MEng Group Project Specification & Design

Cristian Badoi, Aaron Butterworth, Daniel Gardam

1 Project Summary

The aim of this project is to create a system capable of calculating, storing and predicting formulae of chemical compounds given a set of precursors, with a focus on optimization and finding an acceptable solution within a given set of bounds. The program is designed to be remotely accessible, with a web server providing a user front end and a space for necessary calculations, and a database for the purposes of storing chemical data and user account information.

The project will primarily use C++ for computation, NodeJS with several popular packages for the web server and user interface, and MongoDB for the database system.

This project is produced in conjunction with the Materials Innovation Factory and is intended for their use. The development and evaluation of the project will be performed in cooperation with the facility.

This project is mainly based on the prior research topics and problems encountered by researchers in the MIF, for example having a person test possible compounds and slowly refine the formula based on their results is a time and material inefficient process, therefore if we can suggest compounds which are either direct results or very close approximations to their desired outcome then this process can be minimised.

2 Design Overview

2.1 Expected Components

2.1.1 Web Server

The main interface to the system will be supplied by a web server to allow easy and portable access by members of the MIF. The web facing part of the system will be comprised of several smaller packages all working together under NodeJS:

NodeJS — A JavaScript runtime that allows execution of JS outside the context of a web browser.

Express — An open source web application framework for NodeJS, it is responsible for the high level web server logic and allows a high level interface to web traffic as it handles response codes, etc. itself.

Pug — An open source high-performance template engine, it allows us to alter the contents of a web page just before serving it to the user, allowing for finer levels of customisation while keeping the code base manageable.

Express-Session —	An add-on for express that gives us easy control of user sessions. This allows us to keep track of user specific data, such as preferred precursors or recently searched compounds.
bCrypt2 —	A widely trusted package for providing user password encryption.

2.1.2 Computation

The actual computation will be handled by C++ programs with some small python helper scripts for unit conversion and other pre/post processing.

This is done for several reasons, firstly to allow a significant performance increase over JS; secondly to allow us access to widely known mathematics libraries; and thirdly to allow easier modification of the source code by others in the MIF as these are their preferred languages as opposed to JS, this is also the start of an ongoing project which will be maintained and expanded upon after us so maintainability is an important factor.

The first step of computation is the stoichiometry calculator. This takes a selection of user defined elements, and then calculate all possible balanced compounds that may be produced by a chemical reaction of these elements. This process also takes into account user limits such as the maximum amount of atoms in the resulting combination to refine the search and prevent the creation of a technically correct but practically unreasonable result.

The programs data sources are the web interface which will invoke the users query and the database for relevant information about each element, its output is piped back to NodeJS where it may be cached in the database if it is a common search, the result is then displayed to the user via the web interface.

The second stage is the precursor calculator. Its input is a desired ratio of elements, and a set of available chemical precursors. It calculates possible combinations and quantities of the precursors that when mixed together create the desired target ratio. User bounds may also be given to keep the results practically usable.

2.1.3 Database

With the theme of future expansion the database will be hosted on a separate server instance running independently of the NodeJS process, this allows for easier expansion of the system if it cannot handle the user load placed on it as both the web server and database can be scaled completely separately from each other without modification to the program code.

The database will not only store the information needed to run calculations, but will also store the solutions to commonly executed combinations so as to reduce the computational demand by allowing these solutions to be quickly retrieved rather than repeatedly calculated.

3 Algorithm Design

3.1 Stoichiometry Calculator

The queried elements are transformed into a matrix with the following properties:

If the user selected Al and O with a limit of 5 atoms in the result:

$$\begin{array}{cc} & \begin{array}{cc} \text{Al} & \text{O} \end{array} & \\ \mathbf{A} = & \begin{bmatrix} 1 & 1 \\ 3 & -2 \end{bmatrix} & \begin{array}{l} \text{Initial Quantity} \\ \text{Charge Imbalance} \end{array} \end{array}$$

$$\mathbf{B} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{array}{l} \text{Resulting Proportion} \\ \text{Desired Charge Imbalance} \end{array}$$

$$\begin{bmatrix} 1 & 1 \\ 3 & -2 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

This gives us the matrix form of the system of equations we must solve. However the charge imbalance in the matrix **A** is not constant for all elements, in this example O can exists with $-2, -1, +1, +2$ as its charge, solving the equations above only allows for Al_2O_3. Therefore we must also consider all permutations of element charges, in this example O has a total of 4 states, and Al a total of 3 giving a total of 132 permutations.

