GRAKN.AI

Accelerating Biomedical Knowledge Discovery with Grakn

Systems biology produces a tremendous amount of heterogeneous data which present challenges in their integration due to their complex nature and rich semantics. As understanding the complex relationships among these biological data is one of the key goals in biology, solutions that speed up the integration and querying of such data are necessary.

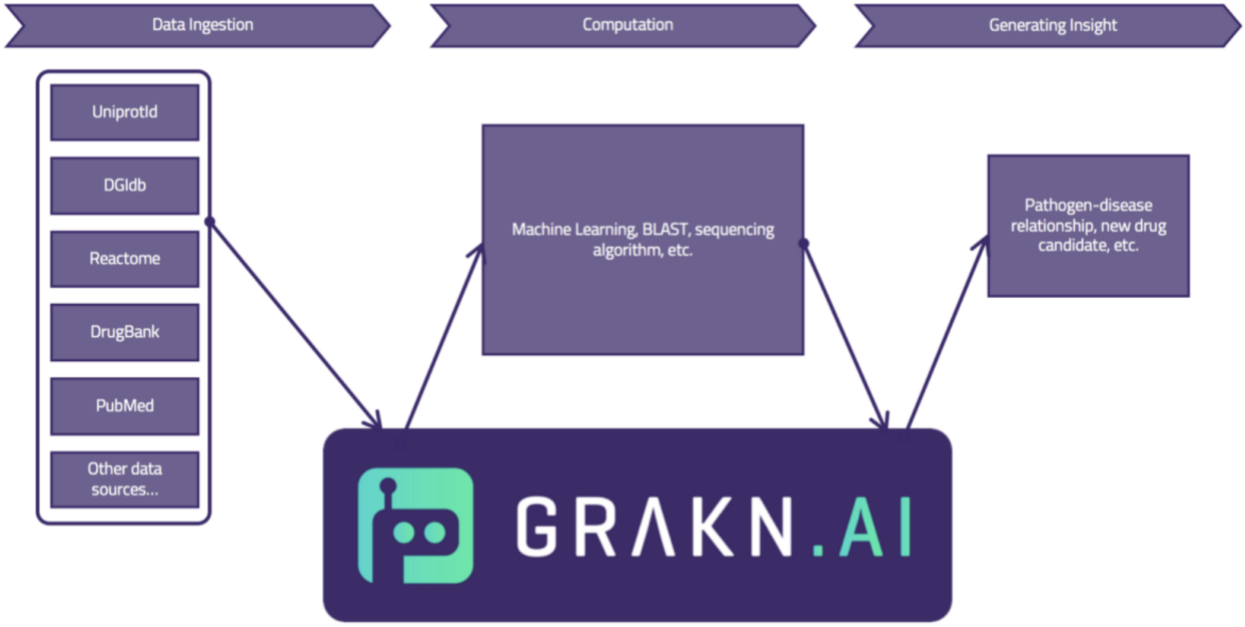
However, analysing large volumes of this biological data through traditional database systems is troublesome and challenging. This white paper demonstrates how using a Grakn knowledge graph for complex biological relationships would accelerate the knowledge discovery process. This is done by integrating nine heterogeneous biomedical datasets.

Integrating and Ingesting Biomedical Data In Traditional Systems

The rapid development and spread of analytical tools in biomedical science have produced a variety of information about all sorts of biological components (cells, tissues, diseases, proteins, pathways, drugs, etc) and their functions. Though important individually, their biological characteristics need to be understood in relation to the interactions they have with different biological components, which requires the integration of vast amounts of heterogenous data. However, as this data is highly complex and semantically rich, doing so becomes very hard.

For example, some genes may be connected to multiple diseases, encoding many sequence-similar proteins, and could include some provenance information. Identifying the relationships between such entities can be crucial in providing the biological context for hypothesis generation and validation.

A generalised workflow can be illustrated as follow.



As is illustrated above, typically the first step before producing any type of insight (e.g. drug discovery) involves the integration of disparate biomedical data. These are often public or proprietary datasets, while others extract data through text mining methods from, for example, PubMed articles. The data sets often come in TSV/CSVs formats (e.g. Uniprot, ENSEMBL, Drug Bank, etc) and as this is flat data, it is necessary to connect these datasets so that the networks can be established. In doing so, it is not uncommon to end up with a few thousand lines of code.

This approach suffers from the fact that biological data is not always uniform. For example, some proteins may have been investigated experimentally, are well annotated and connected with other pieces of information. However, other proteins may just be 'hypothetical proteins' and have little to no information. Further, as biological research is unpredictable, new studies are constantly published which we may want to integrate into our workflow. The result is a long and arduous process that is very time consuming and is difficult to scale.

Once a certain set of data has been integrated (either in-memory or in a traditional relational database), this can then be ingested into a computational layer to produce some form of insight. For example, this could be through the use of a sequencing algorithm to find sequence similarities between proteins or genes, or using machine learning to predict certain patterns.

However, in the workflow described, this insight will not enjoy the biological contextuality insofar as it extends to the previously ingested data unless we go through a complex integration process. For example, this would mean that although the new insight may suggest a new drug candidate for a disease, we are unable to understand how it interacts with other biological components (tissues, pathways, proteins, etc). And even though this data can be represented in traditional relational

