RTX-KG2: a system for building a semantically standardized knowledge graph for translational biomedicine

E. C. Wood[1,*], Amy K. Glen[1,*,c], Lindsey G. Kvarfordt[1], Finn Womack[2], Liliana Acevedo[1], Timothy S. Yoon[1], Chunyu Ma[3], Veronica Flores[1], Meghamala Sinha[1], Yodsawalai Chodpathumwan[7], Arash Termehchy[1], Jared C. Roach[4], Luis Mendoza[4], Andrew S. Hoffman[5], Eric W. Deutsch[4], David Koslicki[2,3,6], and Stephen A. Ramsey[1,8]

[1]School of Electrical Engineering and Computer Science
Oregon State University, Corvallis, Oregon USA

[2]Computer Science and Engineering
Penn State University, State College, Pennsylvania USA

[3]Huck Institutes of the Life Sciences
Penn State University, State College, Pennsylvania USA

[4]Institute for Systems Biology
Seattle, Washington USA

[5]Interdisciplinary Hub for Digitalization and Society
Radboud University, Nijmegen NL

[6]Department of Biology
Penn State University, State College, Pennsylvania USA

[7]King Mongkut's University of Technology North Bangkok, Thailand

[8]Department of Biomedical Sciences
Oregon State University, Corvallis, Oregon USA

[*]These authors contributed equally to the work

[c]Corresponding author: Amy K. Glen, glena@oregonstate.edu

March 25, 2022

Abstract

Background: Biomedical translational science is increasingly leveraging computational reasoning on large repositories of structured knowledge (such as the Unified Medical Language System (UMLS), the Semantic Medline Database (SemMedDB), ChEMBL, DrugBank, and the Small Molecule Pathway Database (SMPDB)) and data in order to facilitate discovery of new therapeutic targets and modalities. Since 2016, the NCATS Biomedical Data Translator project has been working to federate autonomous reasoning agents and knowledge providers within a distributed system for answering translational questions. Within that project and within the field more broadly, there is an urgent need for an open-source framework that can efficiently and reproducibly build an integrated, standards-compliant, and comprehensive biomedical knowledge graph that can be either downloaded in standard serialized form or queried via a public application programming interface (API) that accords with the FAIR data principles.

Results: To create a *knowledge provider* system within the Translator project, we have developed RTX-KG2, an open-source software system for building—and hosting a web API for querying—a biomedical knowledge graph that uses an Extract-Transform-Load (ETL) approach to integrate 70 knowledge sources (including the aforementioned sources) into a single knowledge graph. The semantic layer and schema for RTX-KG2 follow the standard Biolink metamodel to maximize interoperability within Translator. RTX-KG2 is currently being used by multiple Translator reasoning agents, both in its downloadable form and via its SmartAPI-registered web interface. Serializations of RTX-KG2 are available for download in both the pre-canonicalized form and in canonicalized form (in which synonym concepts are merged). The current canonicalized version (KG2.7.3) of RTX-KG2 contains 6.4M concept nodes and 39.3M relationship edges with a rich set of 77 relationship types.

Conclusion: RTX-KG2 is the first open-source knowledge graph of which we are aware that integrates UMLS, SemMedDB, ChEMBL, DrugBank, SMPDB, and 65 additional knowledge sources within a knowledge graph that conforms to the Biolink standard for its semantic layer and schema at the intersections of these databases. RTX-KG2 is publicly available for querying via its API at arax.rtx.ai/api/rtxkg2/v1.2/openapi.json. The code to build RTX-KG2 is publicly available at `github:RTXteam/RTX-KG2`.

Keywords

knowledge graph | biomedical knowledge integration | semantic normalization

1 Background

In biomedical informatics, there is an ongoing need to integrate structured knowledge for translational reasoning, such as for drug repositioning or finding new therapies for monogenic disorders. Progress towards making biomedical knowledge computable has to a large degree tracked advances in information systems. Early steps in the 1950s and 1960s include the framing of clinical reasoning in terms of formal symbolic logic [1]; the creation of Medical Subject Headings (MeSH) [2] for biomedical literature annotation; and the establishment of MEDLINE [3], a database of abstracts that is a cornerstone of today's PubMed. The 1980s

brought progress with the inception of (i) curated online biomedical encyclopedias such as Online Mendelian Inheritance in Man (OMIM) [4]; and (ii) the Unified Medical Language System (UMLS) [5], which integrates knowledge sources into a *metathesaurus* of concepts annotated by semantic types. In the 1990s, the need for interoperability in health-related information systems drove the development of biomedical controlled vocabularies and ontologies [6–9]. The aughts brought frameworks and standards for knowledge representation, such as the Resource Description Framework (RDF) Schema [10], the SemRep natural language-processing algorithm [11], the Web Ontology Language (OWL) [12], the Open Biomedical Ontologies (OBO) standard [13], the BioTop ontology [14], and the Semanticscience Integrated Ontology (SIO) [15]. In 2017, the National Center for Advancing Translational Sciences (NCATS) launched a multi-institution consortium project to develop a universal Biomedical Data Translator [16] (the "Translator"), a distributed computational reasoning and knowledge exploration system for translational science. More recently, the Biolink metamodel [17, 18] advanced the field by (i) providing comprehensive mappings of semantic types and relation types to other ontologies; (ii) standardizing and ranking preferred identifier types for various biological entities; and (iii) providing structured hierarchies of relation types and concept types needed to provide a universal semantic layer for biological knowledge graphs. Knowledge cross-linking has been accelerated by the establishment of ontology repositories and portals like OBO Foundry [19] and the National Center for Biomedical Ontology (NCBO) [20]. Concomitantly, the World Wide Web has fueled the development of online knowledge-bases updated by curation teams, such as the Kyoto Encyclopedia of Genes and Genomes (KEGG) [21], PubChem [22], DrugBank [23], ChEMBL [24], UniProt Knowledgebase (UniProtKB) [25], the Small Molecule Pathway Database (SMPDB) [26, 27], and Reactome [28]. The last ten years brought large-scale natural-language processing (NLP) of biomedical literature, including the establishment of SemMedDB [29], a knowledge-base extracted by SemRep analysis of PubMed abstracts. The crowd-sourcing of literature curation and the use of NLP together drove tremendous growth of structured biomedical knowledge-bases, albeit in forms that are not semantically interoperable due to the use of different systems of concept identifiers, semantic types, and relationship types.

In the last decade, there have been numerous efforts to address the siloing and lack of semantic interoperability of structured biomedical knowledge. BIOZON [30], BioGraphDB [31], and DRKG [32] used custom graph schemas and used standard sets of identifier types, with BIOZON introducing a hierarchy of relationship types. The Bio4j system [33] provided a graph query language with type-checking aligned with their custom schema. The Bio2RDF knowledge repository [34] uses RDF and SIO for linking biomedical knowledge, while largely retaining concept source vocabularies [35]. KaBOB [36], in contrast, uses OBO ontologies as a common vocabulary. The Monarch Initiative [35, 37, 38] similarly leverages existing ontologies such as Relation Ontology (RO) [39] and the Open Biomedical Annotations ontology (OBAN), while using custom concept types. Monarch advanced the field by providing a clique detection-based method for identifying semantically identical concepts (known as graph "canonicalization" [40]). The Hetionet project [41] developed concept types and relationship types specifically for knowledge representation for drug repurposing; these types were expanded in the Scalable Precision medicine Knowledge Engine (SPOKE) database [42, 43]. CROssBAR-DB [44] keeps its source datasets separate from one another and provides an interface for constructing a query-specific, integrated knowledge graph. The Reasoning Over Biomedical Objects linked in Knowledge Oriented Pathways (ROBOKOP) graph [45, 46] uses concept and relationship types from the Biolink

metamodel. The EpiGraphDB system [47] uses SemMedDB semantic types for concepts and source-specific relation types. The BioThings framework is unique in that it provides a single Web-based application programming interface (API) (with a unified semantic model) that proxies queries to many different knowledge source APIs. BioThings leverages the SmartAPI system [48] for registering and documenting knowledge source APIs using standardized metadata. Prior to the present work, as a part of the Biomedical Data Translator project [16], we developed an open-source biomedical knowledge graph called RTX-KG1 [49]; we found that using Biolink for its semantic layer facilitated interoperability by eliminating the need for translation software layers and by allowing systems to use Biolink at the level of granularity appropriate to their application. Notably, Biolink has been adapted as the semantic layer for concepts and relations for knowledge representation within the Translator project. A second finding from our work on RTX-KG1 was the importance of providing both a pre-canonicalized version and canonicalized version (see Sec. 2.3) of the knowledge graph. To date, biomedical knowledge graphs of which we are aware are either canonicalized or standardize on an identifier type for each semantic type, rather than providing both canonicalized and pre-canonicalized graphs; the latter form is important in order to support users that wish to apply their own canonicalization algorithm.

Previous efforts to develop integrated biomedical knowledge systems have used a variety of database types, architectural patterns, and automation frameworks. For persistence, knowledge systems have used relational databases [30], distributed graph databases [33], multimodal NoSQL databases [31], RDF triple-stores [34, 36], document databases [35], document-oriented databases [44, 50], and—with increasing frequency [35, 41, 42, 45, 47, 49]—the open-source graph database Neo4j (`github:neo4j/neo4j`). Knowledge systems have also differed in terms of the ingestion method used in their construction; many systems [31, 34–36, 44] utilized an extract-transform-load (ETL) approach, whereas others [45, 49, 50] used API endpoints to query upstream knowledge sources; one [47] blended both ETL and API approaches for knowledge graph construction. While both approaches have their strengths, from our work on the predecessor RTX-KG1 system and from the present work, we found that an ETL approach has significant advantages in terms of scalability, reproducibility, and reliability. In terms of automation frameworks, previous efforts have used general-purpose scripting languages [34, 36, 41, 42, 45, 49, 51], batch frameworks [44], declarative rule-based build frameworks [31, 33, 52], and parallel build systems such as Snakemake [53] (EpiGraphDB). Liu et al. reported [52] choosing the Snakemake [53] build framework specifically because of its high performance (i.e., parallel capabilities). While previous efforts have resulted in biomedical knowledge graphs incorporating (individually) UMLS, SemMedDB, multiple major drug knowledge bases (such as ChEMBL and DrugBank), a standards-compliant semantic layer, and a high-performance build system, so far as we are aware, none have incorporated all of these features in a single system providing both canonicalized and pre-canonicalized graphs.

Introduction

As a successor to RTX-KG1 [49], we have developed RTX-KG2, an open-source biomedical knowledge-base representing biomedical concepts and their relationships. RTX-KG2 is integrated from 70 sources—including the major sources UMLS, SemMedDB, ChEMBL, DrugBank, SMPDB, Reactome, KEGG, and UniProtKB—and its semantic layer is based on the Biolink metamodel [17, 18]. To accommodate multiple use-cases, the RTX-KG2 build system produces

two knowledge graphs: a precursor knowledge graph (RTX-KG2pre) in which equivalent concepts described using different identifier systems are not identified as a single node; and a canonicalized knowledge graph (RTX-KG2c) in which equivalent concepts described using different identifier systems are identified as a single node. Both RTX-KG2pre and RTX-KG2c are directed multigraphs with node and edge annotations. The software repository for RTX-KG2, including all code to build the database, is publicly available at the `github:RTXteam/RTX-KG2` GitHub project. Users can access RTX-KG2 content via any of three channels: (i) a single-file download version of the canonicalized RTX-KG2 knowledge graph (KG2c) (or, if needed, the pre-canonicalized RTX-KG2pre knowledge graph); (ii) a publicly accessible, SmartAPI-registered API for querying RTX-KG2; and (iii) a web browser interface for querying RTX-KG2. RTX-KG2 uses an ETL approach for knowledge graph construction and it automates builds using Snakemake; together, these enable efficient knowledge graph construction. RTX-KG2 is a built-in knowledge database for ARAX (Autonomous Relay Agent X) [54], a Web-based computational biomedical reasoning system that our team is also developing for answering translational science questions such as questions related to drug repositioning, identifying new therapeutic targets, and understanding drug mechanisms-of-action. We are developing RTX-KG2 and ARAX as a part of the NCATS Translator project. Here, we enumerate the knowledge sources that are incorporated into RTX-KG2 (Sec. 2.1); outline the processes for building RTX-KG2pre from its upstream knowledge sources (Sec. 2.2) and for building the canonicalized RTX-KG2c (Sec. 2.3); describe the schema for RTX-KG2 (Sec. 2.4); describe the RTX-KG2 build system software (Sec. 2.8); provide statistics about the size and semantic breadth of RTX-KG2 (Sec. 2.6); and discuss how RTX-KG2 is being used as a standalone knowledge-base for translational reasoning as well as in conjunction with the ARAX system (Sec. 3).

