A Proposal to Bring the Advanced Power and Energy Program's Data Into a Distributed System

Background:

California's electrical grid is dependent on many factors: (1) the amount of electricity that is being supplied, (2) the types of electrical generation that are available e.g. coal, wind, solar, gas, hydro-electric, and nuclear, (3) regulatory issues like monopoly prevention and pricing, and (4) amount of electrical energy that is demanded from the system. The continual increase in demand has to be met for electricity to be reliably distributed across homes in California. An approach to solving this problem is to take advantage of renewable sources. There have been strives in legislation, like Assembly Bill 32 (AB32), that aim to make changes to the electric power system by reducing greenhouse emissions. Senate Bill 1078 has created renewable electricity goals, set forth, to increase the integration of renewable generation onto California's electrical grid [1]. Some renewable resources have predictable behavior, however resources like solar and wind, require a more complex analysis for understanding their effect on the electrical grid.

The Holistic Grid Resource Integration and Deployment tool (HiGRID) was devised to incorporate different types of renewable generation such as dispatchable, baseload, and intermittent to determine their effects on the cost and benefits of fully implementing those generation resources. Recent studies used the HiGRID model to analyze plug-in hybrid electric vehicles. The effects of solar and wind power on the operation of balancing generators have been examined using HiGRID [2]. Within the HiGRID model there are different modules dedicated to analyzing various aspects of load dependent characteristics. These modules include the Renewable Generation Module, Dispatchable Load Module, Cost of Generation module and Balance of Generation Module. There are dependencies between each of these modules as seen in Figure 1. Data are fed as an input to the modules which affect other aspects of different modules, due to the interdependent relationships. As HiGRID continues to develop, the amount of data required to run various analyses increases. This introduces complex problems for the future of HiGRID. Complexities include resource allocation, distribution, and retrieval.

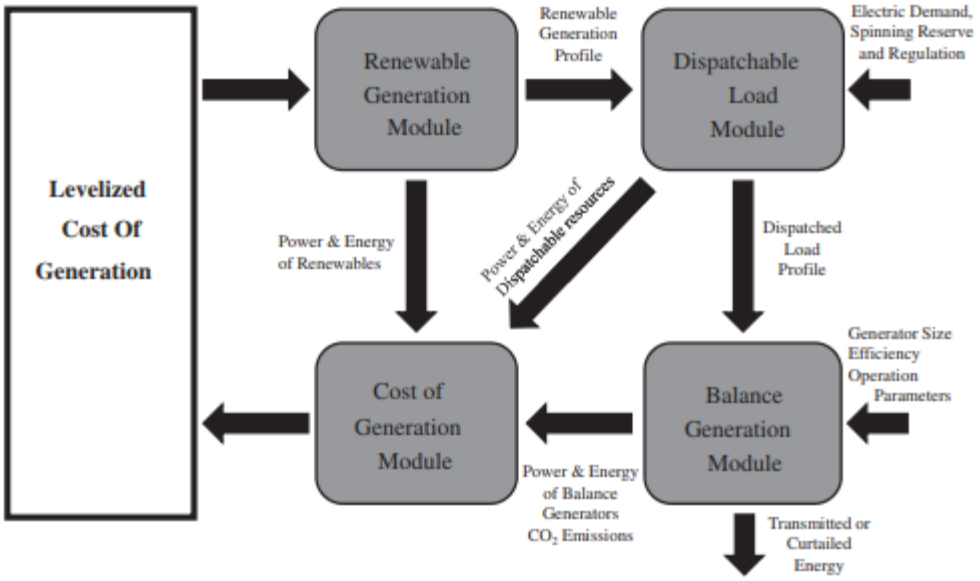

Figure 1: HiGRID model flowchart [1]

Resource allocation becomes problematic because every computer that is used to work on the HiGRID model becomes cluttered with many files needed to run the code. Valuable storage space as a result is wasted. Resource distribution needs to be resolved because the way in which data are given to the developers is unorganized. Two questions become immediately highlighted: which developers have the access to the old data? which developers have the new data? Resource retrieval becomes a difficulty as the amount of code, that uses a particular resource, increases. In the HiGRID model, columns are not referenced by their name, but instead by their location in the table. To new developers these numbers used for location may seem arbitrary and difficult to remember. A lack of coding conventions can lead to misunderstandings of how the code is being used.