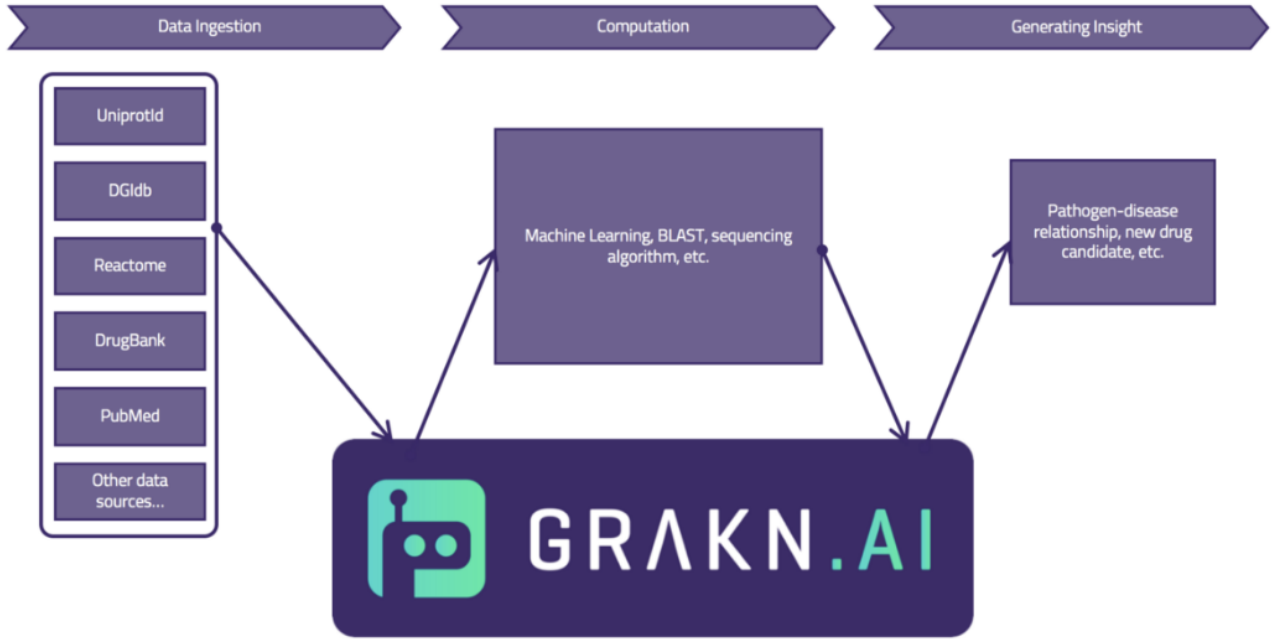
databases, join queries (which connect separate tables) make this computationally too expensive and complicated to design, especially as more complex join statements are done.

Having gone through this high level overview, we can summarise two areas that could be improved using Grakn (which will be touched upon in the next section):

1. **Data integration and ingestion**: Using a hard coded script to integrate flat data, whether in memory or into a relational database, is time consuming and inflexible, especially when it comes to adding new data sources. And because of the data's inherent complexity, navigating it can be computationally too expensive.
2. **Bringing biological context to newly generated insight**: Due to the inflexible nature of the data integration process, it becomes difficult to associate the data produced from a sequencing or machine learning algorithm. Therefore, this new data will lack the biological contextuality in its interaction with other biological components.

Using a Grakn Knowledge Graph to Represent and Query Biomedical Data

The problems above are those that Grakn addresses in order to accelerate the knowledge discovery process. But before delving into the specifics, it will be illustrated where Grakn sits in the knowledge discovery process from a high level point of view:



What is shown here is that Grakn serves as the database that represents every disparate data source throughout the entire knowledge discovery process. This will then allow to easily add new and

unexpected data sources, integrate with generated insights, use complex traversal type queries, and change the model if necessary.

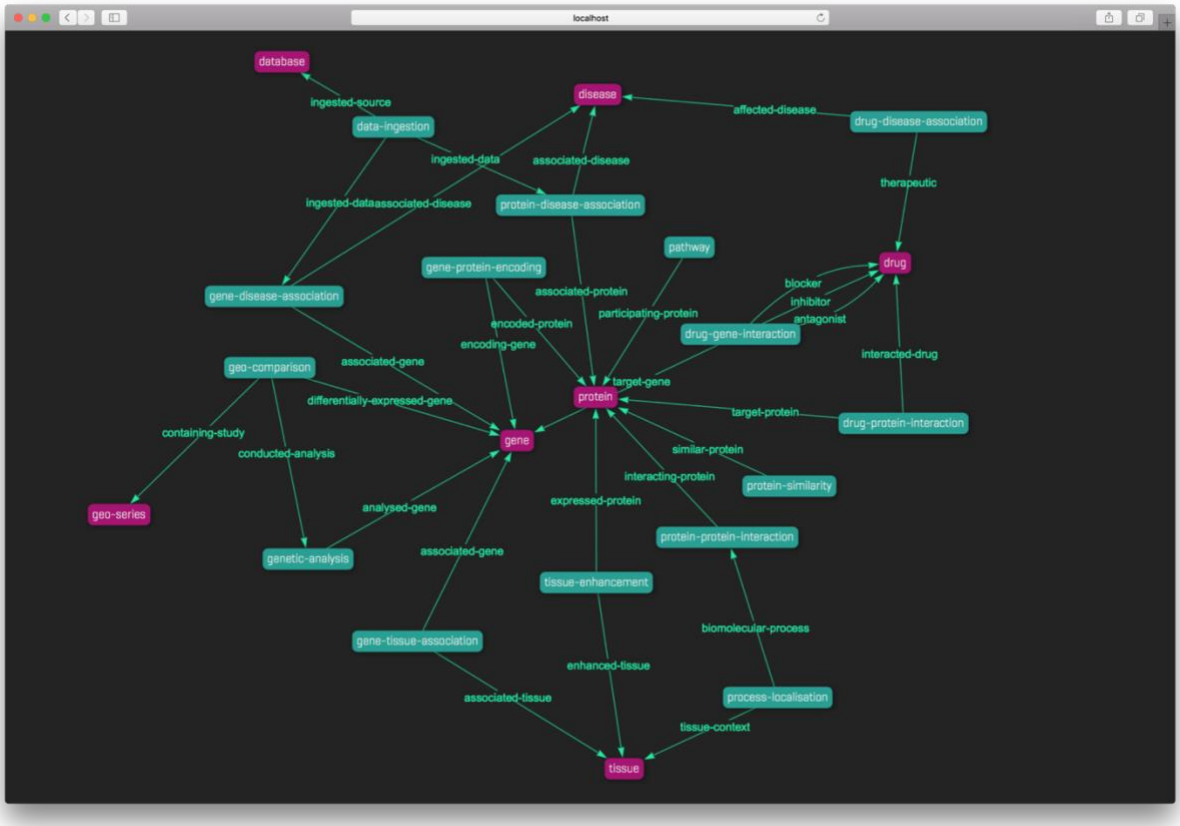
Grakn is an intelligent database in the form of a knowledge graph to organise complex networks of data. The concepts of the systems are entities and relationships, while rules can be added to perform automated reasoning in a distributed environment. Grakn's schema is a type system that implements the principles of knowledge representation and reasoning, which produce a more expressive and useful system than traditional relational and NoSQL databases when managing large-scale linked data.