2 Construction and Content

In this section, we describe how RTX-KG2 is constructed; provide an overview of its graph database schema; and summarize its content in terms of sources, semantic breadth, and size. The overall build process, along with the various outputs of RTX-KG2, is depicted in Figure 1. Broadly speaking, the RTX-KG2 build system does four things: it (i) loads information from source databases (blue triangles in Fig. 1) via the World Wide Web as described in Section 2.1); (ii) integrates the knowledge into a precursor knowledge graph called RTX-KG2pre (upper green hexagon in Fig. 1) and hosts it in a Neo4j database (upper orange cloud in Fig. 1) as described in Section 2.2; (iii) coalesces equivalent concept nodes into a canonicalized knowledge graph called RTX-KG2c (brown circle in Fig. 1) as described in Section 2.3, with a schema that is described in Section 2.4; and (iv) provides various knowledge graph artifacts and services as described in Section 2.6. We provide technical details of the RTX-KG2 build system in Section 2.8.

2.1 Sources and their file formats

Of the 50 RDF-based sources, the system ingests 27 in Terse RDF Triple Language (TTL [56]) format and 23 as OWL ontologies in RDF/XML format [57] (which we abbreviate here as "OWL"). Of the 27 TTL sources, 26 are from the UMLS, obtained as described in Sec. 2.8.3; the remaining TTL source is the Biolink metamodel, which defines the semantic layer for RTX-KG2, including hierarchies of concept types and relation types (see Sec. 2.6). In addition to concept

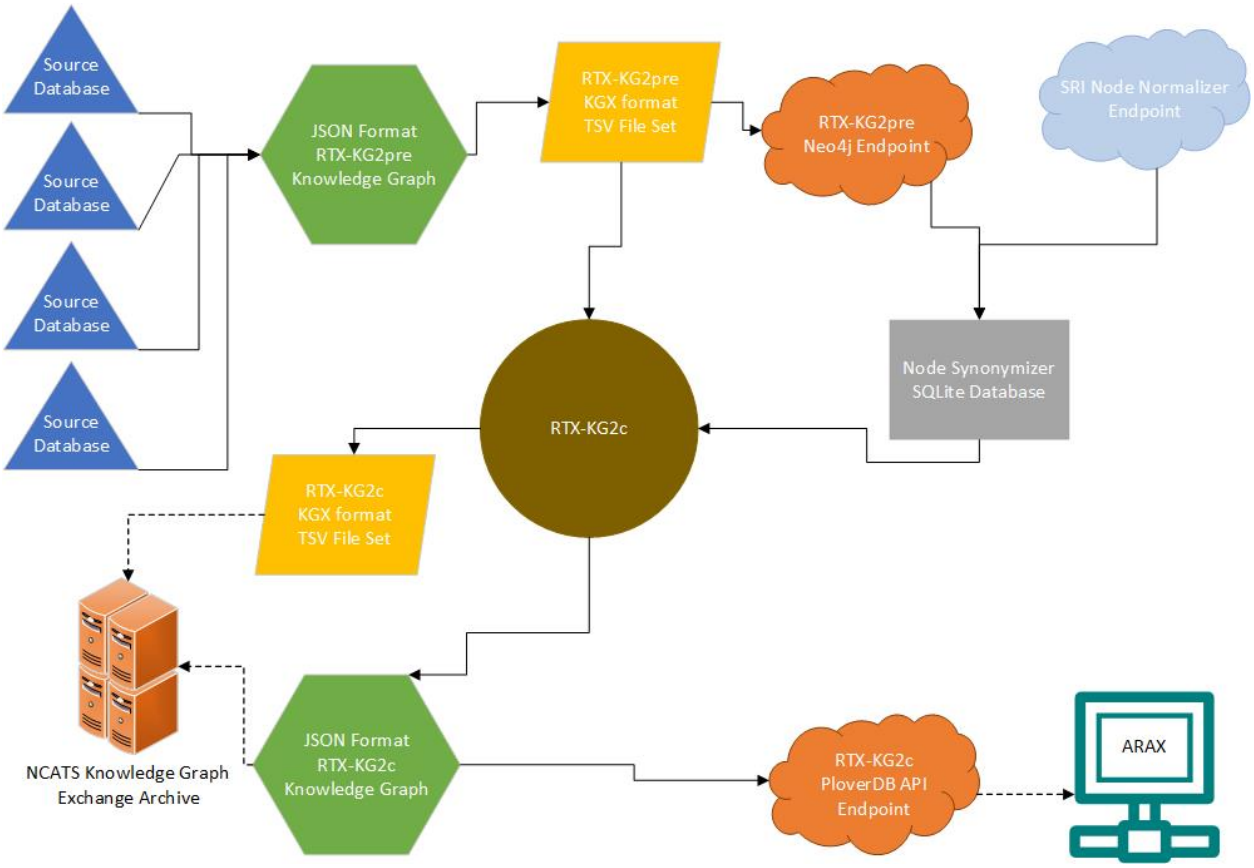


Figure 1: Overall Workflow of RTX-KG2. Blue triangle: individual external source; light blue cloud: external API endpoint; yellow parallelogram: TSV file-set; green hexagon: JSON File; orange cloud: API endpoint output; grey rectangle: SQLite [55] database; brown circle: abstract object-model representation of KG2c; turquoise computer: user/client computer; orange server: Translator knowledge graph exchange (KGE) server.

type and relation type hierarchies, the Biolink metamodel provides equivalence mappings of the Biolink types to classes in other high-level ontologies (such as `biolink:Gene` being equivalent to `SIO:010035`) and of the Biolink concept types to prioritized lists of identifier types for the concept type[1]. Each knowledge source's concepts are assigned Biolink concept semantic types—which are called "categories" in the Biolink metamodel—and relationships are assigned Biolink relationship types at the time that the source is ingested. All but two of the 23 OWL-format sources are ontologies from the OBO Foundry; the remaining two OWL-format sources are the Experimental Factor Ontology (EFO) [58] and Orphanet Rare Disease Ontology [59].

In contrast to the RDF-based method which ingests sources in only TTL and OWL formats, the direct-to-JSON method ingests sources in a variety of file formats (JSON, Structured Query Language (SQL), tab-separated value (TSV), Extensible Markup Language (XML), Gene Product Association (GPA), and SWISS-PROT-like DAT format). One source, KEGG, is queried via an API (rather than using an ETL approach) due to license restrictions, and then saved to JSON. For the 20 direct-to-JSON sources, the RTX-KG2 system has one ETL module for each source, with each script producing a source-specific JSON file according to the RTX-KG2 JSON schema (Sec. 2.4). In contrast, for the 50 RDF-based sources, the system has a single ETL module for ingesting all of the sources together. The RDF-based method merges all of the OWL and TTL sources and generates a single JSON file. The hybrid design for the ETL layer for RTX-KG2 balances the benefits of modularity (where it is feasible in the direct-to-JSON method) with the need for a monolithic ingestion module for ontologies due to their extensive use of inter-ontology axioms [60]. RTX-KG2 integrates 70 knowledge sources (Table 1), 50 of them via a *resource description framework (RDF)-based* ingestion method and 20 of them via a *direct-to-JSON* ingestion method.

[1] An example prioritization would be for the semantic type "gene", to prefer identifier types from Ensembl Gene, National Center for Biotechnology Information (NCBI Gene), and Human Gene Nomenclature Committee (HGNC).

Table 1: RTX-KG2 integrates 70 knowledge sources into a single graph. Each row represents a server site from which sources were downloaded. Columns as follows: *Name,* the short name(s) of the knowledge sources obtained or the distribution name in the cases of UMLS and OBO Foundry; #, the number of individual sources or ontologies obtained from that server; *Format*, the file format used for ingestion (see below); *Method*, the ingestion method used for the source, either `D2J` for direct-to-JSON or `RBM` for the RDF-based method. File format codes: `CSV`, comma-separated value; `DAT`, SWISS-PROT-like DAT format; `JSON`, JavaScript object notation; `OWL`, OWL in RDF/XML [57] syntax; `RRF`, UMLS Rich Release Format [61]; `SQL`, structured query language (SQL) dump; `TSV`, tab-separated value; `XML`, extensible markup language.

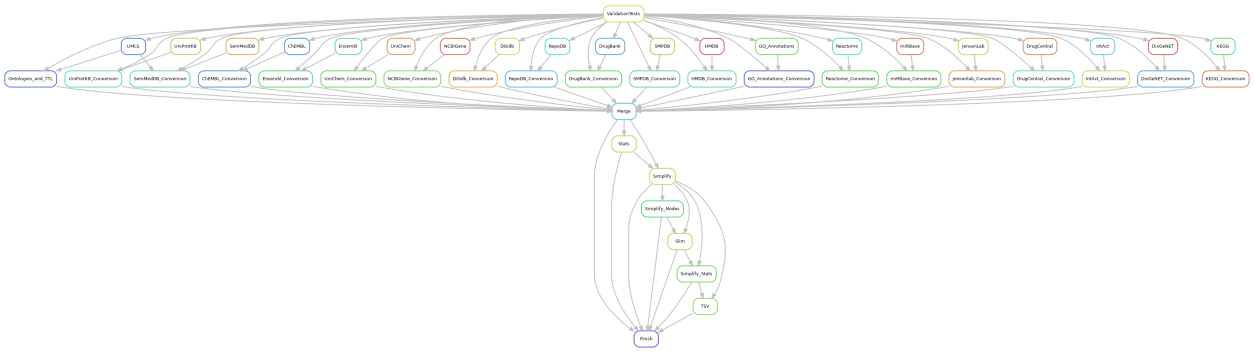
Name	#	Description	Format	Method
Biolink [17, 18]	1	Biolink Metamodel (semantic layer)	TTL	RBM
ChEMBL [24, 62]	1	EMBL Chemogenomic Database	SQL	D2J
DGIdb [63]	1	Drug Gene Interaction Database	TSV	D2J
DisGeNET [64]	1	Disease-Gene Associations	TSV	D2J
DrugBank [23]	1	Pharmaceutical Knowledge Base	XML	D2J
DrugCentral [65]	1	Online drug Compendium	SQL	D2J
Ensembl Gene [66]	1	Ensembl Human Gene annotations	JSON	D2J
EFO [58]	1	Experimental Factor Ontology	OWL	RBM
GO [67, 68]	1	Gene Ontology annotations	TSV	D2J
HMDB [69–72]	1	Human Metabolite Database	XML	D2J
IntAct [73, 74]	1	IntAct Molecular Interaction Database	TSV	D2J
Jensen Lab Diseases [75]	1	Gene to Diseases Dataset	TSV	D2J
KEGG [21, 76, 77]	1	Kyoto Encyclopedia of Genes and Genomes	API	D2J
miRBase [78–82]	1	MicroRNAs Dataset	DAT	D2J
NCBI Gene [83]	1	NCBI Human Gene annotations	TSV	D2J
OBO Foundry	21	OBO Foundry Ontologies (Table S1)	OWL	RBM
Orphanet [84]	1	Orphanet Rare Disease Ontology	OWL	RBM
PathBank [85–87]	1	Wishart Lab Pathway Databases	XML	D2J
Reactome [88]	1	Pathway Database	SQL	D2J
SemMedDB [29]	1	Semantic Medline Database	SQL	D2J
SMPDB [26, 27]	1	Small Molecule Pathway Database	CSV	D2J
UMLS [89]	26	Unified Medical Language System (Table 5)	TTL	RBM
UniChem [90]	1	EBI Small Molecule Cross-refs	TSV	D2J
UniProtKB [25]	1	UniProt Knowledge Base	DAT	D2J
Total	70			

2.2 Building RTX-KG2pre from upstream sources

The process by which the RTX-KG2 system builds its knowledge graph from its 70 sources—the first stage of which is diagrammed in Fig. 2)—begins by executing validation scripts (the "`validationTests`" task in Fig 2) that ensure that the identifiers used in the RTX-KG2 semantic layer are syntactically and semantically correct within the Biolink metamodel. Next, the build process executes in parallel the 21 direct-to-JSON and RDF-based ETL scripts (see second and

third rows in Fig. 2 and Sec. 2.1) to produce a total of 21 JSON files (20 via the direct-to-JSON method and one via the RDF-based method) in the RTX-KG2 knowledge graph schema (described in Sec. 2.4). Next, those JSON files are loaded and their object models are merged (via the "`Merge`" task) into a single self-contained graph that is then saved in an RTX-KG2-schema JSON file in which relationships consist of triples (subject, relation, object) where the relation is from any number of source-specific relationship vocabularies (totalling 1,228 source-specific relationship types in all). The RTX-KG2 object model is then simplified (via the "`Simplify`" task) by consolidating redundant relationships; a redundant relationship is where two or more sources assert the same triple (`ibuprofen`, `treats`, `headache`), in which case, the multiple relationships with identical triples are merged into a single relationship, with the multiple sources noted in the list-type `provided-by` relationship attribute (see Sec. 2.4). Also in the `Simplify` task, relationship types are simplified by mapping each of them to one of 77 standardized relationship types (called "predicates") in the Biolink metamodel. The `Simplify` task also maps source identifiers to Biolink "Information Resource" identifiers. In general, the `Simplify` task standardizes the graph with the Biolink metamodel standards [17, 18]. We call the resulting graph *RTX-KG2pre* in order to emphasize that it is the precursor graph to the canonicalized RTX-KG2 graph (described below). In the final step of the build process, the nodes and edges of

Figure 2: Flowchart of tasks for building RTX-KG2pre (the precursor stage of RTX-KG2) from 21 upstream knowledge-base distributions.



