PCADA (Plasmid Cloning Assembly Display Application)

Nicholas Sacco, Jed Lartey, Michael Sisk, Natalia Villarreal

PCADA is a dynamic price evaluator for DNA sequences designed using Gibson Assembly while also displaying the PCR products and synthetically constructed sequences used to match the user's input. This application is built in conjunction with repository-based plasmid design, or REPP, in order to utilize its software and ability to parse through the Addgene, iGEM, and DNASU libraries. This allows PCADA to still search through numerous databases while looking for potential PCR parts to fill in the user's sequence input. However, rather than using fixed values to determine costs, PCADA looks to IDT DNA's synthetic fragment costs as well as fixed values for the PCR and fragment parameters. Through the integration of company APIs such as Twist Bioscience and IDT DNA, these fixed costs can then become dynamic and provide prices that are accurate and change with developing technology.

Due to complications with gaining authorization for students along with a lack of responses from companies, a proof of concept was created with mock companies represented by CSV files that contained data for fields to match PCADA's configuration files. These mock companies would still represent how different companies may have different prices and restrictions on different variables leading to some companies being cheaper for a particular design than others. Once authorization is granted from these companies, their API should substitute relatively seamlessly in for the mock ones created at the time of submission. Furthermore, PCADA expands on REPP by creating a primer scoring system that scores each primer suggested for PCR builds allowing the user to determine what actions may be necessary for a successful build of their plasmid. Lastly, the addition of a circular plasmid visualizer and a user interface should allow users to access PCADA with ease while also providing a visual example of the parts being suggested by the application in order to minimize build costs.

When creating the mock APIs that would substitute for real companies, it was most efficient to model the files after the configuration file of REPP (config.yaml) as this allowed for a more seamless usage of values found in those CSV files. Three CSV files were created for each "company" which consisted of a main file, a fragment file , and a plasmid file which accounted for the nested structure of config.yaml file for certain variables. The values within these fields were slightly manipulated or updated based on current pricing and research in order to emulate different company's pricing metrics and building capabilities.

A primer scoring system was developed and implemented to further establish PCADA as a more dynamic and realistic model for plasmid design. While REPP is able to find PCR parts to match input sequences, the primers that are associated with that part may not be ideal. There were numerous parameters to evaluate the strength of a primer such as its melting temperature, length, GC content, and potential for secondary structures. Using information found on IDT's website along with other biotech company information, a general scoring system was created that looked for these features using the primer3-py library in Python. A scoring system out of 45 was established and through testing numerous primers in the Addgene database, it was determined that a score of over 35 signaled a primer ready for building, a score between 30 and 35 is a likely satisfactory primer, and a score below 30 tells the user that precautions during PCR such as the usage of PCR enhancers or decreasing the concentration of the given primer. This tool, once again, makes the output of REPP much more usable and translatable to the actual build of the user's sequence.

The Configuration Manager serves as the "bridge" between the configuration files supported in REPP and the potential DNA assembly companies. As previously discussed, the original scope of the project involved incorporating company API calls for pricing and

constructing metrics, but we were unable to integrate any actual company API into PCADA. Therefore, we simulated API calls by creating a local server that responded to basic HTTP requests for cost and construction information stored in CSV files. The Configuration Manager module would query all of the sample companies and build a REPP-compliant configuration file by modifying a template file. Once the configuration file was fully populated, it can be used in the PCADA application. The user should not have to directly create these files; they will be automatically created during the course of normal PCADA operation, simulating the "real-time" access to parameters required for REPP to propose and evaluate a design for the input DNA sequence.

The visualizer is built using a python library, DNA Features Viewer, which was created by the Edinburgh Genome Foundry and functions to produce plots for genome sequencing with overlapping features. We incorporated this into PCADA by parsing the output file As0.output.JSON from REPP and added a sequence analysis function which finds the coordinates at the beginning and end of the fragments within the plasmid map to properly display all elements. Each fragment is displayed in a different color in the plot and is accompanied by a label which displays the type, either synthetic or PCR. If it is a Synthetic type, then the user knows that this part will need to be synthesized and the appropriate cost is displayed alongside it. If it needs to be obtained via PCR, the label will also add a cost as well as the URL to purchase the template plasmid.

The GUI front-end was implemented using SFML, and is designed to be as streamlined as possible. The user just needs to select the target database and enter the file name of the target DNA sequence; once the PCADA application runs, the user just needs to wait for the program to report successful completion of the Repp evaluation, supplemental scoring, and visualization.

Please see the corresponding README for specific build instructions. Essentially, the user should make sure all of the required software is installed, the company server is started, and the GUI application is compiled correctly. Detailed instructions are found in the README.

References:

https://www.idtdna.com/pages/support/faqs/what-secondary-structure-considerations-need-to-be-included-when-designing-primers-for-pcr-

https://www.idtdna.com/pages/support/faqs/how-do-you-calculate-the-annealing-temperature-for-pcr-

http://www.premierbiosoft.com/tech_notes/PCR_Primer_Design.html

https://www.labce.com/spg2095622_melting_temperature_tm.aspx#:~:text=Primers%20with%20melting%20temperatures%20in,a%20tendency%20for%20secondary%20annealing.

https://www.addgene.org/mol-bio-reference/sequencing-primers/

https://github.com/Edinburgh-Genome-Foundry/DnaFeaturesViewer

https://github.com/Lattice-Automation/repp

https://www.sfml-dev.org/index.php

https://pypi.org/project/primer3-py/

https://www.twistbioscience.com/tapi