Ingesting and Integrating Data into Grakn — BioGrakn

In order to demonstrate the usefulness of Grakn, the following data sources have been integrated into a Grakn Knowledge Graph (which is available as BioGrakn Disease Networks):

- **UniProt**: Included are the human subset of protein annotations.
- **Reactome**: Included is the human metabolic pathways and their links with proteins.
- **DGIdb**: The Drug Gene Interaction Database for approved drug compounds and their links with proteins using the UniProt id was used.
- **DisGeNET**: Included were the curated subset for diseases and mapped to UniProt identifiers using gene names and the UniProt API
- **HPA-Tissue**: The data on gene-expression-tissue enhanced associations were acquired, and mapped ENSEMBL ids to UniProt ids using the UniProt API
- **EBI IntAct**: The protein-protein interaction data was included, using UniProt ids as reference points
- **Gene Expression Omnibus**: Three studies (GSE27876, GSE43696, GSE63142) were integrated and differentially expressed genes (DEGs) were identified between asthma subtype/control cohorts using the *limma* Bioconductor package. The ENSEMBL identifiers were mapped to the UniProt id using the UniProt API
- **TissueNet**: This data set contains associations between human tissues and protein-protein interactions, using UniProt ids as reference points

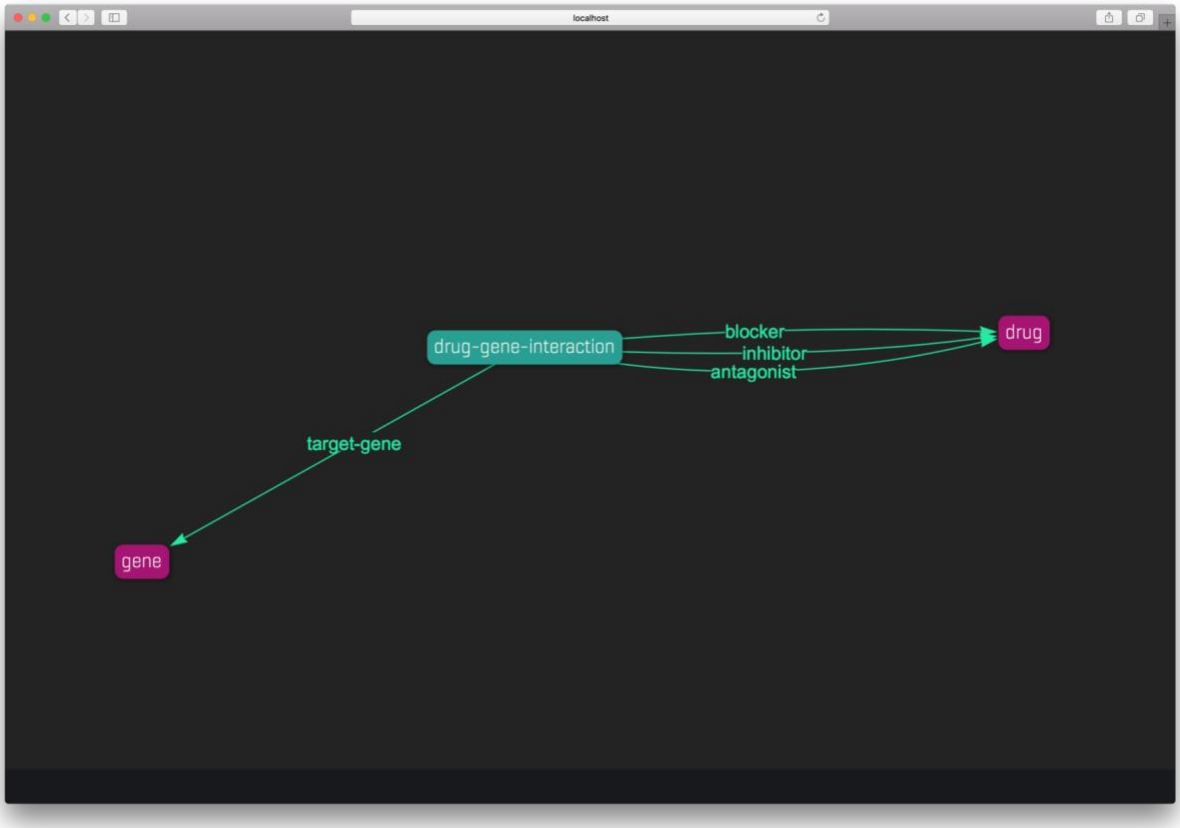
In creating BioGrakn a simple schema was written to represent the above data sets, which is visualised below. Then, the Grakn Python client was used client to load this data from their TSVs and CSVs into BioGrakn.



Purple nodes are entities, green nodes are relationships.

Having created BioGrakn, there were a number of reasons that made this process so easy to do with Grakn. These included:

1. Grakn's flexible and expressive language, Graql, allowed to **modify the schema** as new datasets were ingested. This meant, for example, that when drug-gene relationships were inserted from DGIdb, we could change the schema and create three different roles for drugs when interacting with a gene: inhibitor, antagonist, and blocker (see below). This prevented from having to create three new relationship types for each role. In a traditional relational database, implementing such a change would have meant a re-design of the schema, which can be a costly and complicated process.