RTX-KG2pre are imported into a Neo4j graph database (for details, see Sec. 2.8.2).

2.3 Building RTX-KG2c, the canonicalized version of RTX-KG2

Because the various ontologies that RTX-KG2pre ingests often represent the same concept using multiple different identifiers, some of the nodes in RTX-KG2pre represent equivalent concepts. For example, the concept of Parkinson's disease is represented by several nodes in RTX-KG2pre, such as the nodes with identifiers `MONDO:0005180`, `DOID:14330`, `EFO:0002508`, and `MESH:D010300`, many of which are connected in RTX-KG2pre with relationships of type `biolink:same_as` or non-transitive generalizations of that relationship type. In our work on the predecessor knowledge system RTX-KG1, we found—consistent with reports from other teams engaged in translational reasoning—that coalescing nodes for semantically equivalent concepts into single nodes facilitates reasoning by reducing the complexity of graph paths that represent answers for common translational questions. Thus, to enhance the utility of RTX-KG2 for translational reasoning, we

created a version of RTX-KG2 called RTX-KG2canonicalized (RTX-KG2c) in which semantically equivalent nodes are coalesced to a single concept node. RTX-KG2c has its own automated Python-based build process with similar hardware requirements to the RTX-KG2pre build process. In brief, building RTX-KG2c from RTX-KG2pre proceeds in five steps:

1. RTX-KG2pre nodes and edges are loaded from the RTX-KG2pre TSV files;
2. the set of nodes is partitioned into disjoint subsets of equivalent nodes;
3. from each group of equivalent nodes, a canonical node is chosen, added to RTX-KG2c, and decorated with the identifiers of its synonymous nodes (along with other information);
4. edges from RTX-KG2pre are remapped to refer only to canonical identifiers;
5. edges with the same subject, object, and predicate are merged.

For Steps 2–3, the RTX-KG2 build system uses the ARAX [54] system's *Node Synonymizer* service, which takes into account four sources of evidence in the following order: (i) concept equivalence information obtained dynamically by querying a Translator web service API called the Standards and Reference Implementations (SRI) Node Normalizer (`github:TranslatorSRI/NodeNormalization`); (ii) `biolink:same_as` edges in RTX-KG2pre between RTX-KG2pre nodes; (iii) human-recognizable node (concept) name equivalence; and (iv) node semantic type compatibility. For Step 2, the Node Synonymizer goes through three passes of merging concepts in order to ensure that the partitioning of nodes is independent of the order in which the nodes are loaded into the Node Synonymizer. For Step 3, the Node Synonymizer uses a score-based system that flexibly enables incorporation of new heuristics. Compared to the numbers of nodes and edges in RTX-KG2pre, the process of canonicalization reduces the number of nodes by approximately 62% and edges by 73%. The RTX-KG2c graph is serialized in JSON format (see Sec. 2.4) and archived in an AWS S3 bucket and in GitHub. From the latter, it is imported into a custom-built in-memory graph database, PloverDB (`github:RTXteam/PloverDB`).

2.4 RTX-KG2 schema and RTX-KG2pre Biolink compliance

The RTX-KG2 knowledge graph follows the Biolink metamodel (version 2.1.0) for its semantic layer and (in RTX-KG2pre) its schema. RTX-KG2 uses Biolink's category hierarchy for its concept (node) types (Figure 3) and Biolink's predicate hierarchy for its relationship (edge) types (Figure 4). When mapping terms from their original sources to the Biolink terminology, the RTX-KG2 build system consults the Biolink model's internal mappings in order to detect any inconsistencies between the two. Because relationship terms that are highly specific are often mapped to more generalized terminology, the original source's phrasing is preserved in the `relation` property[2]. In addition to mapping upstream source relations to Biolink predicates, the RTX-KG2 build process coalesces edges that have the same end nodes and the same predicate (it does, however, preserve the provenance information from both of the coalesced edges). The schema of the JSON serialization of RTX-KG2pre is documented in detail in the RTX-KG2 project area `github:RTXteam/RTX-KG2`. In brief, RTX-KG2pre is serialized as a JSON object with

[2]This will be transitioning to the `original_predicate` property in the next release of RTX-KG2, for compatibility with recent changes in the Biolink standard.

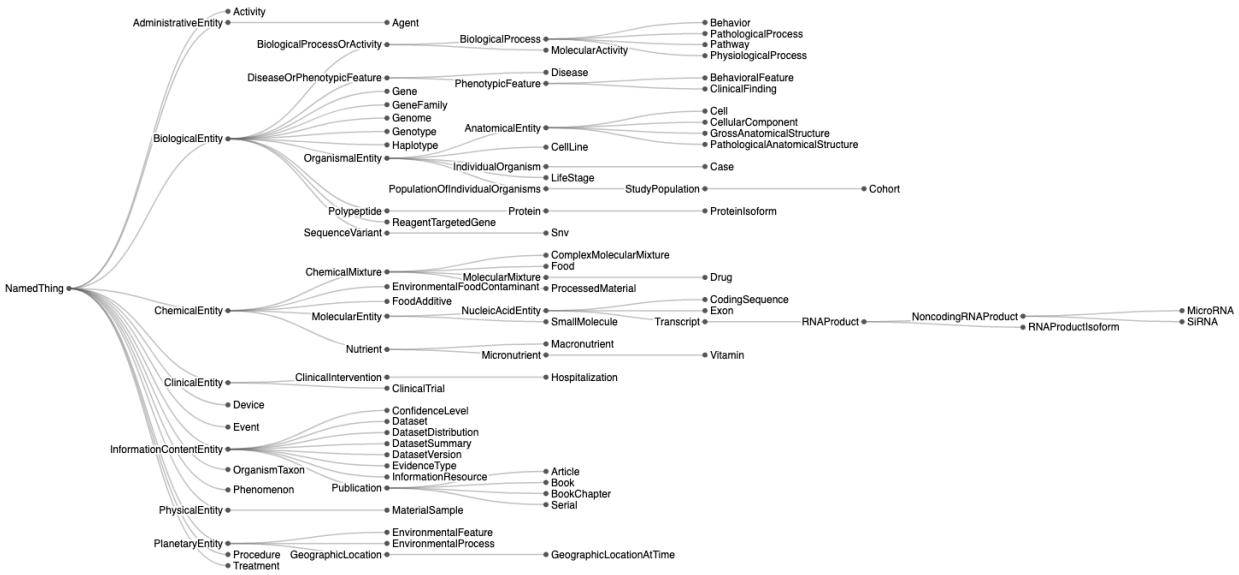


Figure 3: Node concept types in RTX-KG2 are based on the Biolink metamodel [17, 18].

keys `build`, `nodes`, and `edges`. Under the `build` key, a JSON object stores information about the version of RTX-KG2pre and timestamp of the build. Under the `nodes` key is a list containing a JSON object for each node. Each node object contains 16 keys corresponding to the node property types in RTX-KG2pre, detailed in Table 2. The `id` node property contains a compact representation of the canonical uniform resource identifier, i.e., a CURIE identifier [91]. The `category` property of a node describes the node's semantic type, such as `biolink:Gene`. Similarly, the `edges` key is a list containing a JSON object for each edge, with the edge JSON object containing the keys indicated in Table 3.

The schema of the JSON serialization of RTX-KG2c is very close to that of RTX-KG2pre except that the former does not contain the top-level `build` key/object and, for each node object, RTX-KG2c contains some additional keys such as `equivalent_curies`, which enumerates the CURIE IDs of the nodes representing concepts that were semantically identified in the canonicalization step; `all_names`, which contains the `name` properties of the KG2pre nodes that were canonicalized together for the given KG2c node; and `all_categories`, which contains the `category` properties of the nodes that were canonicalized together for the given KG2c node. In addition, the `ids` of the corresponding KG2pre edges that each KG2c edge was created from are documented in the `kg2_ids` property.

2.5 Quality control and reproducibility

The RTX-KG2 build process incorporates multiple layers of quality control, including both automated and manual procedures. As the first step in the build process (in Fig. 2, the `ValidationTests` step), python scripts validate the consistency of the RTX-KG2 semantic layer with the Biolink metamodel. During knowledge integration (in Fig. 2, `Merge`), relationships whose subject or object nodes are not present in the knowledge graph are logged for offline investigation. A report of statistics on the RTX-KG2 knowledge graph is generated—including (1) node counts

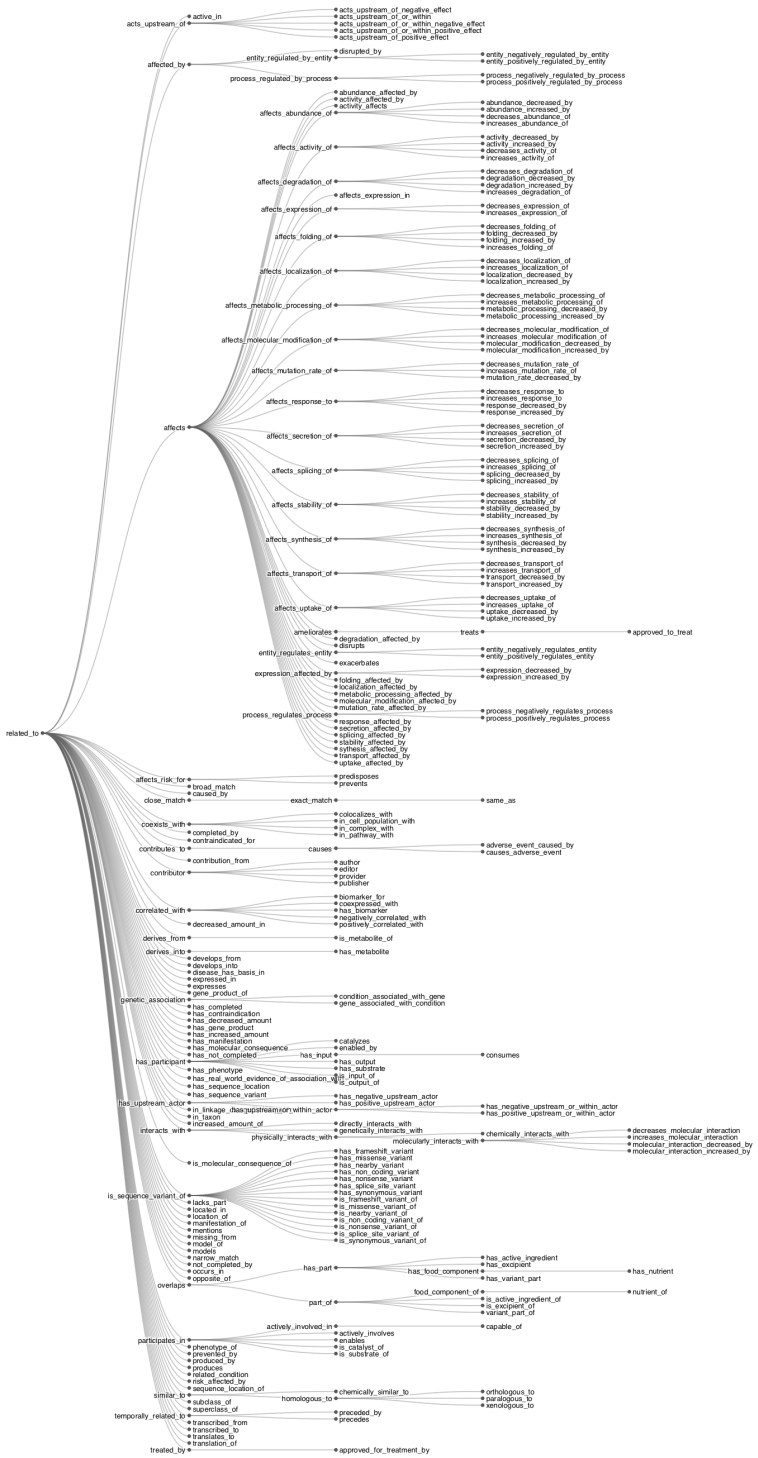


Figure 4: Edge predicate types in RTX-KG2 are based on the Biolink metamodel.