To tackle these problems, a research database can be implemented for writing and retrieving data. While creation of the database is simple, the design of a fast and efficient database is the difficulty [3]. There are many important factors to consider when designing the database: (1) the database must ensure important data (i.e. data that are actually relevant to the model) are stored, (2) it must leave out unused data in the model that would otherwise deplete computer resources, (3) it must have secure protocol that will not be written into without proper authorization, and (4) it must be efficient in the process of storing and ordering

data. Currently, developers require copies of the complete data set in order to run the HiGRID models. This does not always work, it is inefficient, and in the long term very confusing for researchers to expand upon or use existing code made by a previous developer.

Objective:

The objective of this research proposal is to improve accessibility of the HiGRID model by making data available on a central database and improving readability. There are two main parts in meeting this objective. The first part is for the development of a database which will store all data required to run HiGRID. This database will be kept optimal in storage size and speed and made configurable for future additions of data. Data retrieval from the database will be routine, requiring strong database security measures to ensure safety. To meet the second part of the objective, I propose to provide a data management API that would be used by the HiGRID model. The current model will be made easier to use, allowing researchers to leverage the HiGRID model to conduct novel research. The API will have to meet criteria such as speed for data processing, ease of use for developers, and modularity for the expansion tools that the API can provide.

Challenges:

The amount of data utilized by the HiGRID model are large and spans across more than 100 separate files.

1. There are roughly 150 files that provide information to the HiGRID model in varying formats, sizes, purposes and data types.
2. The two different formats of files include Microsoft Excel Spreadsheets (.xls) and MATLAB's own format (.mat)
3. The data types must be chosen to optimize storage space required for the server. This is crucial to handle the large data sets.
4. Developers continuously add to the model. New modules may require different information from existing tables or may require new data entirely.
5. The API must be robust, modular, and easily configurable for additional data.

Design and Feasibility

The first task is to understand how each module accesses and uses its data. In total there are 39 MATLAB source code files; each with their own variables derived from data that are either "hard-coded" or are accessed from files. Data in these files are not always well formatted for a database. These data need to be converted into well formatted database tables. Additionally, some tables need to be altered to deal with excess columns of data or columns that should otherwise be moved to different locations in the database. This will be done with every data file that is currently in use.

In Excel and MATLAB more space is often used to store double precision data types requiring 8 bytes to store each data point [4,5]. For the data used by the HiGRID model, that level of precision is not required, and the storage space is wasted. Most of the data that is used only requires 4 bytes for single floating-point precision. The data table used to store local wind vectors, as shown in Figure 2, only stores at most, 6 decimal places. A float can be used when calculating the regional wind profiles for each city. Any time there is an unneeded level of precision required for data, the values will be stored, and cast down into a lower precision storage variable.

```
1 •  SELECT * FROM higrid.beaumont_stream_hour;
```

Result Grid | Filter Rows: | Edit:

ID	53 TB	109 TB	147 TB	205 TB
1	1522.374143	2959.716857	3996.610571	5721.395143
2	1491.479857	2917.174143	3959.170143	5666.172571
3	1518.019	3101.968143	4177.386714	5912.595143
4	1549.523	3198.326	4286.416143	6020.232857
5	1540.409143	3150.609429	4227.915429	5962.349857
6	1542.127143	3051.528571	4080.498143	5798.119143
7	1548.202429	3077.656286	4061.228714	5718.468286
8	1499.297	2922.609	3943.150857	5589.370857
9	1502.963143	3029.950714	4069.53	5710.037857
10	1502.260571	2986.395	4004.171	5595.353857

Figure 2: Wind Vector per Turbine Count in Beaumont, CA (excerpt)

A model for testing will be created by running scripts through MATLAB using system command calls. This is to evaluate the success and the shortcomings

of any function that will be later fully implemented into the API. After all of the problems for each function have been worked out, the code will be subsequently added to the API's functionality. The API will be routinely tested on existing MATLAB code before it becomes available to the developers on HiGRID.

