Developing a Framework to Analyse Clustering Algorithms for Molecular Dynamics Trajectories

Project Proposal

Nicholas Limbert
lmbnic008@myuct.ac.za
University of Cape Town
Cape Town, South Africa

Wen Kang Lu
lxxwen005@myuct.ac.za
University of Cape Town
Cape Town, South Africa

Robyn McKenzie
mckrob018@myuct.ac.za
University of Cape Town
Cape Town, South Africa

CCS CONCEPTS

• **Applied computing** → *Computational biology*; • **Mathematics of computing** → *Cluster analysis*; • **Theory of computation** → *Unsupervised learning and clustering*.

KEYWORDS

Clustering Analysis, Molecular Dynamic Simulations (MD), Carbohydrate Molecules, Dominant Conformations

1 PROJECT DESCRIPTION

Molecular dynamics (MD) simulations are often used by researchers to determine the dominant conformations of molecular structures. These simulations output what is known as a trajectory of the molecules observed. A trajectory acts as a record of the movement of each atom over time in the form of snapshots, or frames. Problems arise where the number of frames far exceed what can be manually analysed. As the understanding of these macro-molecular structures is important for many applications, such as drug-receptor interaction and protein folding [19], efficient methods need to be used to gather such information effectively. Clustering algorithms are commonly used given their ability to partition a data set into different groups. In this case, the goal of the algorithm would be to group similar molecular structures together into families to give a summary of the various conformations.

Clustering algorithms are not often applied to trajectories of flexible molecules, such as carbohydrates. Prior research has primarily focused on clustering nucleic acids and proteins, which are generally not very flexible. Trajectories of flexible molecules will differ from those of stable molecules. Hence, the algorithms which are successful in clustering the one type of trajectory cannot be assumed to be successful in clustering the other. As there are many clustering algorithms available, a variety of algorithms must be implemented and investigated before we can identify those which may be useful for clustering trajectories of flexible molecules. We will explore three sets of algorithms: standard, widely available algorithms, more complex methods, and novel approaches.

Standard algorithms are those that provide a well-known approach to generating clusters and have already been implemented for clustering MD simulation data.

We explore a range of hierarchical clustering algorithms, specifically of the agglomerative type. This is a bottom-up approach whereby each data observation starts in a single cluster and is merged as we move up the hierarchy. The clusters are merged according to a linkage criteria. Hierarchical has the advantage of being easy to implement and easy to understand cluster formations. The algorithm produces a hierarchical tree (dendrogram) which allows us to understand the underlying structure of the data. We focus on variations of hierarchical clustering algorithms to determine which is most effective for MD data.

The Quality Threshold (QT) algorithm [13] aims to generate high quality clusters by iteratively adding observations to a cluster while not exceeding the pre-defined cluster diameter. Observations are added to minimise the cluster diameter. The Quality Threshold (QT) algorithm has previously been implemented for clustering a range of MD data, however there are some inconsistencies related to certain implementations [11]. We therefore aim to make use of the implementation from González-Alemán et al. which is inline with the original proposed algorithm from Heyer et al. This will allow us to determine the effectiveness of the QT algorithm in clustering highly flexible molecules.

The self organising map (SOM) is an artificial neural network often used to reduce high-dimensional data sets into a two-dimensional U-matrix which can visualise clusters based on colour intensity. This makes it perfect for clustering MD trajectories as often times millions of frames have to be analysed while also requiring some form of visualisation. SOM clusters MD by employing neurons that learn to recognise the patterns of the given molecules in a trajectory, allowing it to classify and partition conformations.

Clustering algorithms can also be more complex in nature, as algorithms can be combined or expanded upon to produce improved clustering. Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [6, 7] uses a density-based clustering approach combined with a hierarchical component. This algorithm has been applied in a number of fields, including MD [17]. Another algorithm, Intelligent Minkowski Weighted K-Means (iMWK-Means) [9] expands upon the k-means algorithm by performing iterative rescaling. Compared to more simple algorithms, HDBSCAN and iMWK-Means can be used without any user-specified parameters, which means that no assumptions about the data need to be made, and they each take steps to handle noise and outliers in the data. Melvin et al. [17] specifically recommend HDBSCAN for intrinsically disordered proteins and believe that it could be used with iMWK-Means, either through integrating HDBSCAN into iMWK-Means, or by feeding the clusters produced by HDBSCAN into iMWK-Means, to further improve the resulting clusters.