Table 2: Node properties in RTX-KG2pre and RTX-KG2c

	KG2pre	KG2c
all_categories		✓
all_names		✓
category	✓	✓
category_label	✓	
creation_date	✓	
deprecated	✓	
description	✓	✓
equivalent_curies		✓
full_name	✓	
has_biological_sequence	✓	
id	✓	✓
iri	✓	✓
knowledge_source	✓	
name	✓	✓
provided_by	✓	
publications	✓	✓
replaced_by	✓	
synonym	✓	
update_date	✓	

Table 3: Edge properties in RTX-KG2pre and RTX-KG2c

	KG2pre	KG2c
id	✓	✓
kg2_ids		✓
knowledge_source	✓	✓
negated	✓	
object	✓	✓
predicate	✓	✓
predicate_label	✓	
provided_by	✓	
publications	✓	✓
publications_info	✓	✓
relation	✓	
relation_label	✓	
subject	✓	✓
update_date	✓	

by knowledge source and by semantic type and (2) edge counts by source and by relationship type—both before (in Fig. 2, `Stats`) and after (in Fig. 2, `Simplify_Stats`) redundant edges are joined in the merge process (in Fig. 2, `Simplify`). The procedure for RTX-KG2 builds includes a script-facilitated comparison of that report for the new build with the equivalent report for the previous build, in order to flag anomalously large changes in node or edge counts conditioned on source, category, or predicate. Once a new RTX-KG2c build is installed into the RTX-KG2 query API, functional correctness of the live API is verified using a python-based test suite. Reproducibility of the RTX-KG2 build is enhanced by the intentional choice of using an ETL approach based on flat-file exports knowledge-source databases. To aid with versioning, most source databases have their version stored on their metanode in RTX-KG2.

2.6 RTX-KG2 content and statistics

The latest released version of RTX-KG2pre as of this writing, RTX-KG2.7.3, contains 10.2 million nodes and 54.0 million edges. Each edge is labeled with one of 77 distinct predicates (Biolink relationship types) and each node with one of 56 distinct categories (Biolink concept semantic types). In terms of frequency distribution, there is over six decades of variation across node categories (Fig. 5) and edge predicates (Fig. 6), with the dominant category being `OrganismTaxon` (reflecting the significant size of the NCBI organism classification ontology [92]) and the dominant predicate being `has_participant` (reflecting the significant size of the PathBank database [85]). Figure 6 shows a breakdown of edges in KG2.7.3 by their Biolink predicate. KG2.7.3c contains 6.4 million nodes and 39.3 million edges, which is approximately 62% of the nodes and 73% of the edges of KG2.7.3pre. Figure 7 shows node neighbor counts by category for the top 20 most common categories in RTX-KGc.

In terms of their total (i.e., in+out) vertex degree distributions, both KG2pre and KG2c appear to be approximately scale-free (Figure 8) with a power law exponent of 2.43, meaning that the frequency of concepts with connectivity k decreases as $\sim k^{-2.43}$. Figure 7 highlights the frequencies of various combinations of subject node category and object node category appearing together in edges in KG2c, indicating (1) high levels of cross-category axioms among "molecular entity", "small molecule", and "chemical entity" and (2) high levels of connections between "pathway" and "molecular entity", "small molecule", "molecular activity", "organism taxon", "anatomical entity", and "transcript". Note that the category-category frequency heatmap is not expected to be symmetric for a knowledge graph (such as RTX-KG2) with a high proportion of relationship types that have non-reflexive subject-object semantics.

2.6.1 Comparison to other knowledge graphs

To objectively evaluate the size and semantic richness of RTX-KG2 in comparison to other biomedical knowledge graphs, we compared it to four other knowledge graphs that are in active use for translational applications: Hetionet [41], SPOKE [42, 43], the SRI Reference Knowledge Graph [93], and ROBOKOP [45, 46]. For the meta-triples counts, we accessed the ROBOKOP and SPOKE graphs via their SmartAPI-registered TRAPI endpoints on March 15, 2022. For Hetionet and ROBOKOP, we used published node and edge counts [41, 46]. For the SRI Reference Knowledge Graph, we downloaded the graph in KGX TSV format from the Biomedical Data Translator Knowledge Graph Exchange (KGE) and analyzed the graph locally.

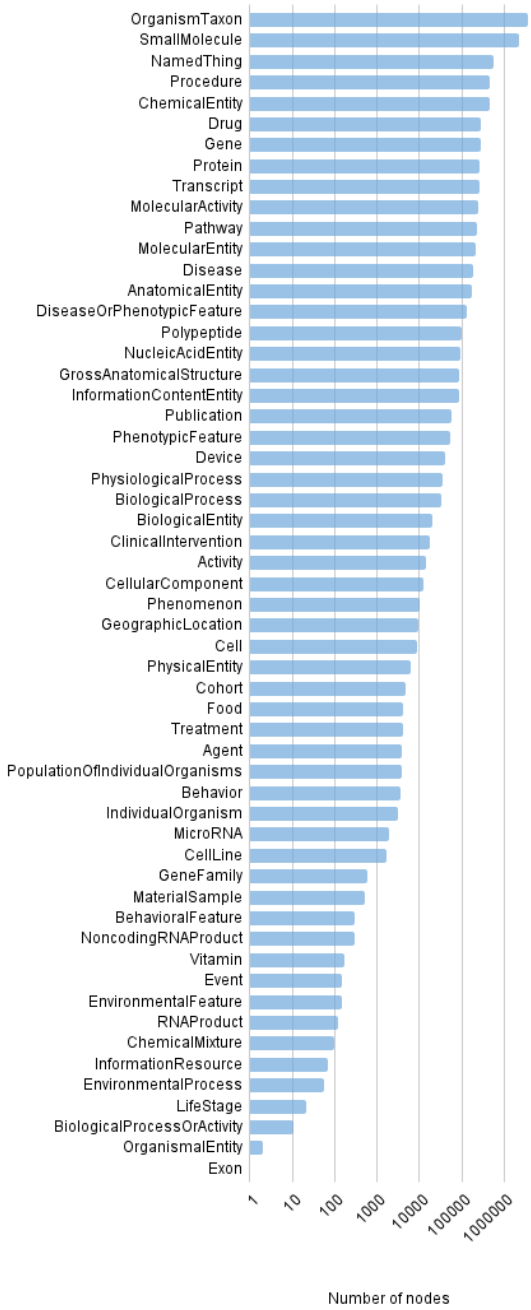


Figure 5: Number of nodes in RTX-KG2.7.3pre by category.

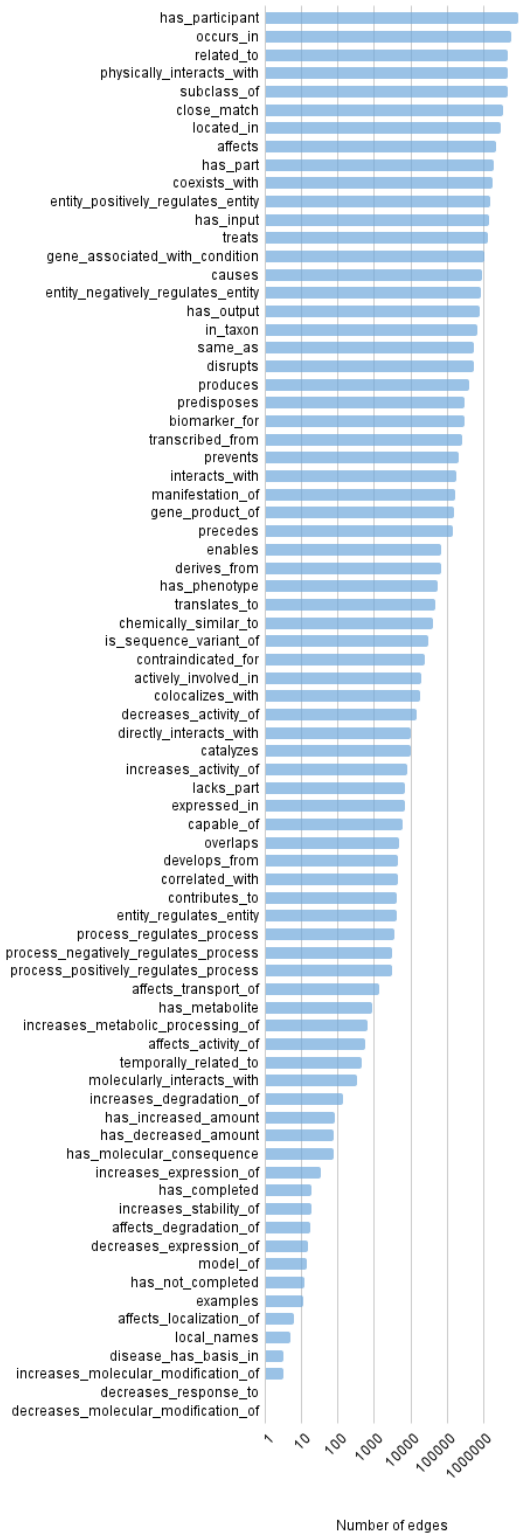


Figure 6: Number of edges in RTX-KG2.7.3pre by predicate.

Figure 7: Node neighbor counts by category for the top 20 most common categories in RTX-KG2.7.3c. Each cell captures the number of distinct pairs of neighbors with the specified subject and object categories.

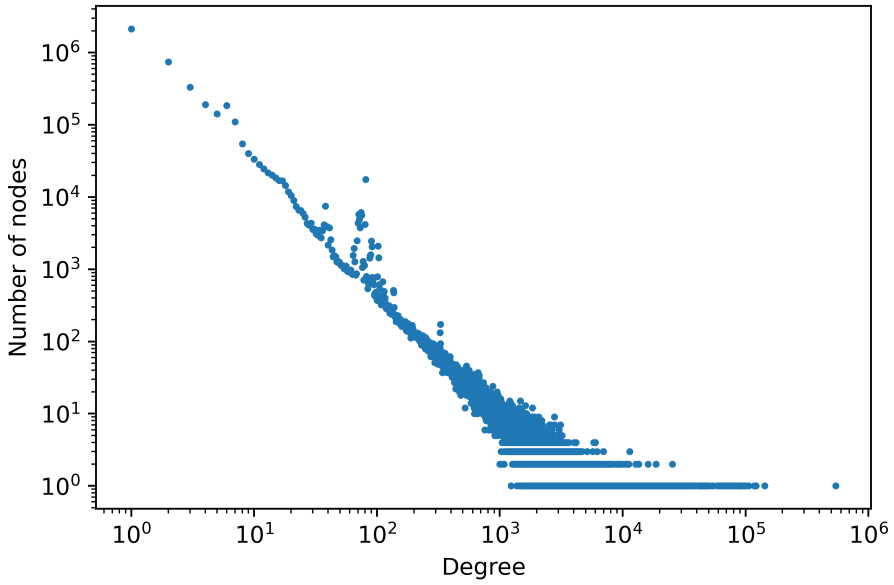


Figure 8: KG2.7.3c has a scale-free degree distribution.