2. Grakn's type system also allowed to **hierarchically model** a number of relationship types and attributes, to enable easier querying afterwards. This is best illustrated with the attributes identifier and name, which were created as the parent type of all other identifiers and names. This is how it looks like:

name sub attribute datatype string;
disease-name sub name;
database-name sub name;
uniprot-name sub name;
gene-symbol sub name;
gene-name sub name;
protein-name sub name;
pathway-name sub name;
tissue-name sub name;
drug-name sub name;

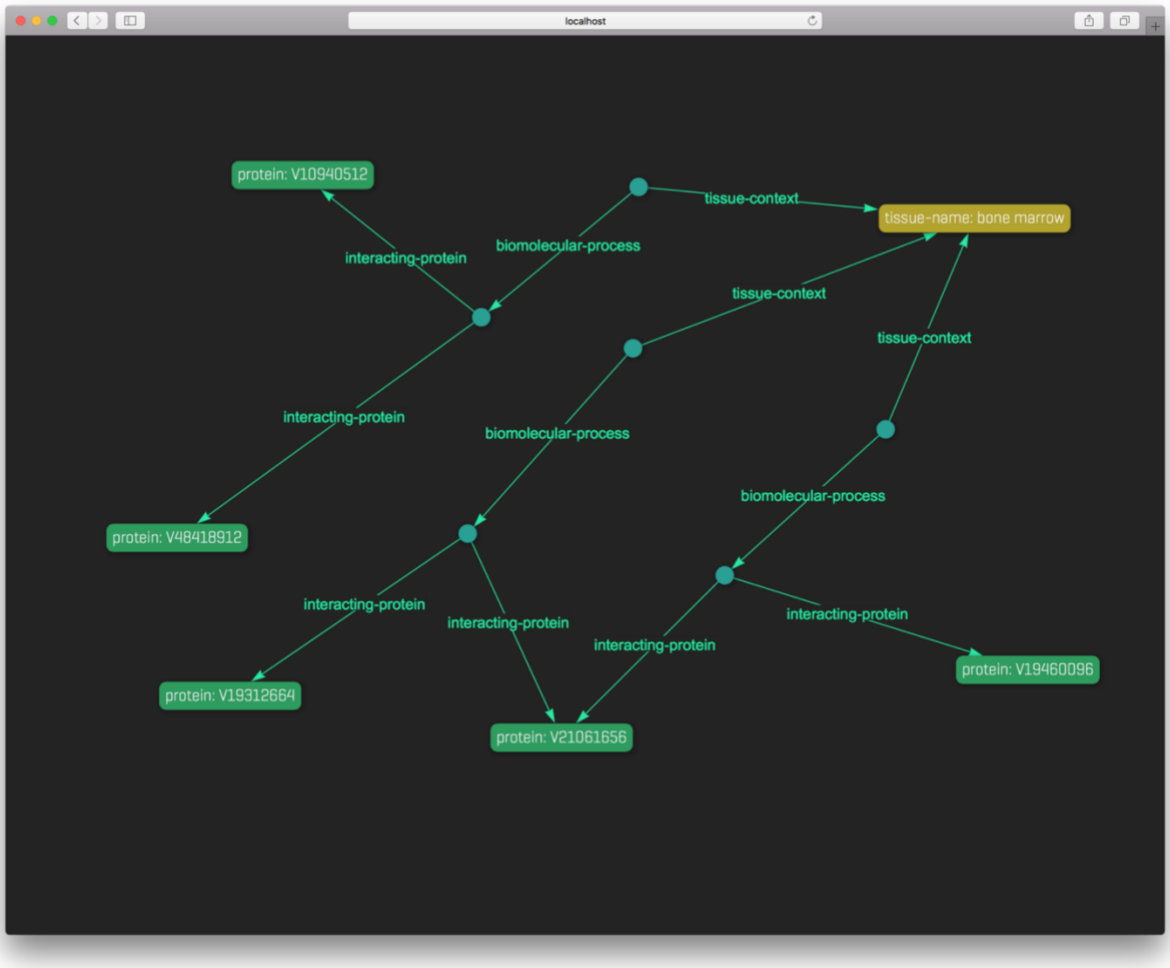
identifier sub attribute datatype string;
disease-id sub identifier;
protein-id sub identifier;
uniprot-id sub protein-id;

gene-id sub identifier;
entrez-id sub gene-id;
ensembl-id sub gene-id;
GEOStudy-id sub identifier;
GEOComparison-id sub identifier;
pathway-id sub identifier;
drug-chembl-id sub identifier;

Modelling like this means that any type of identifier can be queried for without having to explicitly state if, for example, an entrez-id or ensembl-id exist. This is how we would then query for entrez-id 29851 and ask to be returned its gene symbol:

```
>>> match $g isa gene, has identifier "29851", has gene-symbol $gs; get $gs;
>> {$gs val "ICOS" isa gene-symbol;}
```

3. Further, when we wanted to insert tissue-PPI relationships from TissueNet, it was decided to use **hyper-relationships** to express this concept. The tissue entity was modelled as having a process-localisation relationship with the protein-protein-interaction relationship — a relationship inside another relationship (see below). This level of expressivity is useful as it means the model can be designed more closely guided by the needs of its applicatio



These are three relationships between the tissue '"bone marrow" and PPIs. PPIs are modelled as relationships, connected to another relationship that connects to the tissue bone marrow (yellow node). Proteins are modelled as entities (green nodes)

