Engineering Genetic Design Spaces

Using Knox to complete complex manipulations on genetic design space graphs

---

TEAM MEMBERS

Evan Bowman
Melissa Garcia
Erika Schwartz

DESCRIPTION

Our project is centred around a tool called Knox which is responsible for storing and tracking changes to combinatorial genetic designs. Knox had many different functionalities, and at times, these functionalities could be overwhelming to the user. One of our goals was to redesign the user interface to improve the user experience. Moreover, our team refactored Knox's graph algorithms in order for more complex operations to be performed when sampling and enumerating graphs. We also opened these endpoints up to the front end so that the user is now able to work with them.

COMPONENTS

Knox is a simple web application. The back end is written in Java whilst the front end is written in Javascript. We will be detailing only the components of the parts we changed, as Knox had pre-existing code.

Front End

- *Cover page*: The landing page that users see first upon opening the Knox webapp. Previously the application had no cover page, so this is an entirely new feature.
- *Search page*: An interface for visualizing design spaces. We renamed the page to *Explore*, and expanded its set of responsibilities to include design space synthesis and design space removal. The search box now autocompletes as users type.
- *Graph Visualization (search page part II):* we rebuilt the graph visualizer with a

cleaner design, scalable vector graphics, and pan and zoom features.

- *Import*: We regrouped all of the various input tabs into one page (called *Input*), which includes the import settings for CSV, Eugene, and SBOL.

Back End

- *Domain*: The domain of Knox provides all basic classes which represents the units in every design space graph, such as an Node or an Edge. We extended the functionalities of Node and Edge to allow for default properties to be set and for probabilities to exist on edges.
- *Repository*: The repositories interact with the database by performing Cypher queries in order to either Create, Read, Update, or Delete an entry. We extended the DesignSpaceRepository in order to query for specific design spaces correctly.
- *Controller*: The controller acts as the main Web Resource to the application, providing public endpoints for the front end to call. We added in endpoints for sampling, adding probabilities to edges, enumerating, and partitioning.
- *DesignSpaceSampler*: The DesignSpaceSampler acts at the service level of the application, and performs complex operations on the graphs.
    - *Enumeration:* Enumeration can now be applied both by DFS and BFS on a graph. Moreover, the user is now able to specify the number of designs they desire.
    - *Sampling:* The sampling method uses user-defined edge probabilities in order to enumerate designs.
    - *Partitioning:* This method uses the Markov Clustering algorithm in order to determine areas of the design space that are highly clustered.

INSTRUCTIONS

In order to compile our code, the README markdown file that is found on the Knox Github page must be followed exactly.