2.7 RTX-KG2 access channels

The complete software code for building RTX-KG2 and for hosting an indexed RTX-KG2 graph database in Neo4j is publicly available in an open-source repository (see Sec. 6.3). In addition, the content of the latest RTX-KG2 graphs that we have built can be easily accessed via three different channels (see Sec. 6.3), depending on the use-case: (i) serialized flat-file download; (ii) REpresentational State Transfer (REST) [94] API (i.e., a web API); or (iii) web browser user interface, through the ARAX system. Tab-separated value (TSV) serializations of RTX-KG2pre and JSON serializations of RTX-KG2c are available in a public GitHub repository (see Sec. 6.3) via the `git-lfs` file hosting mechanism, and their schemas are documented as described in Sec. 2.4 and in the RTX-KG2 documentation sections that are linked therein. RTX-KG2c can be queried via a REST API that implements the Translator API, or "TRAPI" specification (`github:NCATSTranslator/ReasonerAPI`) and that is registered via the SmartAPI [48] framework and therefore discoverable using SmartAPI-associated tooling such as Biothings Explorer [50]. The RTX-KG2 API enables both one-hop and multi-hop querying of the knowledge graph; queries are internally serviced by the PloverDB in-memory graph database (see Sec. 2.3). Further, RTX-KG2c is archived in Biolink Knowledge Graph eXchange [17] TSV format (documented at `github:biolink/kgx`) through the Knowledge Graph Exchange (KGE; see Figure 1) archive and registry system for the NCATS Biomedical Data Translator project (`github:NCATSTranslator/Knowledge_Graph_Exchange_Registry`) (currently in testing phase).

Table 4: Upstream source files that must be staged in S3 in order to build RTX-KG2

DrugBank	XML Download	Requires browser to download
RepoDB	TSV Download	Requires browser to download
SemMedDB	MySQL Download	Requires browser to download
SMPDB Pubmed IDs	CSV Download	Obtained courtesy of Wishart Lab
UMLS Metathesaurus	ZIP Download	Requires browser to download

2.8 RTX-KG2 build system and software

2.8.1 Requirements

The software for building RTX-KG2pre is designed to run in the Ubuntu Linux version 18.04 operating system on a dedicated system with at least 256 GiB of memory, 1 TiB of disk space in the root file system, $\geq$ 1 Gb/s networking, and at least 20 cores (we use an Amazon Web Services (AWS) Elastic Compute Cloud (EC2) instance of type `r5a.8xlarge`). The software for building RTX-KG2 makes use of AWS Simple Storage Service (S3) for network storage of both build artifacts and input knowledge source distribution files that cannot be retrieved by a scripted HTTP GET from their respective providers (see Table 4). These build files must be pre-staged in an AWS S3 bucket before the build process for RTX-KG2pre is started. For hosting RTX-KG2 in a Neo4j server, the system requirements are 64 GiB of system memory, 8 virtual CPUs, and $\sim$200 GiB of root filesystem storage (we use a `r5a.2xlarge` instance).

2.8.2 RTX-KG2 uses Snakemake for building RTX-KG2pre

RTX-KG2pre is built by a series of Python modules and bash scripts that extract, transform and load (ETL) 45 data downloads (corresponding to the rows of Table 1, with the "OBO Foundry" row counting for 21 separate downloads) from 24 source websites (Sec. 2.1) into a standardized property knowledge graph format integrated with the Biolink metamodel as the semantic layer. To maximize the reproducibility of RTX-KG2 builds, the build system is fully automated, including scripts for (i) setting up and configuring the build system to run, (ii) downloading and transforming data, and (iii) exporting the final graph to the graph database Neo4j. RTX-KG2 utilizes the Snakemake [53] workflow management tool to schedule multicore execution of the RTX-KG2pre build process. In addition to reducing the computational costs of the build and the amount of time it takes to run, Snakemake increases modularity by enabling individual components (and their upstream dependencies) to be executed, when necessary. This is particularly useful for allowing failed builds to resume at the point of failure (via a so-called "partial" build), once the root cause (which could be a parsing error from an upstream ontology, for example) has been fixed.

The build process starts with parallel source extractions, in which all of the source databases are downloaded and prepared for the format that their respective conversion script uses. Then, each upstream source's dataset is processed by a Python conversion module. This converts each source's data into the RTX-KG2pre JSON format (Sec. 2.4). Once all of the upstream data sources are converted into their RTX-KG2pre JSON file, a module merges all of them into a cohesive graph, such that no two nodes have the same CURIE ID. One of the challenges in this step is when different upstream sources provide different names for the same concept CURIE ID;

the RTX-KG2pre build system addresses such name conflicts by having a defined order of precedence of upstream sources. After the merge step, edge source relation types (as described in Sec. 2.2) are each mapped to one of 77 predicate types in the Biolink predicate hierarchy (see Sec. 2.2), and redundant edges (same combination of subject node ID, object node ID, and Biolink predicate) are coalesced, with source relation information and source provenance information added to lists in the coalesced edge. The graph is then serialized as JSON (see Sec. 2.4) and to TSV format. The build artifacts, including the unprocessed and processed JSON files and the TSV files, are uploaded into an AWS S3 bucket. RTX-KG2pre is then hosted in Neo4j on a smaller AWS instance (see Sec. 2.8.1); the Neo4j endpoint is mainly used for debugging and data exploration.

2.8.3 umls2rdf and owltools

In the RTX-KG2pre build process, the 26 UMLS sources are ingested as TTL files that are generated in the extraction stage of the build process from the Rich Release Format (RRF [61]) UMLS distribution using two software programs, Metamorphosys [95] (to load the RRF files into the relational database system, MySQL) and umls2rdf [20] (to extract TTL files Sec. 2.8.3). Thus, a local MySQL database is used as an intermediate data source in the build process, from which TTL files are generated via `umls2rdf`. The build system uses the owltools package to convert biomedical ontologies (see Table 1 and Table S1) in OWL format and the UMLS TTL files into OBO (Open Biological and Biomedical Ontology) JSON format for processing. The ontologies in OBO-JSON format are then loaded using the Python package `ontobio` and processed/merged together, enabling use of cross-ontology axioms in determining concept semantic types.

3 Utility and Discussion

3.1 Utility and user interface

Due to its comprehensiveness and/or its speed, RTX-KG2 is already being used as a core knowledge provider (see `github:NCATSTranslator/Translator-All/wiki/KG2`) or knowledge graph by four diverse reasoning agents within the Translator system: ARAX [54] (which our team developed and which provides sophisticated workflow capabilities and overlay of virtual edges for associations based on literature co-occurrence or network structural equivalence); mediKanren (which provides sophisticated network motif-finding and path-finding using the miniKanren logic programming language); Biothings Explorer (the engine for autonomous querying of distributed biomedical knowledge, described in Section 1); and ARAGORN (`github:ranking-agent/aragorn`). Key to the utility of RTX-KG2 in Translator is that RTX-KG2 can be (1) queried via a RESTful, Translator-standard API (Sec. 2.7) and (2) downloaded from the Translator Knowledge Graph Exchange (KGE; see Figure 1) registry in Biolink KGX format [17] (see Sec. 2.7). Both access channels comply with information standards—TRAPI and Biolink in the case of the REST API, and KGX and Biolink in the case of the KGE registry—that ensure interoperability with any other standards-compliant agent operating within the Translator system.

In addition to its primary intended use-case for on-demand knowledge exploration and concept-specific reasoning, the RTX-KG2 knowledge graph can be used as a structure prior for data-driven network inference, for example, causal network learning. We have recently described

Table 5: UMLS sources that are integrated into RTX-KG2. See Sec. 5 for definitions of abbreviations.

UMLS Semantic Network	
Anatomical Therapeutic Chemical Classification System	ATC
DrugBank database	DRUGBANK
Foundational Model of Anatomy	FMA
Gene Ontology	GO
Healthcare Common Procedure Coding System	HCPCS
Human Gene Nomenclature Committee	HGNC
Health Level Seven version 3.0	HL7V3.0
Human Phenotype Ontology	HPO
ICD-10 Procedure Coding System	ICD10PCS
ICD-9, Clinical Modification	ICD9CM
Logical Observation Identif. Names & Codes	LNC
Medication Reference Terminology	MED-RT
MEDLINE Plus	MEDLINEPLUS
Medical Subject Headings (MeSH)	MSH
Metathesaurus	MTH
NCBI Taxon	NCBI
National Cancer Institute Thesaurus	NCI
National Drug Data File	NDDF
National Drug Data File - Reference Terminology	NDFRT
Online Mendelian Inheritance in Man	OMIM
Physician Data Query	PDQ
Psychological Index Terms	PSY
RxNorm (normalized drug names)	RXNORM
National Drug File	VANDF

a computational method, *Kg2Causal* [96], for using a general-purpose biomedical knowledge graph to extract a network structure prior distribution for data-driven causal network inference from multivariate observations. Using the predecessor graph, RTX-KG1, we found that using a general knowledge graph as a prior significantly improved the accuracy of data-driven causal network inference compared to using any of several uninformative network structure priors [96]. To the extent that it incorporates a variety of graph structural variations, RTX-KG2 can also be used as a test-bed for evaluating the performance of structurally generalizable graph analysis methods such as a subset of us have done for the case of a structurally generalizable node-node similarity measure [97].

As stated in Sec. 2.7, our ARAX reasoning system [54] provides a web browser interface (which is publicly available as described in Sec. 6.3) that can be used to both construct queries of RTX-KG2 and browse ranked results from those queries. The browser interface provides various capabilities including node synonymization; graphical layout and exploration of annotated result-graphs; and a graphical, point-and-click query builder. The UI provides persistent uniform resource locator-based results retrieval, which facilitates result-sharing and aggregation.

3.2 Discussion

In designing RTX-KG2, we developed five design principles that guided our selection of knowledge sources to incorporate as well as the architecture of the RTX-KG2 build system:

1. Source is publicly available in a flat-file (e.g., TSV, XML, JSON, DAT, or SQL dump) that can be downloaded via a script
2. Source is being maintained and updated periodically
3. Source provides knowledge triples that complement (i.e., not duplicate) what is already in RTX-KG2
4. Source connects concept identifier types that are already in RTX-KG2
5. Ideally, source provides knowledge based on human curation (favored over computational text-mining)

Principle 1, and the deliberate choice of using an ETL approach, theoretically would allow RTX-KG2 to be reconstructed consistently and independently of the state of external APIs[3]. This is useful for reproducibility, since each knowledge source is stored in its original downloaded form as a build artifact. Using flat files instead of API interfaces also increases the probability that a future build can be completed successfully at any time, since it does not rely on multiple web services to be up for an extended period of time. Additionally, it is in many (though by no means all) cases computationally faster to ETL a file than to dynamically query an API over the Internet. Development of RTX-KG2 is ongoing and our team welcomes recommendations of new sources to include, via issue reports on the RTX-KG2 GitHub project page (see Sec. 6.3).

In selecting the 70 sources for RTX-KG2, we generally adhered to the aforementioned principles but made a few exceptions based on specific trade-offs. For Principle 1, for one source (as described in Sec. 2), we used an API rather than a file download, and for the "via a script" part

[3]Note however, that one API is used in constructing RTX-KG2; see Sec. 2.1.

Table 6: Node and edge counts for various knowledge graphs.

	Nodes	Edges
`HETIONET, v1 [41]`	47,031	2.3 million
`SPOKE ver. 20190707 [43]`	2.15 million	6.16 million
`SRI Reference KG, ver. 2.0`	20.2 million	41.6 million
`ROBOKOP` [46]	6 million	140 million
`RTX-KG2.7.3pre`	10.2 million	54.0 million
`RTX-KG2.7.3c`	6.4 million	39.3 million

of Principle 1, we manually downloaded source dump files for DrugBank, UMLS, and SemMedDB (due to those three sources' comprehensiveness) and RepoDB (due to its information on drug approval status). For Principle 2, an exception was miRbase, due to the lack of a clear alternative source. For Principle 3, partial exceptions were made for the various pathway databases such as Reactome, PathWhiz/SMPDB, and KEGG, which have many overlapping pathways but which also had systems of pathway identifiers that needed to be included in RTX-KG2. Further, each of the pathway databases has different strengths: PathWhiz/SMPDB offer useful links to HMDB and DrugBank; Reactome is popular, trusted, and is well connected with sources like GO and CHEBI; and KEGG CURIEs are popular with users and link to CHEMBL, CHEBI, and GO. The primary exception to Principle 5 is SemMedDB which is based on natural-language processing of biomedical research article abstracts to extract knowledge triples. SemMedDB is particularly useful for downstream reasoning because of its breadth across biomedical literature and because it includes source article references for each triple.