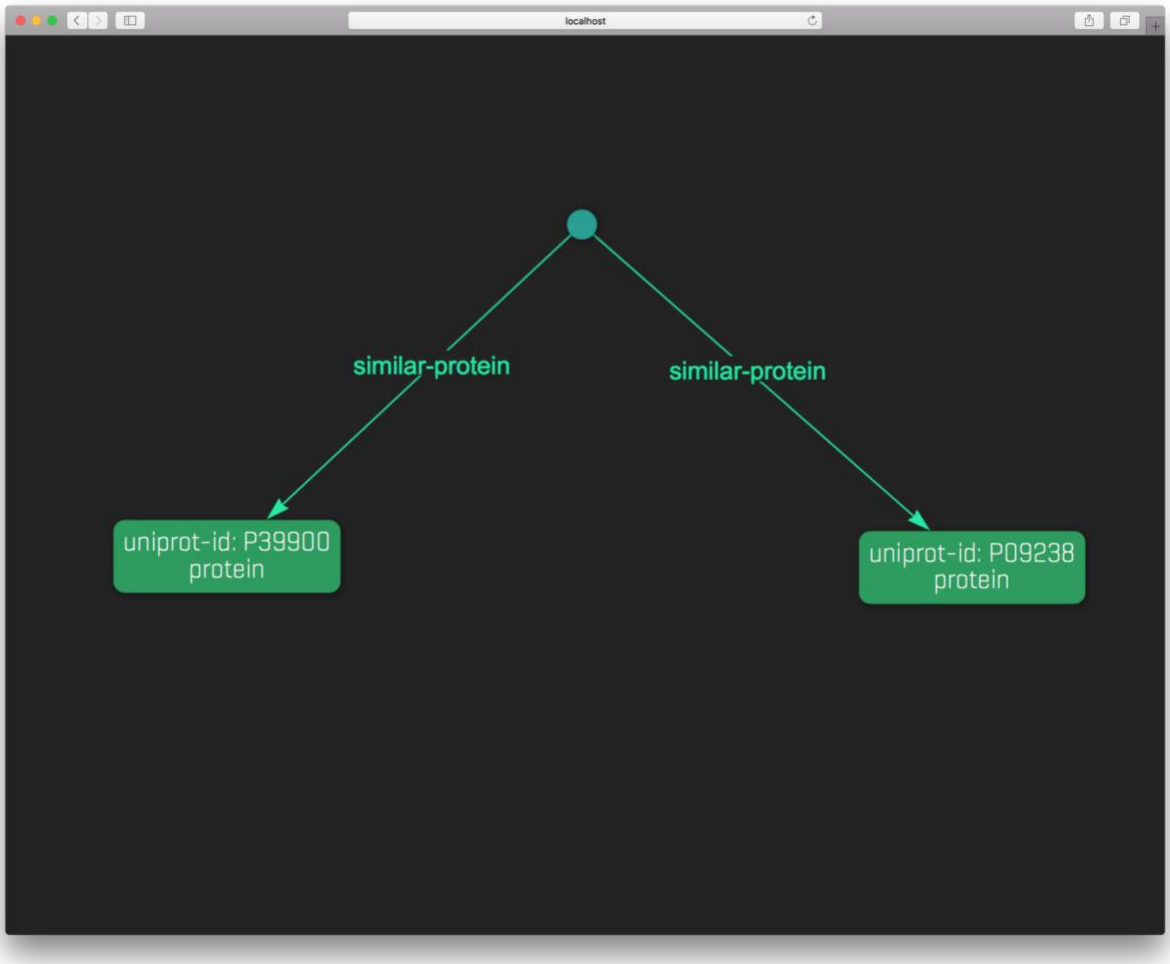
Having integrated this data in BioGrakn, it can be leveraged to do some form of computation — a sequencing or machine learning algorithm, for example. In the next section we will touch on how BioGrakn can be used to provide the biological context to the insights generated in these computations.

Bringing Biological Context to Newly Generated Insight

When running a sequencing or ML algorithm, a new type of insight is created. On its own this can be extremely valuable. But in order to go further in the knowledge discovery process, we should provide the biological context to that insight, by for example integrating it into BioGrakn. To outline the benefits for Grakn for the contextualisation of these insights is as follows:

1. If a **sequencing algorithm** was used that found similarities between sequences of proteins, these can be inserted as sequence similarity relationships between two protein entities. As an example: below you

can see how we would model a sequence similarity between proteins with uniport-id P09238 and P39900.



2. Then, **Grakn rules** can be defined in the schema to find new insights in the data. Below is an example rule. This particular one will create a new drug-disease-association relationship when Grakn finds two proteins with a sequence similarity, where one protein is a target for a disease and the other relates to a drug.

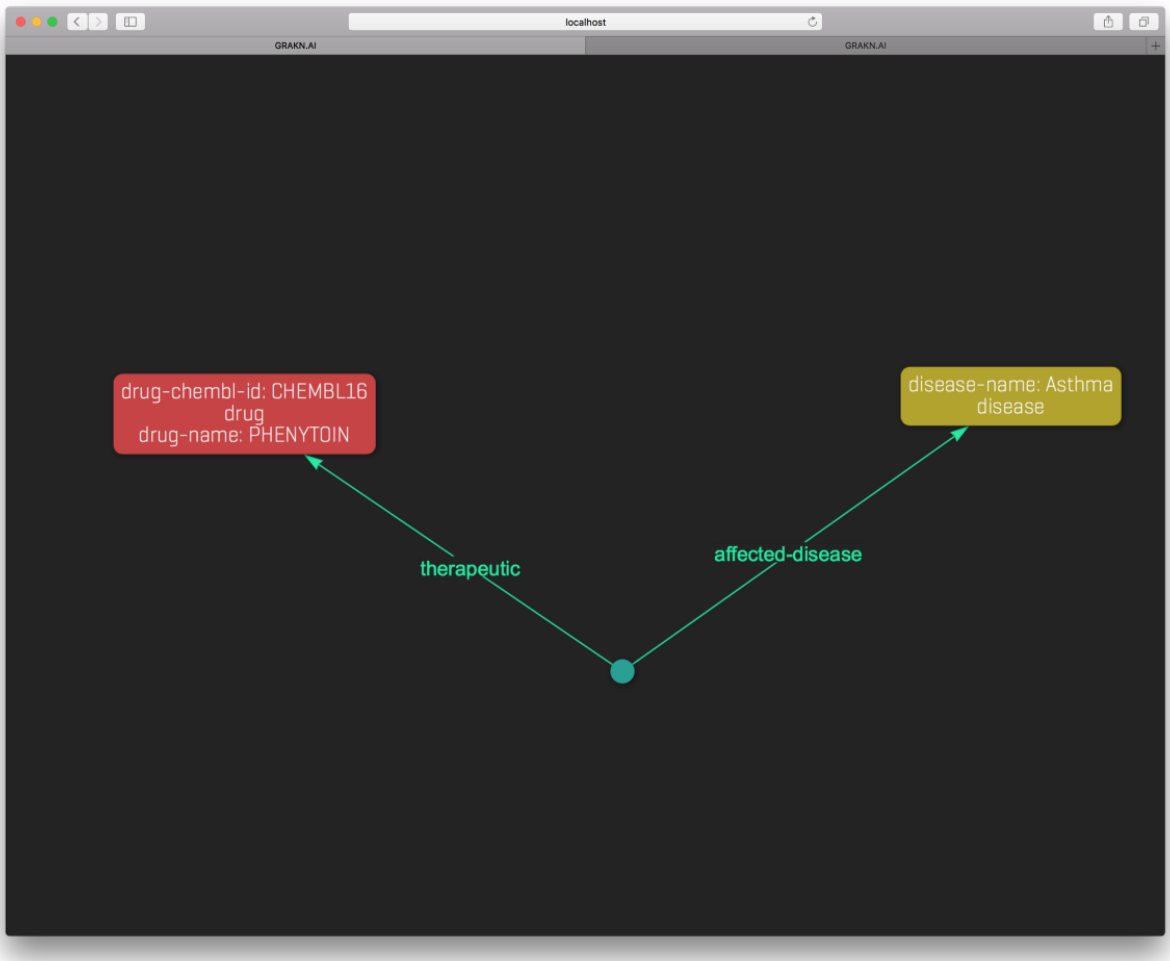
```
when {
 $di isa disease;
 $pr isa protein;
 $pr2 isa protein;
 $pr != $pr2;
 $dr isa drug;
 (associated-disease: $di, associated-protein: $pr) isa protein-disease-association;
 (similar-protein: $pr, similar-protein: $pr2) isa protein-similarity;
 (target-protein: $pr2, interacted-drug: $dr) isa drug-protein-interaction;
```

```
} then {
 (affected-disease: $di, therapeutic: $dr) isa drug-disease-association;
};
```