Novel clustering algorithms in the form of metaheuristics have become increasingly prevalent in the past few years. Metaheuristics take a stochastic approach and treat clustering problems as optimisation problems by minimising or maximising some objective function(s) similar to that of partitional clustering. Bandyopadhyay et al. [3] introduced a multi-objective algorithm, AMOSA, based on the principle of simulated annealing. Multiple objective functions are optimised simultaneously, producing a set of Pareto-optimal results where no solution can improve one of its objective functions without degrading another. This feature could be particularly useful when clustering molecular dynamics as often times there may be ambiguous clusters that consider only certain traits of a trajectory. By providing a set of results, the user may then select a solution that best matches their interests.

2 PROBLEM STATEMENT

There are many clustering algorithms available and many factors differentiate them. Certain algorithms may be more vulnerable to outliers and noise, or may have a tendency to produce clusters of similar sizes and shapes. Additionally, algorithms differ in the parameters that they require from the user, which is important as these can introduce bias or force assumptions about the data to be made. While some clustering algorithms have been implemented and tested for clustering MD trajectories, there are many that have not. Algorithms that have been included in standard MD simulation or visualisation software, or are otherwise well-known and commonly applied to MD, do not necessarily produce the best results. Specifically when clustering flexible molecules, it is not obvious which algorithms may be most suited to clustering the trajectories. As there are many factors that affect the results of clustering algorithms, many algorithms must be evaluated for this use case in order to select useful algorithms.

3 AIMS

We aim to implement hierarchical clustering, Quality Threshold clustering and SOM, which are standard algorithms commonly used for MD, HDBSCAN and iMWK-Means, which are more complex clustering algorithms and have the possibility of being used in combination, as well as AMOSA, which is a novel approach to clustering and has not previously been used for MD.

We aim to investigate and compare the performance of the above algorithms in order to identify which are the most effective in clustering trajectories of flexible molecules. An effective clustering algorithm is one that generates high quality clusters with good partitioning. This means that the clusters have high intra-cluster similarity and low inter-cluster similarity. This indicates how well a cluster is formed and how similar the objects within a specific cluster are. Inter-cluster similarity is used to determine how dissimilar different clusters are from one another. This is also referred to as cluster separation.

4 RESEARCH QUESTIONS

We aim to address the following research question.**Which clustering algorithms are most effective in clustering Molecular Dynamics (MD) simulation data of highly flexible molecules?**

Each member has also outlined their own individual research questions which pertain specifically to their contribution.

Nicholas

(1) Which hierarchical clustering method of Ward, maximum, single and average linkage produces the most effective clustering results of MD trajectory data?
(2) Is the Quality Threshold Algorithm able to effectively cluster MD simulation data from highly flexible molecules?

Wen Kang

(1) What combination of objective functions are best suited for clustering trajectories when using the AMOSA metaheuristic?
(2) Given that self organising maps are a commonly used artificial neural network for clustering trajectories, how does it compare to AMOSA in terms of cluster quality?

Robyn

(1) How do HDBSCAN and iMWK-Means perform when clustering trajectories of flexible molecules, given that they are both able to handle noise in the data?
(2) Can HDBSCAN and iMWK-Means be used together, either in combination or succession, and does this provide better clustering, based on CVIs and visualisation, than either of the algorithms alone?

5 RELATED WORK

5.1 Cluster Analysis Validation

Validation is a fundamental part of clustering analysis and it is essential for achieving meaningful results. Cluster validation allows us to generate either graphical or numerical value summaries of the clustered data in order to determine how well a clustering algorithm has partitioned and clustered the data. Due to the nature of clustering, there are a wide variety of algorithms available which in turn has allowed for many validation techniques to be developed. There is, however, no consensus on a singular clustering validation measure.

Arbelaitz et al. [2] provide an extensive study on the comparison of multiple different cluster validation indices in order to provide better insight into selecting a suitable validation measure. They conclude that the choice of valuation measure should be selected based on the dataset. They do, however, indicate that the Silhouette, Davies-Bouldin and Calinski–Harabasz perform considerably well with many different datasets. Pablo et al. [14] develop a framework for comparing relative clustering validation measures. Notably, a single index cannot capture all the aspects involved in the clustering problem hence the need for multiple indices. Due to the fact that a single index may fail in capturing all aspects of the problem, they propose the use of multiple validation indices. These should again be selected based on the dataset.