To provide context for the size of RTX-KG2, node and edge counts for various knowledge graphs are shown in Table 6. The large number of edges in the ROBOKOP knowledge graph versus RTX-KG2 reflects the latter's practice of joining edges that have identical subject node, object node, and predicate type. The large number of nodes in the SRI Reference Knowledge Graph [93] versus RTX-KG2pre is largely due to more `InformationContentEntity` nodes (3.7 million vs. 144,396) and `SequenceVariant` nodes (2.4 million vs. 0) in the former. Our observation that RTX-KG2c has a scale-free degree distribution is consistent with previous reports from empirical studies of text-based semantic networks [98] and ontologies [99] and generalizes the scale-free phenomenon into the realm of large-scale knowledge graphs.

Notably, RTX-KG2 contains a richer set of meta-triples (distinct combinations of subject node category, edge predicate, and object node category) versus the other knowledge graphs (Table 7); RTX-KG2c contains 4.6-fold more meta-triples than the second-ranked knowledge graph and 18.5-fold more meta-triples than the third-ranked knowledge graph, by meta-triple count. The finding that canonicalization increases the number of meta-triples can be understood as follows: since each node in RTX-KG2c has multiple categories, the number of meta-triples increases by the product of the count of subject node categories and the count of object node categories, for all subject-object node pairs joined by edges in the graph. The approximately 2.4-fold increase in the number of predicates in ROBOKOP versus in RTX-KG2 partially reflects the design choice in RTX-KG2 to standardize predicate directions and to orient triples so that for any inverted pair of predicates (e.g., "has part" and "part of"), only one of the predicates is used in the graph.

As described in Sec. 1, the Neo4j graph database management system is used by many biomedical

Table 7: Numbers of unique categories, predicates, and meta-triples for various knowledge graphs.

	Categories	Predicates	Meta-triples
`SPOKE, TRAPI v1.2.0 API`	14	24	44
`SRI Reference KG, ver. 2.0`	62	59	2,047
`ROBOKOP, TRAPI v1.2.0 API`	20	185	2,234
`RTX-KG2.7.3pre`	56	77	10,269
`RTX-KG2.7.3c`	56	77	41,225

knowledge graphs including Hetionet, Monarch, SPOKE, EpiGraphDB, and ROBOKOP. Using Neo4j to host RTX-KG2 has both benefits (specifically, the flexibility of the Cypher query language [100]) and drawbacks (in our testing, slow JSON loading performance and slow response times in comparison to an in-memory database). It was due to its drawbacks that we ultimately switched to hosting RTX-KG2c using a dedicated in-memory graph database engine, PloverDB (Sec. 2.3), that we developed. On the other hand, our standard procedure of hosting a Neo4j database server for RTX-KG2pre has been invaluable as a debugging aid and for developing graph queries and analysis workflows.

The performance of the RTX-KG2 build process has also been optimized. Since RTX-KG2 was intended to be a large knowledge graph, the build process was designed to limit the amount of time it would take to build, especially by using the ETL process described above and by utilizing a parallel build system. A full Snakemake parallel build of RTX-KG2pre takes about 47 hours to run, which is $\sim$57% faster than the $\sim$74 hours it would take without parallelization. In addition, Snakemake functionality along with the ETL-based approach allows for unchanged source databases to be ignored until the `Merge` step (see Fig. 2), which reduces build time.

We have found it challenging to balance the importance of manually curated knowledge resources with those that provide numerical data and provenance (such as supporting publications) of their assertions. While these two are not mutually exclusive *per se*, relatively few knowledge sources seem to provide both. Increasingly, reasoning agents in the Translator system will use structured provenance and confidence information/annotations for edges in knowledge graphs such as RTX-KG2; the issue of knowledge sources that are important connectors in translational reasoning but do not provide structured provenance information is an ongoing problem [101].

4 Conclusions

Despite the advances in the field outlined in Sec. 1, no open-source software toolkit was available that could integrate UMLS, SemMedDB, ChEMBL, DrugBank, SMPDB, and other core biomedical knowledge-bases into a single semantically unified knowledge graph. To fill this gap and to provide a comprehensive knowledge-base to serve as as an efficient knowledge-substrate for a biomedical reasoning engine, we constructed RTX-KG2, comprising a set of ETL modules, an integration module, a REST API, and a parallel-capable build system that produces and hosts both pre-canonicalized (RTX-KG2pre) and canonicalized (RTX-KG2c) knowledge graphs for download and for querying. RTX-KG2 is currently extensively used by multiple reasoning agents in the NCATS Biomedical Data Translator project, validating the ETL-focused, monolithic-graph, standards-based design philosophy that guided RTX-KG2's development.

5 List of Abbreviations

- ARAX: Autonomous Relay Agent X
- AWS: Amazon Web Services
- D2J: direct-to-JSON method
- EC2: Elastic Compute Cloud
- ETL: extract–transform–load paradigm
- GO: Gene Ontology
- ICD: International Classification of Diseases
- JSON: JavaScript Object Notation
- KEGG: Kyoto Encyclopedia of Genes and Genomes
- NCATS: National Center for Advancing Translational Sciences
- NCBI: National Center for Biotechnology Information
- OBO: Open Biomedical Ontologies
- OWL: Web Ontology Language
- RBM: RDF-based method
- RDF: Resource Description Framework
- REST: REpresentational State Transfer
- RTX-KG2: Reasoning Tool X, Knowledge Graph Generation Two
- RTX-KG2c: Reasoning Tool X, Knowledge Graph Generation Two, Canonicalized
- RTX-KG2pre: Reasoning Tool X, Knowledge Graph Generation Two, Pre-canonicalization
- S3: Simple Storage Service
- SemMedDB: Semantic Medline Database
- SMPDB: Small Molecule Pathway Database
- SQL: Structured Query Language
- Translator: NCATS Biomedical Data Translator
- TSV: tab-separated value
- TTL: Terse RDF Triple Language
- UMLS: Unified Medical Language System
- XML: eXtensible Markup Language

(See also Table 2.1, Table 5, and Table S1).

6 Declarations

6.1 Ethics approval and consent to participate

Not applicable

6.2 Consent for publication

Not applicable

6.3 Availability of data and materials

The code supporting the conclusions of this article is available in the GitHub repository `github:RTXteam/RTX-KG2`. Downloadable versions of RTX-KG2pre and RTX-KG2c are publicly available at `github:ncats/translator-lfs-artifacts`. The RTX-KG2 API is registered via the SmartAPI framework and can be reached at `arax.rtx.ai/api/rtxkg2/v1.2/openapi.json`. The web browser user interface for querying RTX-KG2 is publicly available at `arax.rtx.ai`.

6.4 Competing Interests

The authors declare that they have no competing interests.

6.5 Funding

Support for this work was provided by NCATS, through the Biomedical Data Translator program (NIH award OT2TR003428). Any opinions expressed in this document are those of the Translator community at large and do not necessarily reflect the views of NCATS, individual Translator team members, or affiliated organizations and institutions. The authors thank Amazon Web Services for in-kind computing infrastructure support.

6.6 Authors' contributions

Wrote the paper: ECW AKG LGK SAR MS ASH; designed the studies: SAR ECW AKG EWD DK; carried out computational work: ECW AKG SAR DK EWD FW LGK LA LM MS CM TSY JCR MS AT YC VF; evaluation, testing, and feedback: AKG ECW DK EWD JCR ASH SAR FW LGK CM LM MS LA TSY AT YC VF.

6.7 Acknowledgements

We thank Yao Yao, Zheng Liu, and Deqing Qu for technical assistance. We thank Chris Mungall, Tom Conlin, Matt Brush, Chunlei Wu, Harold Solbrig, Will Byrd, Michael Patton, Jim Balhoff, Chris Bizon, Deepak Unni, Richard Bruskiewich, Andrew Su, Kevin Xin, and Jeff Henrikson for technical advice and/or feedback. We thank David Wishart and Carin Li for providing a download link for the SMPDB PubMed annotations and NCATS for help with hosting RTX-KG2 on GitHub. AKG gratefully acknowledges support from the ARCS Foundation. TSY gratefully acknowledges support from the Semiconductor Research Corporation.

References

[1] Robert S. Ledley and Lee B. Lusted. Reasoning foundations of medical diagnosis. *Science*, 130(3366):9–21, 1959. URL: `https://science.sciencemag.org/content/130/3366/9`, `arXiv:https://science.sciencemag.org/content/130/3366/9.full.pdf`, `doi:doi:10.1126/science.130.3366.9`.

[2] F B Rogers. Medical subject headings. *Bulletin of the Medical Library Association*, 51(1):114–116, January 1963.

[3] National Library of Medicine (US). Pubmed [internet], 1964. URL: `ttps://www.ncbi.nlm.nih.gov/pubmed/`.

[4] Victor A McKusick. Mendelian Inheritance in Man and its online version, OMIM. *American journal of human genetics*, 80(4):588–604, April 2007.

[5] B L Humphreys, D A Lindberg, H M Schoolman, and G O Barnett. The Unified Medical Language System: an informatics research collaboration. *Journal of the American Medical Informatics Association : JAMIA*, 5(1):1–11, January 1998.

[6] A W Forrey, C J McDonald, G DeMoor, et al. Logical observation identifier names and codes (LOINC) database: a public use set of codes and names for electronic reporting of clinical laboratory test results. *Clin Chem*, 42(1):81–90, January 1996.

[7] Y A Lussier, D J Rothwell, and R A Côté. The SNOMED model: a knowledge source for the controlled terminology of the computerized patient record. *Methods of information in medicine*, 37(2):161–164, June 1998.

[8] E G Brown, L Wood, and S Wood. The medical dictionary for regulatory activities (MedDRA). *Drug safety*, 20(2):109–117, February 1999.

[9] Stuart J Nelson, Kelly Zeng, John Kilbourne, et al. Normalized names for clinical drugs: RxNorm at 6 years. *Journal of the American Medical Informatics Association : JAMIA*, 18(4):441–448, July 2011.

[10] D Brickley and R V Guha. Resource description framework (rdf) schema specification. Technical Report 19990303, World Wide Web Consortium, Cambridge, MA, USA, March 1999. URL: `https://www.w3.org/TR/1999/PR-rdf-schema-19990303/`.

[11] Thomas C Rindflesch and Marcelo Fiszman. The interaction of domain knowledge and linguistic structure in natural language processing: interpreting hypernymic propositions in biomedical text. *Journal of Biomedical Informatics*, 36(6):462–477, December 2003.

[12] Sean Bechhofer, Frank van Harmelen, Jim Hendler, et al. Owl web ontology language reference. Technical Report 20040210, World Wide Web Consortium, Cambridge, MA, USA, February 2004. URL: `https://www.w3.org/TR/2004/REC-owl-ref-20040210/`.

[13] Jonathan Bard, Seung Y Rhee, and Michael Ashburner. An ontology for cell types. *Genome Biology*, 6(2):R21, 2005.

[14] Elena Beisswanger, Stefan Schulz, Holger Stenzhorn, and Udo Hahn. Biotop: An upper domain ontology for the life sciences. *Applied Ontology*, 3(4):205–212, 2008.

[15] Michel Dumontier, Christopher JO Baker, Joachim Baran, et al. The Semanticscience Integrated Ontology (SIO) for biomedical research and knowledge discovery. *Journal of Biomedical Semantics*, 5(1):14, 2014.

[16] Biomedical Data Translator Consortium. Toward a universal biomedical data translator. *Clinical and translational science*, 12(2):86–90, 2019.

[17] Justin Reese, Deepak Unni, Tiffany J Callahan, et al. KG-COVID-19: a framework to produce customized knowledge graphs for COVID-19 response. *bioRxiv*, August 2020.

[18] Richard Bruskiewich, Deepak Unni, Chris Mungall, et al. biolink/biolink-model: 2.0.0, 2021. `doi:10.5281/ZENODO.4895425`.