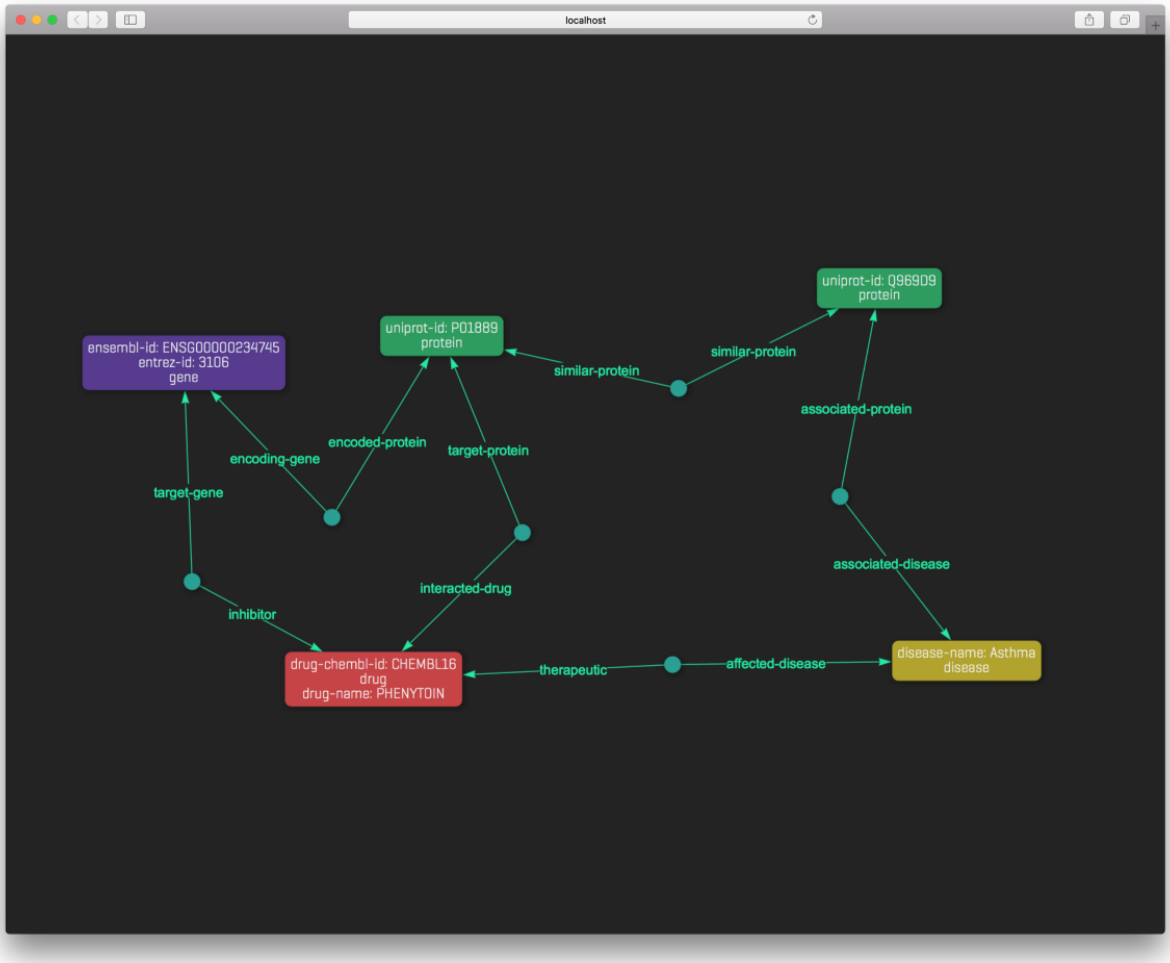
With this rule defined, we can now directly query for drug-disease-association relationships, even though we didn't insert any drug-disease data. But because of the above rule, Grakn will infer for us and find candidate drugs. This is how such a query would look like:

```
match
$di isa disease, has disease-name "Asthma";
$dr isa drug;
$r (affected-disease: $di, therapeutic: $dr) isa drug-disease-association; get;
```

This query looks for potential candidate drugs against Asthma, and below is what it returns. We see that the drug PHENYTOIN may be a potential candidate against Asthma. Remember that no direct association actually exists in the data — these relationships were created (inferred) by Grakn.