As mentioned previously, the main issue with clustering validations is the wide variety of different metrics available and how to select a metric suitable for the dataset. A range of validation indices should be selected to try and capture all different aspects of the clustering problem.

5.2 Validation Techniques for Clustering Molecular Dynamic Simulations

Clustering analysis has already been implemented with a range of MD simulations. Below we outline some of the validation techniques and indices implemented in previous research. This provides a guideline to determine which validation indices may be best suited for our research.

Shao et al. [18] use several validation metrics such as the pseudo-F statistic, Davies-Bouldin index (DBI) [8], SSR/SST ratio and the "critical distance" in the clustering of MD simulations. The DBI and pseudo-F statistic are used to determine the overall compactness and separation of all the clusters. The SSR/SST and critical distance are used to determine the ideal cluster count. It is shown that low DBI values and high pseudo-F values indicate good partitions. The validation indices behaved as expected, with high pseudo-F statistic and low DBI values for good cluster formations. They also observe a constant SSR/SST ratio when the ideal cluster count is reached. Together these metrics provided comprehensive validation of the algorithms implemented, however, none could be used as a metric individually.

Abramyan et al. [1] make use of three internal cluster validation measures; Calinski Harabasz (CH), Davies-Bouldin (DB), and Silhouette (S) indices when clustering of MD simulations. These simulations focus on clustering protein adsorption MD simulation data. High Calinski Harabasz, Silhouette indices and a low Davies-Bouldin index indicate good cluster formations. All the indices implemented by Abramyan et al. are internal validation measures. These indices are used to determine the compactness and separations of clusters formed.

Melvin et al. [17] implement a wide variety of different algorithms however, only make use of the Silhouette index [12] as their basis of comparison. The high Silhouette index indicates ideal cluster formations with high intra-cluster similarity and low inter-cluster similarity.

It is evident that many papers make use of a variety of cluster validation indices. This further reinforces the idea that multiple indices should be used in order to better understand the cluster analysis output. We will focus on the use of internal validation indices that determine the overall compactness and separation of cluster formations. Specifically we aim to make use of the Calinski Harabasz (CH), Davies-Bouldin (DB) and Silhouette (S) indices.

5.3 AMOSA and SOM

Bandyopadhyay et al. [3], as mentioned, proposed the AMOSA algorithm based on simulated annealing. The algorithm was tested against two multi-objective evolutionary algorithms NGSA-II [10] and PAES [15]. It was found that AMOSA performed better overall, producing more distinct solutions in its Pareto-optimal set of results. This may prove beneficial when disambiguating similar molecular conformations.

A self-organising maps (SOM) solution was used by Bouvier et al. [4] to specifically cluster the conformations of a trajectory undergoing multiple protein folding and unfolding events. Not only was the method able to successfully cluster the trajectory, the package included visualisation by means of a U-matrix.

5.4 HDBSCAN and iMWK-Means

HDBSCAN [6, 7] is density-based algorithm, which constructs clusters from areas of high data density separated by low data density, and outputs a hierarchical tree [17]. Its ability to handle noise in the data has led to it being recommended for clustering trajectories of intrinsically disordered proteins. Additionally, HDBSCAN is useful for exploratory clustering as it can be used non-parametrically. Melvin et al. [17] applied HDBSCAN to a series of nucleic acids and proteins and found it to be effective for detecting large scale conformational changes in trajectories of stable molecules. Although a larger number of clusters were produced when clustering trajectories of unstable molecules, HDBSCAN was still able to detect the intermediate stable conformations of the molecule.

iMWK-Means [9] is a variant of k-means which, like HDBSCAN, can be used non-parametrically [17]. In not requiring a desired cluster count to be supplied, it eliminates the key weakness of traditional implementations of k-means. iMWK-Means eliminates the need for cluster count parameter by overestimating the required number of clusters and then iteratively performing k-means and rescaling each of the data points based on the results. The algorithm handles noise and outliers by assigning low weights to the points in the sparse, or noisy, regions of the data, which causes them to affect the resulting clustering to a lessor degree.

