 2013, http://www.w3.org/TR/2013/REC-sparql11-query-20130321/.

[7] U. Consortium, "Uniprot: the universal protein knowledgebase," *Nucleic Acids Research*, vol. 47, p. D506, 2019.

[8] C. Rickett, U. Haus, J. Maltby, and K. Maschhoff, "Loading and Querying a Trillion RDF Triples with Cray Graph Engine on the Cray XC," in *Cray User Group Conference (CUG '18)*, Stockholm, Sweden, 2018.

[9] Y. Guo, Z. Pan, and J. Heflin, "Lubm: A benchmark for owl knowledge base systems," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 3, no. 2, pp. 158–182, 2005.

[10] K. Maschhoff, R. Vesse, S. Sukumar, M. Ringenburg, and J. Maltby, "Quantifying Performance of CGE: A Unified Scalable Pattern Mining and Search System," in *Cray User Group Conference (CUG '17)*, Seattle, WA, 2017.

[11] "Functions in arq," Tech. Rep. [Online]. Available: https://jena.apache.org/documentation/query/library-function.html

[12] S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang, and E. E. Bolton, "PubChem 2019 update: improved access to chemical data," *Nucleic Acids Research*, vol. 47, no. D1, pp. D1102–D1109, 10 2018. [Online]. Available: https://doi.org/10.1093/nar/gky1033

[13] D. Mendez, A. Gaulton, A. P. Bento, J. Chambers, M. De Veij, E. Félix, M. Magariños, J. Mosquera, P. Mutowo, M. Nowotka, M. Gordillo-Marañón, F. Hunter, L. Junco, G. Mugumbate, M. Rodriguez-Lopez, F. Atkinson, N. Bosc, C. Radoux, A. Segura-Cabrera, A. Hersey, and A. Leach, "ChEMBL: towards direct deposition of bioassay data," *Nucleic Acids Research*, vol. 47, no. D1, pp. D930–D940, 11 2018. [Online]. Available: https://doi.org/10.1093/nar/gky1075

[14] F. Belleau, M.-A. Nolin, N. Tourigny, P. Rigault, and J. Morissette, "Bio2RDF: Towards a mashup to build bioinformatics knowledge systems," *Journal of Biomedical Informatics*, vol. 41, pp. 706–716, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1532046408000415

[15] "Bio2rdf SPARQL endpoint," Tech. Rep. [Online]. Available: http://bio2rdf.org/sparql

[16] "Bio2rdf RDF download," Tech. Rep. [Online]. Available: http://download.bio2rdf.org/release/4

[17] E. M. Zdobnov, F. Tegenfeldt, D. Kuznetsov, R. M. Waterhouse, F. A. Simão, P. Ioannidis, M. Seppey, A. Loetscher, and E. V. Kriventseva, "OrthoDB v10: sampling the diversity of animal, plant, fungal, protist, bacterial and viral genomes for evolutionary and functional annotations of orthologs," *Nucleic Acids Research*, vol. 47, no. D1, pp. D807–D811, 2019.

[18] "Orthodb SPARQL endpoint," Tech. Rep. [Online]. Available: http://sparql.orthodb.org

[19] R. S. Malik-Sheriff, M. Glont, T. V. N. Nguyen, K. Tiwari, M. G. Roberts, A. Xavier, M. T. Vu, J. Men, M. Maire, S. Kananathan, E. L. Fairbanks, J. P. Meyer, C. Arankalle, T. M. Varusai, V. Knight-Schrijver, L. Li, C. Dueñas-Roca, G. Dass, S. M. Keating, Y. M. Park, N. Buso, N. Rodriguez, M. Hucka, and H. Hermjakob, "BioModels — 15 years of sharing computational models in life science," *Nucleic Acids Research*, vol. 48, no. D1, pp. D407–D415, 1 2020, gkz1055. [Online]. Available: https://doi.org/10.1093/nar/gkz1055

[20] V. Chelliah, N. Juty, I. Ajmera, R. Ali, M. Dumousseau, M. Glont, M. Hucka, G. Jalowicki, S. Keating, V. Knight-Schrijver, A. Lloret-Villas, K. Nath Natarajan, J.-B. Pettit, N. Rodriguez, M. Schubert, S. M. Wimalaratne, Y. Zhao, H. Hermjakob, N. Le Novère, and C. Laibe, "BioModels: ten-year anniversary." *Nucl. Acids Res.*, vol. 43, pp. D542–D548, 2015.

[21] S. Jupp, J. Malone, J. Bolleman, M. Brandizi, M. Davies, L. Garcia, A. Gaulton, S. Gehant, C. Laibe, N. Redaschi, S. M. Wimalaratne, M. Martin, N. Le Novère, H. Parkinson, E. Birney, and A. M. Jenkinson, "The ebi rdf platform: linked open data for the life sciences," *Bioinformatics*, vol. 30, no. 9, pp. 1338–1339, 05 2014. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3998127/

[22] A. Fabregat, S. Jupe, L. Matthews, K. Sidiropoulos, M. Gillespie, P. Garapati, R. Haw, B. Jassal, F. Korninger, B. May, M. Milacic, C. D. Roca, K. Rothfels, C. Sevilla, V. Shamovsky, S. Shorser, T. Varusai, G. Viteri, J. Weiser, G. Wu, L. Stein, H. Hermjakob, and P. D'Eustachio, "The reactome pathway knowledgebase." *Nucleic Acids Res*, vol. 46, no. D1, pp. D649–D655, Jan 2018.