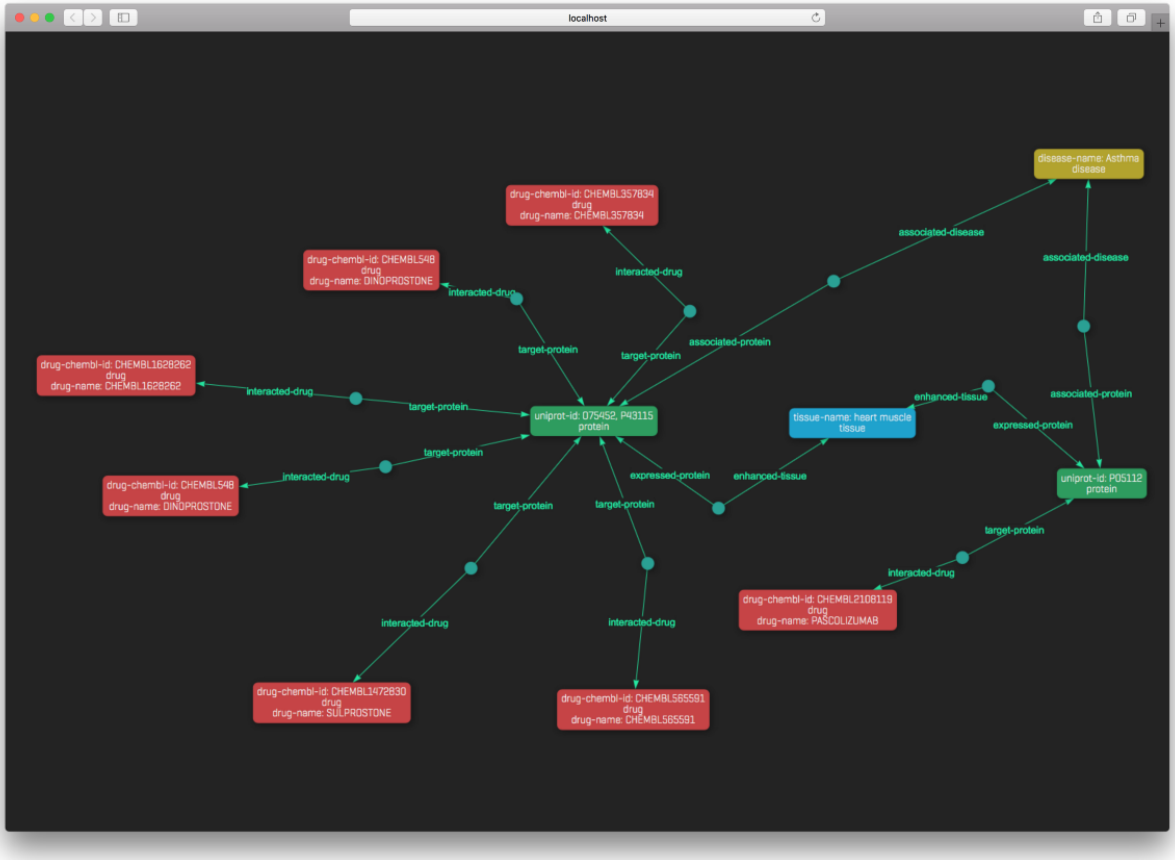


If we want to explore why Grakn returned this answer, we can double click on the relationship, and the graph will expand (see below). We see that protein Q969D9 is associated to Asthma, and has a sequence similarity with P01889, which has a relationship with the drug PHENYTOIN. But as none of the original datasets included protein-drug associations, this relationship is also inferred through another rule — which states that if a disease is associated with a gene (data from DGIdb), that disease should also be associated with the proteins which that gene encodes. Therefore, as PHENYTOIN has been reported to be an inhibitor to gene with entrez-id 3106, the protein it encodes, P01889, gets also associated with it. Exploring such transitive relationships will be of great interest to drug development research.



3. When exploring the biological context of new insights and comparing the network of neighbourhoods of biological components, **traversal type queries** are also important. These can reveal paths connecting components that may not have been anticipated initially. Such queries are easily done using Graql, but would be computationally too expensive for a traditional relational database because of the multiple joins. Below a query is demonstrated that asks for connections between asthma, the heart muscle, proteins, and drugs:

match
$di isa disease, has disease-name "Asthma";
$ti isa tissue, has tissue-name "heart muscle";
$dr isa drug; $pr isa protein;
$pda (associated-disease: $di, associated-protein: $pr) isa protein-disease-association;
$te (expressed-protein: $pr, enhanced-tissue: $ti) isa tissue-enhancement;
$dpi (target-protein: $pr, interacted-drug: $dr) isa drug-protein-interaction;
limit 20; get;



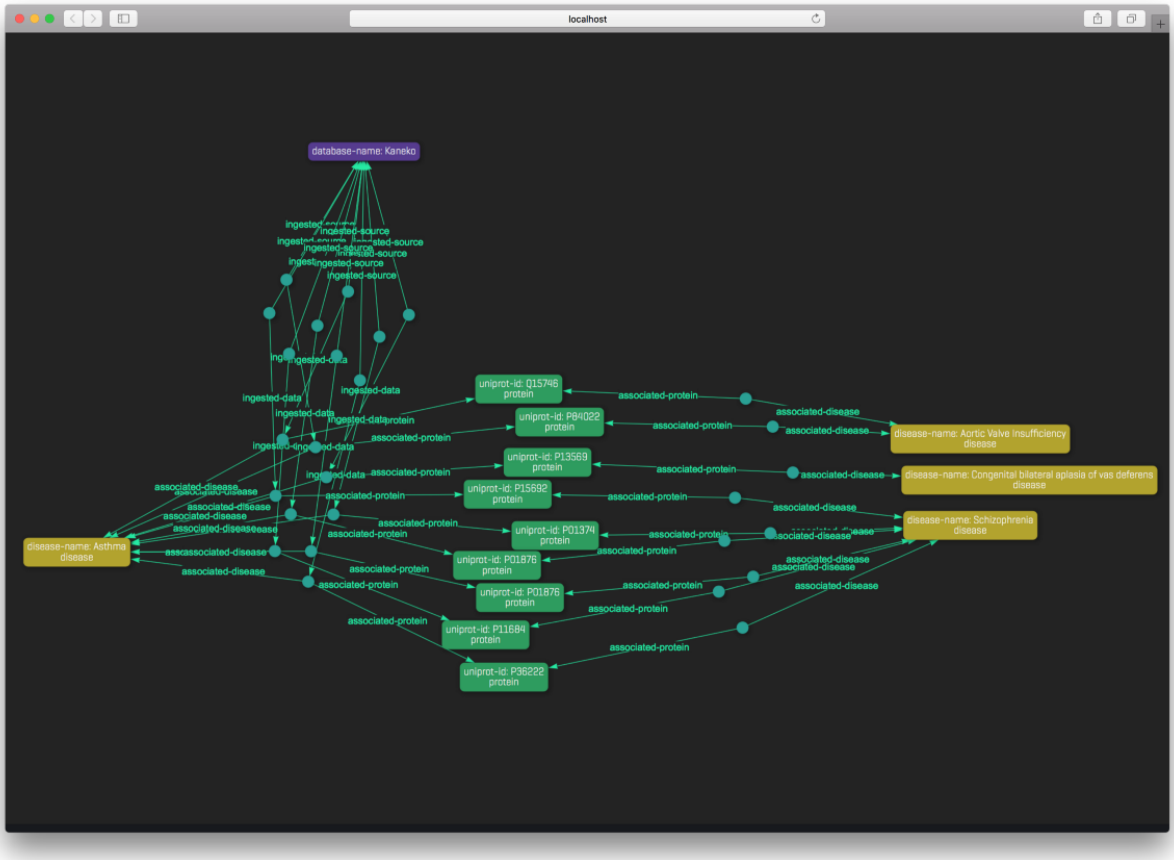
Visualisation of the above query, where red nodes are drugs, green are proteins, blue is the tissue heart muscle, and yellow represents the disease Asthma.