As with HDBSCAN, Melvin et al. [17] applied iMWK-Means to a number of nucleic acids and proteins and found it to be ideal for detecting subtle differences between conformations. As iMWK-Means is primarily suited to clustering trajectories of stable molecules, Melvin et al. outline the possibility that HDBSCAN could the replace the final step of iMWK-Means, or that iMWK-Means could be applied to the stable clusters produced by HDBSCAN to reveal finer resolution details.

6 PROCEDURES AND METHODS

As a number of algorithms will need to be implemented, we propose the development of a framework which will facilitate the application of each of the algorithms to the MD trajectories. This framework will also provide the opportunity to include options to validate and visualise the clusters, which will be necessary in order to evaluate and compare the relative success of the algorithms.

6.1 Framework

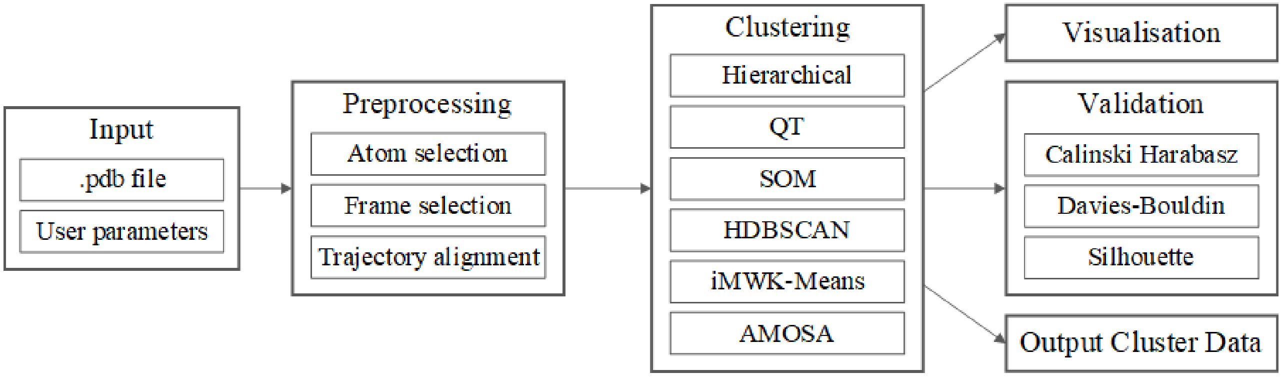
We aim to develop a framework whereby the user can specify the input file, algorithm and validation index via command line arguments. As the result of each clustering algorithm is dependent on the selection of the atoms and frames to be included in the clustering, the framework should ideally include preprocessing functions, to facilitate this, as well as post-processing functions. Below we outline our envisioned implementation for each section.

The framework will be structured as a pipeline **(Figure 1)** to allow for additional algorithms, and other functionality, to be easily implemented at a later stage. It will be implemented in Python, as the majority of the existing packages we plan to use are Python libraries. Additionally, Python code is more easily understood than that of many other languages, and will be simple to maintain and extend in the future.

6.1.1 Input.

The program accept a Protein Data Bank (.pdb) file as the input dataset. These files contain three-dimensional structures of molecules and their change in conformations over a time period. The focus of this project is on clustering highly flexible molecules, therefore we will use MD simulation data of carbohydrate conformations. The user will provide the input file name as a command line argument.

Figure 1: Pipeline of Framework Design



6.1.2 Preprocessing.

The user may wish to preprocess the files. We plan to implement two core features that will allow atom or frame selection for clustering. Basic editing of the PDB file will also be available. This will include generating a new trajectory file by taking a range of selected frames from another PDB file. This reformatting, specified by the user, will be done before the initial clustering. All files must go through prepossessing in order to ensure the correct format for the clustering algorithms implementation. An additional feature that may be implemented will allow the user to align the trajectory file. This is subject to the completion of the core functionality of the entire framework.

6.1.3 Clustering.

The preprocessed data is clustered with the algorithm specified by the user. The user will have a range of clustering algorithms to choose from. These include, Hierarchical, QT, HDBSCAN, iMWK-Means, SOM and AMOSA. All clustering algorithms are configured to accept data in the format output by the preprocessing phase. The clustering output must also be uniform across each algorithm implementation to allow for the validation techniques and visualisation to be used.