[19] Rebecca Jackson, Nicolas Matentzoglu, James A Overton, et al. OBO Foundry in 2021: operationalizing open data principles to evaluate ontologies. *Database*, 2021, 10 2021. baab069. `arXiv:https://academic.oup.com/database/article-pdf/doi/10.1093/database/baab069/40854912/baab069.pdf`, `doi:10.1093/database/baab069`.

[20] Mark A Musen, Natalya F Noy, Nigam H Shah, et al. The National Center for Biomedical Ontology. *Journal of the American Medical Informatics Association : JAMIA*, 19(2):190–195, March 2012.

[21] M. Kanehisa. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 28(1):27–30, jan 2000. `doi:10.1093/nar/28.1.27`.

[22] Sunghwan Kim, Jie Chen, Tiejun Cheng, et al. PubChem in 2021: new data content and improved web interfaces. *Nucleic Acids Res*, 49(D1):D1388–D1395, January 2021.

[23] D. S. Wishart. DrugBank: a comprehensive resource for in silico drug discovery and exploration. *Nucleic Acids Research*, 34(90001):D668–D672, jan 2006. `doi:10.1093/nar/gkj067`.

[24] David Mendez, Anna Gaulton, A Patrícia Bento, et al. ChEMBL: towards direct deposition of bioassay data. *Nucleic Acids Research*, 47(D1):D930–D940, nov 2018. URL: `https://doi.org/10.1093/nar/gky1075`, `doi:doi:10.1093/nar/gky1075`.

[25] Alex Bateman, Maria-Jesus Martin, Sandra Orchard, et al. UniProt: the Universal Protein Knowledgebase in 2021. *Nucleic Acids Research*, 49(D1):D480–D489, nov 2020. URL: `https://doi.org/10.1093/nar/gkaa1100`, `doi:doi:10.1093/nar/gkaa1100`.

[26] Alex Frolkis, Craig Knox, Emilia Lim, et al. SMPDB: The Small Molecule Pathway Database. *Nucleic Acids Research*, 38(suppl_1):D480–D487, nov 2009. URL: `https://doi.org/10.1093/nar/gkp1002`, `doi:doi:10.1093/nar/gkp1002`.

[27] Timothy Jewison, Yilu Su, Fatemeh Miri Disfany, et al. SMPDB 2.0: Big improvements to the small molecule pathway database. *Nucleic Acids Research*, 42(D1):D478–D484, nov 2013. URL: `https://doi.org/10.1093/nar/gkt1067`, `doi:doi:10.1093/nar/gkt1067`.

[28] Antonio Fabregat, Florian Korninger, Guilherme Viteri, et al. Reactome graph database: Efficient access to complex pathway data. *PLOS Computational Biology*, 14(1):e1005968, jan 2018. URL: `https://doi.org/10.1371/journal.pcbi.1005968`, `doi:doi:10.1371/journal.pcbi.1005968`.

[29] H. Kilicoglu, D. Shin, M. Fiszman, et al. SemMedDB: a PubMed-scale repository of biomedical semantic predications. *Bioinformatics*, 28(23):3158–3160, oct 2012. URL: `https://doi.org/10.1093/bioinformatics/bts591`, `doi:doi:10.1093/bioinformatics/bts591`.

[30] Aaron Birkland and Golan Yona. BIOZON: a system for unification, management and analysis of heterogeneous biological data. *BMC Bioinformatics*, 7(1):70, 2006.

[31] Antonino Fiannaca, Massimo La Rosa, Laura La Paglia, et al. Biographdb: a new graphdb collecting heterogeneous data for bioinformatics analysis. In *Eighth International Conference on Bioinformatics, Biocomputational Systems and Biotechnologies*, Wilmington, 2016. IARIA.

[32] Vassilis N Ioannidis, Da Zheng, and George Karypis. Few-shot link prediction via graph neural networks for covid-19 drug-repurposing. *arXiv preprint arXiv:2007.10261*, 2020.

[33] Pablo Pareja-Tobes, Raquel Tobes, Marina Manrique, et al. Bio4j: a high-performance cloud-enabled graph-based data platform. *bioRxiv*, 2015. `doi:10.1101/016758`.

[34] Michel Dumontier, Alison Callahan, Jose Cruz-Toledo, et al. Bio2rdf release 3: a larger connected network of linked data for the life sciences. In *Proceedings of the 2014 International Conference on Posters & Demonstrations Track*, volume 1272, pages 401–404. Citeseer, 2014.

[35] Christopher J Mungall, Julie A McMurry, Sebastian Köhler, et al. The Monarch Initiative: an integrative data and analytic platform connecting phenotypes to genotypes across species. *Nucleic acids research*, 45(D1):D712–D722, 2017.

[36] Kevin M Livingston, Michael Bada, William A Baumgartner, and Lawrence E Hunter. KaBOB: ontology-based semantic integration of biomedical databases. *BMC Bioinformatics*, 16(1):126, 2015.

[37] Julie A McMurry, Sebastian Köhler, Nicole L Washington, et al. Navigating the phenotype frontier: The Monarch Initiative. *Genetics*, 203(4):1491–1495, aug 2016. URL: `https://doi.org/10.1534/genetics.116.188870`, `doi:doi:10.1534/genetics.116.188870`.

[38] Kent A Shefchek, Nomi L Harris, Michael Gargano, et al. The Monarch Initiative in 2019: an integrative data and analytic platform connecting phenotypes to genotypes across species. *Nucleic Acids Research*, 48(D1):D704–D715, nov 2019. URL: `https://doi.org/10.1093/nar/gkz997`, `doi:doi:10.1093/nar/gkz997`.

[39] Barry Smith, Werner Ceusters, Bert Klagges, et al. Relations in biomedical ontologies. *Genome Biology*, 6(5):R46, 2005.

[40] Luis Galárraga, Geremy Heitz, Kevin Murphy, and Fabian M Suchanek. Canonicalizing open knowledge bases. In *Proceedings of the 23rd acm international conference on conference on information and knowledge management*, pages 1679–1688, 2014.

[41] Daniel Scott Himmelstein, Antoine Lizee, Christine Hessler, et al. Systematic integration of biomedical knowledge prioritizes drugs for repurposing. *eLife*, 6:e26726, sep 2017. doi:10.7554/eLife.26726.

[42] Sergio Baranzini, Sui Huang, Sharat Israni, et al. Scalable precision medicine knowledge engine, 2021. Accessed: 2021-06-01. URL: https://spoke.ucsf.edu.

[43] Geoffrey Sanders, Roger Pearce, and Sergio E. Baranzini. Topological analysis of the spoke graph. Technical report, U. S. Department of Energy, 9 2020. URL: https://www.osti.gov/biblio/1669224, doi:10.2172/1669224.

[44] Tunca Doğan, Heval Atas, Vishal Joshi, et al. Crossbar: Comprehensive resource of biomedical relations with deep learning applications and knowledge graph representations. *bioRxiv*, 2020. arXiv:https://www.biorxiv.org/content/early/2020/09/15/2020.09.14.296889.full.pdf, doi:10.1101/2020.09.14.296889.

[45] Kenneth Morton, Patrick Wang, Chris Bizon, et al. ROBOKOP: an abstraction layer and user interface for knowledge graphs to support question answering. *Bioinformatics*, 35(24):5382–5384, August 2019.

[46] Karamarie Fecho, Chris Bizon, Frederick Miller, et al. A biomedical knowledge graph system to propose mechanistic hypotheses for real-world environmental health observations: Cohort study and informatics application. *JMIR Med Inform*, 9(7):e26714, Jul 2021. doi:10.2196/26714.

[47] Yi Liu, Benjamin Elsworth, Pau Erola, et al. EpiGraphDB: a database and data mining platform for health data science. *Bioinformatics*, November 2020.

[48] Amrapali Zaveri, Shima Dastgheib, Chunlei Wu, et al. smartapi: Towards a more intelligent network of web apis. In Eva Blomqvist, Diana Maynard, Aldo Gangemi, et al., editors, *The Semantic Web*, pages 154–169, Cham, 2017. Springer International Publishing.

[49] Stephen Ramsey, David Koslicki, Yao Yao, et al. RTXteam/RTX: Initial proof-of-concept software version from november 2017, 2018. doi:10.5281/ZENODO.1185486.

[50] Jiwen Xin, Cyrus Afrasiabi, Sebastien Lelong, et al. Cross-linking BioThings APIs through JSON-LD to facilitate knowledge exploration. *BMC Bioinformatics*, 19(1):30, 2018.

[51] Christopher J. Mungall, Julie A. McMurry, Sebastian Köhler, et al. The Monarch Initiative: an integrative data and analytic platform connecting phenotypes to genotypes across species. *Nucleic Acids Research*, 45(D1):D712–D722, nov 2016. URL: https://doi.org/10.1093/nar/gkw1128, doi:doi:10.1093/nar/gkw1128.

[52] Ben Elsworth. Epigraphdb, 2021. doi:10.5281/ZENODO.4534128.

[53] Johannes Köster and Sven Rahmann. Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics*, 28(19):2520–2522, August 2012.

[54] RTX Team. Arax: a modular graph-based reasoning tool to explore biomedical questions, 2021. URL: https://github.com/RTXteam/RTX.

[55] Richard D Hipp. SQLite, 2020. URL: https://www.sqlite.org/index.html.

[56] World Wide Web Consortium et al. Rdf 1.1 turtle: terse rdf triple language. Technical Report 20140225, World Wide Web Consortium, Cambridge, MA, USA, jan 2014. URL: https://www.w3.org/TR/turtle/.

[57] Fabien Gandon, Guus Schreiber, and Dave Beckett. Rdf 1.1 xml syntax. Technical Report 20140225, World Wide Web Consortium, Cambridge, MA, USA, February 2014. URL: http://www.w3.org/TR/2014/REC-rdf-syntax-grammar-20140225/.

[58] James Malone, Ele Holloway, Tomasz Adamusiak, et al. Modeling sample variables with an Experimental Factor Ontology. *Bioinformatics*, 26(8):1112–1118, March 2010.

[59] Drashtti Vasant, Laetitia Chanas, James Malone, et al. Ordo: an ontology connecting rare disease, epidemiology and genetic data. In *Proceedings of ISMB*, volume 30, 2014.

[60] Fatima Zohra Smaili, Xin Gao, and Robert Hoehndorf. Formal axioms in biomedical ontologies improve analysis and interpretation of associated data. *Bioinformatics*, 36(7):2229–2236, 12 2019. arXiv:https://academic.oup.com/bioinformatics/article-pdf/36/7/2229/33027575/btz920.pdf, doi:10.1093/bioinformatics/btz920.

[61] UMLS Team. *UMLS Reference Manual*, chapter 3. National Library of Medicine (US), Bethesda, 2009. URL: https://www.ncbi.nlm.nih.gov/books/NBK9685.

[62] Mark Davies, Michał Nowotka, George Papadatos, et al. ChEMBL web services: streamlining access to drug discovery data and utilities. *Nucleic Acids Research*, 43(W1):W612–W620, apr 2015. URL: https://doi.org/10.1093/nar/gkv352, doi:doi:10.1093/nar/gkv352.

[63] Sharon L Freshour, Susanna Kiwala, Kelsy C Cotto, et al. Integration of the Drug–Gene Interaction Database (DGIdb 4.0) with open crowdsource efforts. *Nucleic Acids Research*, 49(D1):D1144–D1151, nov 2020. URL: https://doi.org/10.1093/nar/gkaa1084, doi:doi:10.1093/nar/gkaa1084.

[64] Janet Piñero, Juan Manuel Ramírez-Anguita, Josep Saüch-Pitarch, et al. The DisGeNET knowledge platform for disease genomics: 2019 update. *Nucleic Acids Research*, nov 2019. URL: https://doi.org/10.1093/nar/gkz1021, doi:doi:10.1093/nar/gkz1021.

[65] Sorin Avram, Cristian G Bologa, Jayme Holmes, et al. DrugCentral 2021 supports drug discovery and repositioning. *Nucleic Acids Research*, 49(D1):D1160–D1169, nov 2020. URL: https://doi.org/10.1093/nar/gkaa997, doi:doi:10.1093/nar/gkaa997.

[66] Andrew D Yates, Premanand Achuthan, Wasiu Akanni, et al. Ensembl 2020. *Nucleic Acids Research*, nov 2019. URL: https://doi.org/10.1093/nar/gkz966, doi:doi:10.1093/nar/gkz966.