However, each permutation is quick to solve. Continuing the above example:

Via LU decomposition:

$$\begin{bmatrix} 1 & 1 \\ 3 & -2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 \\ 0 & -5 \end{bmatrix}$$

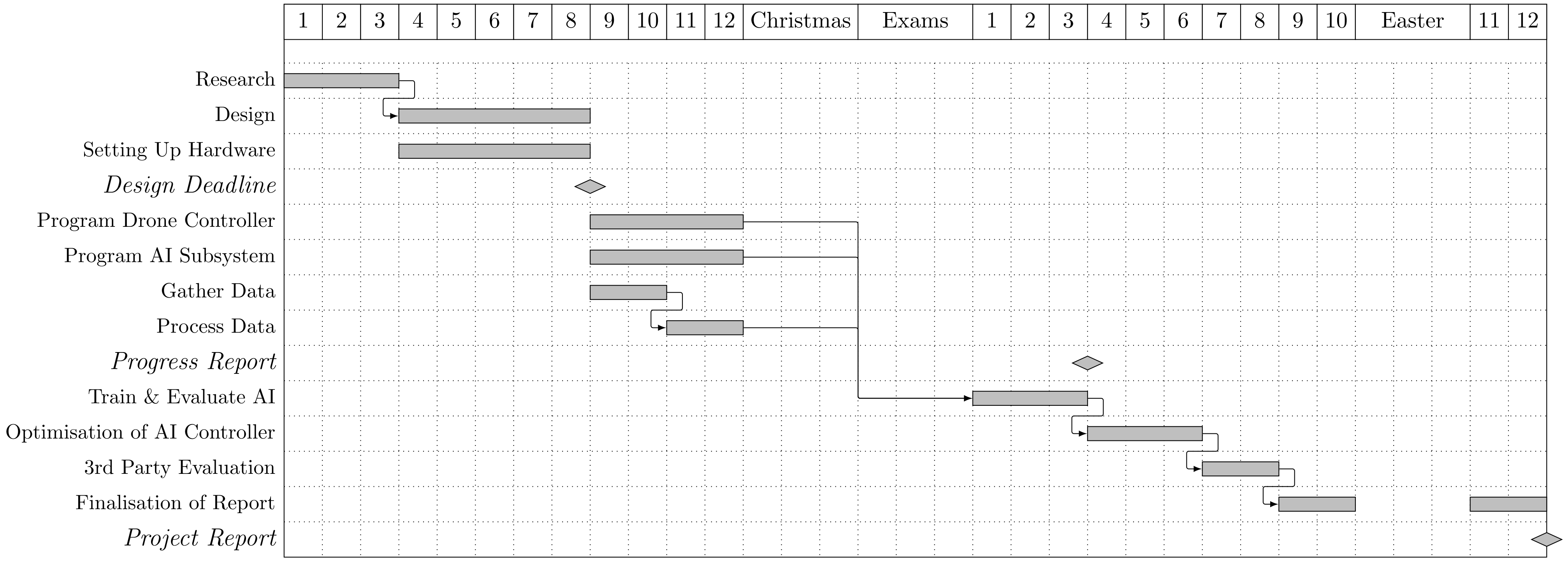
$$\begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix} \times \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \\ 0 & -5 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -3 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}$$

While the example seems trivial when many elements are selected the system of equations grows rapidly and the merits of this method become more apparent.


1
2
3
4
5
6
7
8
9
10
11
12
Christmas
Exams
1
2
3
4
5
6
7
8
9
10
Easter
11
12
Research
Design
Setting Up Hardware
Design Deadline
Program Drone Controller
Program AI Subsystem
Gather Data
Process Data
Progress Report
Train & Evaluate AI
Optimisation of AI Controller
3rd Party Evaluation
Finalisation of Report
Project Report