6.1.4 Validation.

There will be numerous validation measures available for the user. We recommend using multiple Cluster Validation Indices (CVI's). If the user has specified a CVI, or set of CVIs, to be used, this will be applied to the output and results will be written to a file. We aim to implement the Calinski Harabasz (CH), Davies-Bouldin (DB) and Silhouette (S) indices as validation measures. The framework should also allow for additional indices to be included at a later stage. In the instance of Hierarchical clustering, a dendrogram will also be generated. This illustrates how the clusters are formed and merged over time.

6.1.5 Visualisation.

We intend to make use of the VMD software package to visualize the dominant conformations. Once the the MD simulation data has been clustered, an output file with dominant conformations will be used to illustrate which conformations of the molecules are most prevalent.

6.1.6 Clustering Output.

The clustered frames will be saved in a user specified format, in addition to the default format. The default format will include all frames grouped into the clusters generated. This will allow the user to see all conformations and in which clusters they belong to. The user may also specify whether they only want to output the largest n conformations to a .pdb format instead of all clustered data. This will allow the user to easily view the dominant confirmations.

6.2 Algorithm Breakdown

6.2.1 Hierarchical Clustering.

Hierarchical clustering is a type of agglomerative clustering whereby it initially treats each data object as a cluster, then merges these clusters iteratively according to some distance linkage criteria. This process continues until one homogeneous cluster has been formed. Importantly hierarchical clustering produces a dendrogram which illustrates how the cluster formations merge over iterations. This can be used to investigate similar conformations and determine levels of similarity as we can look at different iterations where clusters are merged.

There are four types of linkage criteria we will investigate in our research. These linkage criteria outline what is defined as the minimum distance between two clusters. These include - Ward, maximum, single and average linkage. Single-linkage merges two clusters based on the shortest distance between a pair of points within the two clusters. Searching for the minimum distance between a pair of elements withing two sets before merging. Maximum-linkage focuses on searching for the maximum distance between a pair of elements from two sets to define the distance between two clusters. Average-linkage calculates the minimum distance between two clusters as the average distance between each point in one cluster to another. Ward's method aims to minimise the total variance between clusters merging. At each iteration it determines which two clusters once merged will have the smallest variance increase. This continues until we have once homogeneous cluster. As we can see the linkage criteria is what defines minimum distance between two clusters and how this is calculated.

All implementations of these algorithms are available in the python SciPy library. We will make use of this existing code and modify it to allow for clustering of MD simulation data. To validate the changes and ensure the algorithms integrity we will make use of unit tests and cluster artificial datasets with known cluster formations. Once we have ensured the algorithms have been correctly implemented, we will cluster and analyse MD simulation data of highly flexible molecules.

6.2.2 *Quality Threshold Clustering.*

The Quality Threshold Algorithm requires two user specified parameters, similarity threshold and the minimum amount of elements in each cluster. A candidate cluster is formed from the first object in the data set, then iteratively every data objects similarity measure is compared and those below the threshold are added to the cluster. Only data objects with similarity measure differences below the threshold for all current data objects in the cluster will be added. Once this step is complete we proceed to do this for all data objects in the data set. Consequently, each data object forms a cluster. All data objects are candidates for all clusters at this stage.

Once this process has been completed, we take the largest cluster and remove it from the dataset. Iteratively removing the next largest cluster from the pool until we reach the minimum user defined size of a cluster. At this point there may be data objects that do not belong to a cluster. It is important to note that we may set the minimum size of the cluster to one. This will allow all data objects to be assigned to a specific cluster, albeit we will have many more cluster formations - particularly those of a smaller size.

We will make use of a Python implementation of the QT algorithm from González-Alemán et al. [11]. This has already been modified for MD data. We would therefore only need to integrate the algorithm with the current framework. We will then perform validation tests with artificial datasets to determine whether the algorithm implementation and framework function as expected. Once this has been completed, we will cluster the MD data of highly flexible molecules with different quality thresholds in order to determine the algorithms effectiveness. In addition we will use various CVIs to determine the performance of the algorithm with the datasets.