[67] Seth Carbon, Eric Douglass, Benjamin M Good, et al. The Gene Ontology resource: enriching a GOld mine. *Nucleic Acids Research*, 49(D1):D325–D334, dec 2020. URL: https://doi.org/10.1093/nar/gkaa1113, doi:doi:10.1093/nar/gkaa1113.

[68] Michael Ashburner, Catherine A. Ball, Judith A. Blake, et al. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, may 2000. URL: https://doi.org/10.1038/75556, doi:doi:10.1038/75556.

[69] D. S. Wishart, D. Tzur, C. Knox, et al. HMDB: the human metabolome database. *Nucleic Acids Research*, 35(Database):D521–D526, jan 2007. doi:10.1093/nar/gkl923.

[70] D. S. Wishart, C. Knox, A. C. Guo, et al. HMDB: a knowledgebase for the human metabolome. *Nucleic Acids Research*, 37(Database):D603–D610, jan 2009. doi:10.1093/nar/gkn810.

[71] David S. Wishart, Timothy Jewison, An Chi Guo, et al. HMDB 3.0—the human metabolome database in 2013. *Nucleic Acids Research*, 41(D1):D801–D807, nov 2012. doi:10.1093/nar/gks1065.

[72] David S Wishart, Yannick Djoumbou Feunang, Ana Marcu, et al. HMDB 4.0: the human metabolome database for 2018. *Nucleic Acids Research*, 46(D1):D608–D617, nov 2017. doi:10.1093/nar/gkx1089.

[73] H. Hermjakob. IntAct: an open source molecular interaction database. *Nucleic Acids Research*, 32(90001):452D–455, jan 2004. doi:10.1093/nar/gkh052.

[74] S. Kerrien, B. Aranda, L. Breuza, et al. The IntAct molecular interaction database in 2012. *Nucleic Acids Research*, 40(D1):D841–D846, nov 2011. doi:10.1093/nar/gkr1088.

[75] Sune Pletscher-Frankild, Albert Pallejà, Kalliopi Tsafou, et al. DISEASES: Text mining and data integration of disease–gene associations. *Methods*, 74:83–89, mar 2015. doi:10.1016/j.ymeth.2014.11.020.

[76] Minoru Kanehisa. Toward understanding the origin and evolution of cellular organisms. *Protein Science*, 28(11):1947–1951, sep 2019. doi:10.1002/pro.3715.

[77] Minoru Kanehisa, Miho Furumichi, Yoko Sato, et al. KEGG: integrating viruses and cellular organisms. *Nucleic Acids Research*, 49(D1):D545–D551, oct 2020. doi:10.1093/nar/gkaa970.

[78] S. Griffiths-Jones. The microRNA registry. *Nucleic Acids Research*, 32(90001):109D–111, jan 2004. URL: https://doi.org/10.1093/nar/gkh023, doi:doi:10.1093/nar/gkh023.

[79] S. Griffiths-Jones. miRBase: microRNA sequences, targets and gene nomenclature. *Nucleic Acids Research*, 34(90001):D140–D144, jan 2006. URL: https://doi.org/10.1093/nar/gkj112, doi:doi:10.1093/nar/gkj112.

[80] S. Griffiths-Jones, H. K. Saini, S. van Dongen, and A. J. Enright. miRBase: tools for microRNA genomics. *Nucleic Acids Research*, 36(Database):D154–D158, dec 2007. URL: https://doi.org/10.1093/nar/gkm952, doi:doi:10.1093/nar/gkm952.

[81] A. Kozomara and S. Griffiths-Jones. miRBase: integrating microRNA annotation and deep-sequencing data. *Nucleic Acids Research*, 39(Database):D152–D157, oct 2010. URL: https://doi.org/10.1093/nar/gkq1027, doi:doi:10.1093/nar/gkq1027.

[82] Ana Kozomara, Maria Birgaoanu, and Sam Griffiths-Jones. miRBase: from microRNA sequences to function. *Nucleic Acids Research*, 47(D1):D155–D162, nov 2018. URL: https://doi.org/10.1093/nar/gky1141, doi:doi:10.1093/nar/gky1141.

[83] NCBI Resource Coordinators. Database resources of the National Center for Biotechnology Information. *Nucleic Acids Research*, 44(D1):D7–D19, nov 2015. URL: https://doi.org/10.1093/nar/gkv1290, doi:doi:10.1093/nar/gkv1290.

[84] S. S. Weinreich, R. Magnon, J. J. Sikkens, et al. Orphanet: een europese database over zeldzame ziekten [orphanet: a european database for rare diseases]. *Nederlands tijdschrift voor geneeskunde*, 152(9):518–519, mar 2008. URL: https://pubmed.ncbi.nlm.nih.gov/18389888/.

[85] Allison Pon, Timothy Jewison, Yilu Su, et al. Pathways with PathWhiz. *Nucleic Acids Research*, 43(W1):W552–W559, may 2015. URL: https://doi.org/10.1093/nar/gkv399, doi:doi:10.1093/nar/gkv399.

[86] Miguel Ramirez-Gaona, Ana Marcu, Allison Pon, et al. A web tool for generating high quality machine-readable biological pathways. *Journal of Visualized Experiments*, 120, feb 2017. URL: https://doi.org/10.3791/54869, doi:doi:10.3791/54869.

[87] David S Wishart, Carin Li, Ana Marcu, et al. PathBank: a comprehensive pathway database for model organisms. *Nucleic Acids Research*, 48(D1):D470–D478, oct 2019. URL: https://doi.org/10.1093/nar/gkz861, doi:doi:10.1093/nar/gkz861.

[88] Bijay Jassal, Lisa Matthews, Guilherme Viteri, et al. The Reactome pathway knowledgebase. *Nucleic Acids Research*, nov 2019. URL: https://doi.org/10.1093/nar/gkz1031, doi:doi:10.1093/nar/gkz1031.

[89] O. Bodenreider. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32(90001):267D–270, jan 2004. URL: https://doi.org/10.1093/nar/gkh061, doi:doi:10.1093/nar/gkh061.

[90] Jon Chambers, Mark Davies, Anna Gaulton, et al. UniChem: a unified chemical structure cross-referencing and identifier tracking system. *Journal of Cheminformatics*, 5(1), jan 2013. URL: https://doi.org/10.1186/1758-2946-5-3, doi:doi:10.1186/1758-2946-5-3.

[91] Mark Birbeck and Shane McCarron. Curie syntax 1.0: a syntax for expressing compact uris. Technical Report 20101216, World Wide Web Consortium, Cambridge, MA, USA, December 2010. URL: https://www.w3.org/TR/2010/NOTE-curie-20101216/.

[92] Conrad L Schoch, Stacy Ciufo, Mikhail Domrachev, et al. NCBI taxonomy: a comprehensive update on curation, resources and tools. *Database*, 2020, jan 2020. URL: https://doi.org/10.1093/database/baaa062, doi:doi:10.1093/database/baaa062.

[93] Deepak Unni and Kent Shefchek. Sri reference kg, 2022. URL: https://github.com/Knowledge-Graph-Hub/sri-reference-kg.

[94] Roy Thomas Fielding. *REST: Architectural Styles and the Design of Network-based Software Architectures*. Doctoral dissertation, University of California, Irvine, 2000. URL: http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm.

[95] Olivier Bodenreider. The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Res*, 32(Database issue):D267–70, January 2004.

[96] Meghamala Sinha and Stephen A Ramsey. Using a general prior knowledge graph to improve data-driven causal network learning. In *AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering*, 2021.

[97] Yodsawalai Chodpathumwan, Arash Termehchy, Stephen A. Ramsey, et al. Structural generalizability: The case of similarity search. In *Proceedings of the 2021 International Conference on Management of Data*, SIGMOD/PODS '21, page 326–338, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3448016.3457316.

[98] Mark Steyvers and Joshua B Tenenbaum. The large-scale structure of semantic networks: statistical analyses and a model of semantic growth. *Cognitive Science*, 29(1):41–78, January 2005.

[99] Yuehang Ding, Hongtao Yu, Ruiyang Huang, and Yunjie Gu. Complex network based knowledge graph ontology structure analysis. In *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*. IEEE, aug 2018. doi:10.1109/hoticn.2018.8606002.

[100] Nadime Francis, Alastair Green, Paolo Guagliardo, et al. Cypher: An evolving query language for property graphs. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1433–1445, 2018.

[101] Leslie F Sikos and Dean Philp. Provenance-Aware Knowledge Representation: A Survey of Data Models and Contextualized Knowledge Graphs. *Data Science and Engineering*, 5(3):293–316, 2020.

[102] Janna Hastings, Gareth Owen, Adriano Dekker, et al. ChEBI in 2016: Improved services and an expanding collection of metabolites. *Nucleic Acids Research*, 44(D1):D1214–D1219, oct 2015. URL: https://doi.org/10.1093/nar/gkv1031, doi:doi:10.1093/nar/gkv1031.

[103] Petra Fey, Robert J. Dodson, Siddhartha Basu, and Rex L. Chisholm. One stop shop for everything dictyostelium: dictyBase and the Dicty Stock Center in 2012. In *Methods in Molecular Biology*, pages 59–92. Humana Press, 2013. URL: https://doi.org/10.10072F978-1-62703-302-2_4, doi:doi:10.1007/978-1-62703-302-2_4.

[104] Siddhartha Basu, Petra Fey, Yogesh Pandit, et al. dictyBase 2013: integrating multiple dictyostelid species. *Nucleic Acids Research*, 41(D1):D676–D683, nov 2012. URL: https://doi.org/10.1093/nar/gks1064, doi:doi:10.1093/nar/gks1064.

[105] Petra Fey, Pascale Gaudet, Tomaz Curk, et al. dictyBase—a dictyostelium bioinformatics resource update. *Nucleic Acids Research*, 37(suppl_1):D515–D519, oct 2008. URL: https://doi.org/10.1093/nar/gkn844, doi:doi:10.1093/nar/gkn844.

[106] R. L. Chisholm. dictyBase, the model organism database for dictyostelium discoideum. *Nucleic Acids Research*, 34(90001):D423–D427, jan 2006. URL: https://doi.org/10.1093/nar/gkj090, doi:doi:10.1093/nar/gkj090.

[107] L. Kreppel. dictyBase: a new dictyostelium discoideum genome database. *Nucleic Acids Research*, 32(90001):332D–333, jan 2004. URL: https://doi.org/10.1093/nar/gkh138, doi:doi:10.1093/nar/gkh138.

[108] Chris Mungall, Shawn Tan, Nicole Vasilevsky, et al. obophenotype/cell-ontology: 2021-04-22 release, 2021. URL: https://zenodo.org/record/592969, doi:doi:10.5281/ZENODO.592969.

[109] Jonathan Bard. A new ontology (structured hierarchy) of human developmental anatomy for the first 7 weeks (carnegie stages 1-20). *Journal of Anatomy*, 221(5):406–416, sep 2012. URL: https://doi.org/10.1111/j.1469-7580.2012.01566.x, doi:doi:10.1111/j.1469-7580.2012.01566.x.

[110] Chuming Chen, Hongzhan Huang, Karen E. Ross, et al. Protein ontology on the semantic web for knowledge discovery. *Scientific Data*, 7(1), oct 2020. URL: https://doi.org/10.1038/s41597-020-00679-9, doi:doi:10.1038/s41597-020-00679-9.

7 Supplementary Material

Table S1: Ontologies from the OBO Foundry that are included in RTX-KG2.

Basic Formal Ontology	BFO
Chemical Entities of Biological Interest (ChEBI)	CHEBI [102]
Gene Ontology, with external relationships	`go-plus`
Relation Ontology	RO
Uberon multi-species anatomy ontology, extended with external relationships	UBERON
Foundational Model of Anatomy	FMA
Dictyostelium discoideum anatomy	DDANAT [103–107]
Cell Ontology	CL [108]
Food Ontology	FOODON
Human Developmental Anatomy, abstract	EHDAA2 [109]
Biological Spatial Ontology	BSPO
Human Phenotype Ontology	HPO
Neuro Behavior Ontology	NBO
NCBI organismal classification, taxslim subset	ncbitaxon [92]
Phenotype and Trait Ontology	PATO
Mondo Disease Ontology	MONDO
Disease Ontology	DO
Protein Ontology	PRO [110]
Interaction Network Ontology	INO
Genomic Epidemiology Ontology	GENEPIO
Molecular Interactions Controlled Vocabulary	MI