Personal Responsibility:

My role in this project will involve coding functions to retrieve data from the database. Programming the functions will involve: (1) researching each of the modules in the model that use data extraction tools in MATLAB such as the xlsread or the load function, (2) designing a naming convention for the functions that are specifically made for each code section of the modules, and (3) creating a fast system for parsing and packaging data retrieved from the database into the MATLAB workspace. More responsibilities for this project include creating the database from which the scripts can retrieve data. That involves managing the database and all of its resources, ensuring accessibility of each data table to every developer that requires it. I will be keeping an updated record of newly created modules that will be implemented in the future, or modules that have just been recently added. I also need to make sure that the database can be easily updated and maintained for future use by other developers.

To highlight, these are my responsibilities:

1. Import existing and relevant data into the database, while maintaining the security and safety of the data
2. Provide a useful naming convention for developers
3. Ensure that there is a fast implementation of data retrieval code
4. Making the code easily expandable

Timeframe:

After the timeframe of 10 weeks, the projects core components, the database design and implementation and the database data retrieval functions, will be complete. Database design and implementation are complete when all input data used by the current version of HiGRID are represented by a record in the database.

The readability of the code will improve as a result of providing a simple toolset of functions for the developers to work with. To complete this objective a strong understanding is needed of how every data pull operation is used in regard to the context of code.

Weekly Schedule

- Implement Data Management API (API)
- Database Design and Implementation (DB)

Week	Module	API	DB
1	Renewable Generation	Read MATLAB code and design data functions	Design layout of tables
2	Renewable Generation	Write data retrieval functions, concurrently test program	Import tables with addition of each data retrieval script
3	Balance Generation	Read MATLAB code and design data functions	Design layout of tables
4	Balance Generation	Write data retrieval code, concurrently test program	Import tables with addition of each data retrieval script
5	Cost Generation	Read MATLAB code and design data functions	Design layout of tables
6	Cost Generation	Write data retrieval function, concurrently test program	Import tables with addition of each data retrieval script
7	Dispatchable Load	Read MATLAB code and design data functions	Design layout of tables
8	Dispatchable Load	Write data retrieval function, concurrently test program	Import tables with addition of each data retrieval script
9	Migration of database onto a permanent server	Ensuring functions can call from new	Moving tables, and finalizing table

		location	names, and columns
10	Final production server testing	Testing data retrieval functionality	Testing query performance

References

1. Eichman, Mueller, Tarroja, Schell, Samuelsen. "Exploration of the integration of renewable resources into California's electric system using the Holistic Grid Resource Integration and Deployment (HiGRID) tool." *Journal of Energy* 23 Jul. 2013. InPress.[Accessed:Online] Available: https://www.sciencedirect.com/science/article/pii/S0360544212008857
2. Tarroja, Eichman, Zhang, Brown, Samuelsen. "The effectiveness of plug-in hybrid electric vehicles and renewable power in support of holistic environmental goals: Part 2 – Design and operation implications for load-balancing resources on the electric grid" *Journal of Power Sources* 15 Mar 2015. In Press. [Accessed: Online] Available: https://www.sciencedirect.com/science/article/pii/S0378775314010520
3. Cross, Palmer, Stephenson. "How to design and use a research database" *Diagnostic Histopathology* April 2018 [Accessed: Onli ne] Available: https://www.sciencedirect.com/science/article/pii/S1756231717301494
4. "Double." MATLAB & Simulink, www.mathworks.com/help/matlab/ref/double.html.
5. "Data Types Used by Excel." About Processes and Threads (Windows), msdn.microsoft.com/en-us/library/office/bb687869.aspx.