6.2.3 *HDBSCAN.*

HDBSCAN is a density-based clustering algorithm which begins by creating a network containing all of the data points [17]. The weight of the edge between any two points is a value which describes how close together the points are, while also taking into account whether or not the point is in a dense or sparse region of the data. Once the network has been constructed, the edges are removed in descending weight order, until removing another edge would separate two regions of the data. At this point, a hierarchical single-linkage algorithm is used to build a dendrogram from the network. The final clustering is then extracted based on the minimum cluster size and a stability metric.

HDBSCAN does not require an assumption to be made regarding the number of clusters in the data. Although it has two parameters, minimum cluster size and minimum neighbourhood size, there are suitable default values which make it possible to use it effectively non-parametrically. The fact that HDBSCAN can handle noise in the data and can produce clusters of varying shapes and sizes makes it an attractive option for trajectories of flexible molecules where the underlying structure of the data is unknown. Additionally, HDBSCAN has been shown to successfully identify distinct conformations, even in trajectories of disordered proteins [17], and hence may be an ideal candidate for clustering flexible molecules.

A Python library, made available by Melvin et al. [17] will be used to implement HDBSCAN and iMWK-Means. The library is freely available and makes use of the Python implementation of HDBSCAN by McInnes et al. [16]. As this implementation was created for use clustering MD trajectories, it will not need to be adapted in this respect, however, a level of adaptation will be required to integrate algorithm into the framework. The algorithm will also need be validated with unit tests and basic artificial datasets with obvious inherent clusters to confirm that the clustering is working as expected. The actual testing of HDBSCAN will makes use of both default and non-default values for the parameters, and the MD trajectories used will include those of flexible carbohydrates.

6.2.4 *iMWK-Means.*

iMWK-Means is a variant of k-means which differs in that it does not require the user to select a desired cluster count, and, in doing so, make an assumption about the structure of the data [17]. It has a single parameter which can be derived from the chosen distance metric. iMWK-Means begins by overestimating the numbers of clusters and then iteratively performing rounds of standard k-means clustering and explicit rescaling. The rescaling involves adding weights to the data points based on how close they are to the centroids of their clusters. The result of this is that dense areas of the data are clustered together, while sparse areas and outliers receive low weights, effectively classifying them as noise. Once the rescaling step results in no changes to the weights, one more round of k-means is performed, which outputs the final clustering.

The clusters produced by iMWK-Means can vary in shape and size and are resistant to noise in the data. This algorithm and HDBSCAN compliment each other in the fact that iMWK-Means performs better with stable systems but is more sensitive to details, while HDBSCAN detects larger conformational changes [17]. Using the algorithms either in combination or succession may produce better clustering than either of the them alone.

The Python library from Melvin et al. [17] also includes an implementation of iMWK-Means. Similarly to HDBSCAN, the implementation is intended for clustering MD trajectories but will need to be adapted to our framework. The initial validation will also be similar to that of HDBSCAN and will involve unit tests and artificial datasets. iMWK-Means does not have parameters for which a variety of values need to be tested, so the testing will focus on the results of clustering flexible molecules with iMWK-Means alone, as well as in combination with HDBSCAN. This will also require further adaptation of the existing library.

6.2.5 *AMOSA.*

AMOSA, being a multi-objective algorithm, looks to optimise multiple objective functions simultaneously. By doing so it can produce a set of results with each solution catering for different properties of the data, rather than just outputting one. Throughout the process, the concept of dominance is used. For a solution to dominate another, it must yield better values for all of its objective functions than the other solution. An archive of these best-performing non-dominated solutions is kept, which eventually become the Pareto-optimal set outputted.

Being a simulated annealing algorithm, the main process remains largely the same. A single solution from the archive is chosen to be perturbed, i.e., altered slightly to produce a neighbouring solution. This new solution's optimisation of the objective functions is compared to the original and accepted into the archive depending on its performance and the current temperature of the system. This process is repeated until the system temperature has reached the minimum value set by the user.

The algorithm will be implemented using existing source code provided by one of the co-authors of the paper which introduced the AMOSA method. The algorithm is written in C and so it will require a wrapper when interacting with the framework as the latter will be implemented in Python.

As adaptations of AMOSA will be necessary since the algorithm was not made specifically for MD clustering, testing to verify correctness of the changes will be important. Validation in the form of unit and integration testing will allow us to see whether the changes affect the output of the algorithm in anyway, and if it works with the framework.