[23] D. Okada, F. Ino, and K. Hagihara, "Accelerating the smith-waterman algorithm with interpair pruning and band optimization for the all-pairs comparison of base sequences," *BMC Bioinformatics*, vol. 16, no. 321, 2015.

[24] M. Zhao, W.-P. Lee, E. P. Garrison, and G. T. Marth, "Ssw library: An simd smith-waterman c/c++ library for use in genomic applications," *PLoS One*, vol. 8, no. 12, 2013.

[25] "Biostars," https://www.biostars.org/p/187592/, 2018.

[26] N. Petrosillo, G. Viceconte, O. Ergonul, G. Ippolito, and E. Petersen, "Covid-19, sars and mers: are they closely related?"

[27] "What happens in a clinical trial?" [Online]. Available: https://www.healthline.com/health/clinical-trial-phases

[28] "Clinicaltrials.gov," 2020, accessed 6/3/2020. [Online]. Available: https://clinicaltrials.gov/ct2/results?cond=COVID-19

[29] A. E. Cha, "Forty percent of people with coronavirus infections have no symptoms. might they be the key to ending the pandemic?" Aug. 2020. [Online]. Available: https://www.washingtonpost.com/health/2020/08/08/asymptomatic-coronavirus-covid/

[30] "Vaccination coverage among adults in the united states, national health interview survey, 2017," CDC, Tech. Rep., 2018. [Online]. Available: https://www.cdc.gov/vaccines/imz-managers/coverage/adultvaxview/pubs-resources/NHIS-2017.html

[31] B. Agrawal, "Heterologous immunity: Role in natural and vaccine-induced resistance to infections," *Frontiers in Immunology*, Nov. 2019.

[32] S. M. Miri, F. Noorbakhsh, S. R. Mohebbi, and A. Ghaemi, "Higher prevalence of asymptomatic or mild covid-19 in children, claims and clues," *Journal of Medical Virology*, May 2020.

[33] J. Michaud and J. Kates, "What do we know about children and coronavirus transmission?" Jul. 2020. [Online]. Available: https://www.kff.org/coronavirus-covid-19/issue-brief/what-do-we-know-about-children-and-coronavirus-transmission/

[34] "Immunization coverage," Jul. 2020. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/immunization-coverage

[35] A. P. S. Sato, "What is the importance of vaccine hesitancy in the drop of vaccination coverage in brazil?" *Revista de Saude Publica*, vol. 52, no. 96, Nov. 2018.

[36] "Coronavirus: Brazil passes 100,000 deaths as outbreak shows no sign of easing," Aug. 2020. [Online]. Available: https://www.bbc.com/news/world-latin-america-53712087

[37] D. Sutton, K. Fuchs, M. D'Alton, and D. Goffman, "Universal screening for sars-cov-2 in women admitted for delivery," *New England Journal of Medicine*, vol. 382, no. 22, pp. 2163–2164, 2020. [Online]. Available: https://doi.org/10.1056/NEJMc2009316

[38] Y. B. Tobah, "Which vaccines during pregnancy are recommended and which ones should i avoid?" Feb. 2020. [Online]. Available: https://www.mayoclinic.org/healthy-lifestyle/pregnancy-week-by-week/expert-answers/vaccines-during-pregnancy/faq-20057799

[39] *Health Services Policy Procedure Manual*. [Online]. Available: https://files.nc.gov/ncdps/documents/IC-8\%20Immunizations.pdf

[40] K. R. Sidiq, D. K. Sabir, S. M. Ali, and R. Kodzius, "Does early childhood vaccination protect against covid-19?" *Frontiers in Molecular Biosciences*, Jun. 2020.

[41] S. K. Lau, H. K. Luk, A. C. Wong, K. S. Li, L. Zhu, Z. He, J. Fung, T. T. Chan, K. S. Fung, and P. C. Woo, "Possible bat origin of severe acute respiratory syndrome coronavirus 2," *Emerging Infectious Diseases*, vol. 26, no. 7, Jul. 2020.

[42] B. L. Jacobs, J. O. Langland, K. V. Kibler, K. L. Denzler, S. D. White, S. A. Holechek, S. Wong, T. Huynh, and C. R. Baskin, "Vaccinia virus vaccines: past, present and future," *Antiviral Research*, vol. 84, no. 1, Jun. 2009.

[43] S. W. Wassilew, P. Wutzler, and B. H. Z. S. Group, "Oral brivudin in comparison with acyclovir for improved therapy of herpes zoster in immunocompetent patients: results of a randomized, double-blind, multicentered study," *Antiviral Research*, vol. 59, no. 1, Jun. 2003.

[44] "Penciclovir." [Online]. Available: https://pubchem.ncbi.nlm.nih.gov/compound/Penciclovir