4. We may also want to **query and identify** proteins that are related to a disease from one particular study, and explore how it relates with diseases from other studies. This query is illustrated below:

match
$di isa disease, has disease-name "Asthma";
$pr isa protein; $di2 isa disease;
$db isa database, has database-name "Kaneko";
$di2 != $di;

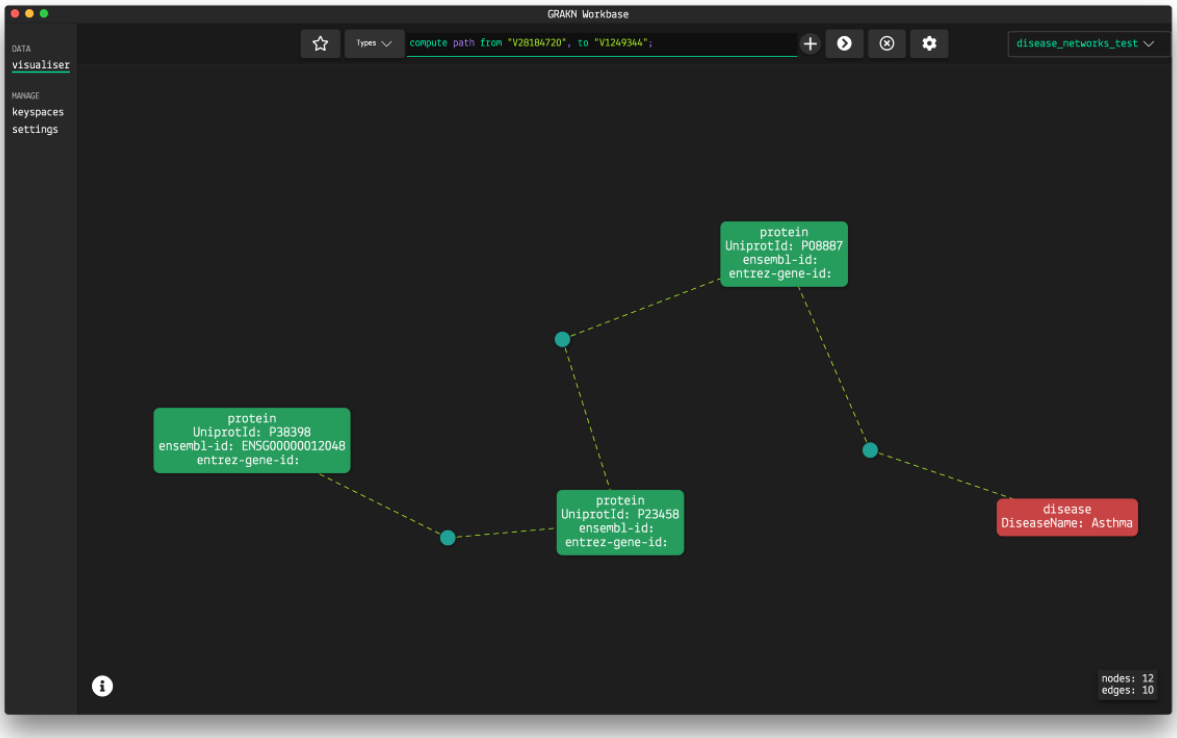
$pda (associated-disease: $di, associated-protein: $pr) isa protein-disease-association;
$di (ingested-source: $db, ingested-data: $r) isa data-ingestion;
$pda2 (associated-disease: $di2, associated-protein: $pr) isa protein-disease-association;
limit 10; get;

This asks for proteins that are common to Asthma and have been reviewed by Kaneko et al. (2013), and are also associated to other diseases. This query can easily be created by asking for all protein-disease-association relationship that are associated with the Kaneko database entity.



Yellow nodes are diseases, and green are proteins. The purple node represents the Kaneko data source.

5. Shortest path queries are also important in order to find the nearest connections between certain biological components. Below is the shortest path between the protein P38398 and Asthma.



Conclusions

Recent advances in omics technologies have created a wealth of high throughput genome-wide scanning data. Therefore, bioinformatics must keep innovating and exploring new techniques for effective and scalable bio-data analysis. In this white paper, we created BioGrakn and demonstrated how using a Grakn knowledge graph we can accelerate the knowledge discovery process in biomedical research.

In summary, there are two areas where Grakn helps in bioinformatics:

1. ***Ingesting and Integrating Biomedical Data into Grakn***
   Due to the unpredictable nature of biomedical research, and the dynamic and constantly changing requirements of biomedical communities, Grakn facilitates the rapid ingestion and integration of new data types. Its schema language, through a hierarchical model and hyper relationships, give a level of expressivity to model non-uniform biomedical data closely guided by the needs of its intended application.

2. ***Bringing Biological Context to Newly Generated Insight***
   New insights/data generated through sequencing/ML algorithms need to be understood in their biological context to advance the knowledge discovery process. Grakn allows for easy ingestion of such new data. What's more, to leverage this biological context, rules can be used to find inferred relationships between unconnected biological components, which may suggest new candidate drugs. The ease of doing traversal type queries is also useful in this knowledge

discovery. And in doing such queries, Grakn's hierarchical model allows us to quickly and easily query for specific type of relationships.

About GRAKN.AI

GRAKN.AI is developed by Grakn Labs Ltd, a London based software company. Grakn Core is open source, published under the GNU Affero General Public License v3.0, Grakn Labs also provides a commercial license to Grakn KGMS.

Find Out More

For more information, please visit grakn.ai or contact or contact us at enterprise@grakn.ai