6.2.6 *SOM.*

SOM uses neurons that are scattered around the vector space of the input data. Each neuron has an associated weight with the same number of dimensions as the input. Input data is fed into the SOM one at a time where the closest neuron is designated as the best matching unit (BMU). The BMU and the neurons in its neighbourhood are shifted towards the input while also updating their weight values. Over time the neurons start converging on clusters as less neighbours are updated and to a lesser degree. The result is often a two-dimensional U-matrix that depicts the different clusters using colour intensity to illustrate cluster cohesion and separation.

A SOM implementation proposed by Bouvier et al. [5] has been made publicly available on Github. This implementation will be used and adapted upon. The package is written in Python 2 which is not backwards compatible with Python 3 and, therefore, the proposed framework. This should not be a problem as the package can be ported to Python 3 automatically using a tool known as 2to3.

The conversion from Python 2 to 3 itself will justify testing to ensure that no functionality has been altered. Like AMOSA, the SOM will also have to be refactored with the framework in mind, leading to unit and integration testing as well.

6.3 Validity Testing and Evaluation

To validate our algorithms we will make use of UCI data sets and MD trajectories with known cluster information. Initially artificial data sets will also be used to validate the algorithms. Some of these datasets will be used specifically for certain algorithms, and these are outlined below. Validity indices, including the Calinski Harabasz , Davies-Bouldin and Silhouette (S) indices, will also be used for testing and will be specifically relevant when no known cluster information is available. They will also determine the effectiveness of a clustering algorithm and will allow for comparisons between different algorithms.

Shao et al. [19] make use of artificial data sets with their implementations of hierarchical clustering. The same data sets can used our implementations of hierarchical clustering in order to evaluate and compare outputs. We can use common UCI data sets with known characteristics to verify our implementation of QT clustering. As QT clustering has already been implemented with MD trajectories we can take previous data and compare it to our implementation to verify results. Once we have validated our algorithms implementations, we can then evaluate their effectiveness with highly flexible molecules.

The AMOSA package includes the artificial testing data used in the related paper and so comparisons can be made to the results found by the co-authors. Common UCI data sets such as Wine and Iris can also be used to test the algorithm as their properties and characteristics are known beforehand and can be used as the ground truth. Testing may also involve the use of MD trajectories that have already been clustered in the past so we can ensure that AMOSA is able to replicate the same results. SOM will also require this form of testing especially since its implementation has been made specifically with trajectory clustering in mind.

7 ETHICAL, PROFESSIONAL, AND LEGAL ISSUES

The ethical concerns about the project are minimal as we will not be conducting any research that involves personal or identifiable information. All datasets and simulation data sources will be acknowledged, and any relevant research that impacts our project will be cited accordingly. Where we make use of third party software and packages, the authors will be credited as well. As the packages used will undoubtedly require adaptation to suit our needs, explicit permission from respective authors will be obtained prior to doing so if there is any legal ambiguity. All resulting software and research will be made publicly available and open source, and we will follow the University of Cape Town's policy when publishing any work.

We do not anticipate any ethical, professional or legal issues associated with this research.

8 ANTICIPATED OUTCOMES

We expect a framework to be produced which offers a versatile range of functions for clustering MD trajectories. The framework should be able to cluster said trajectories using any of a variety of algorithms. In the future, this algorithms may be one not already included in the framework and so extensibility and modularity must be adhered to. The user should also be able to customise the properties of the clustering process to some degree, such as selecting which frames to cluster and selecting what outputs to produce. These outputs will include files containing the partitioned conformations to visualisation of the clusters themselves.

We anticipate the framework to also be able to validate the clustering algorithms by comparing their results to ground truths by using data with known properties and characteristics. We expect this work to greatly benefit workflows that routinely involve the process of clustering and analysing trajectories.

We expect to be able to answer the research question by identifying an algorithm or set of algorithms which are most effective for clustering MD trajectories of flexible molecules.

Nicholas expects hierarchical clustering to produce clusters of large variety, however Ward's method will likely outperform the other variants. This is due to its ability to take all data objects in a cluster into account when determining whether to merge with another cluster. We suspect that the other linkage functions will produce arbitrary results. The dendrogram will allow for important and rigorous analysis of any underlying patterns with each linkage function. QT clustering is expected to produce effective clustering results with highly flexible molecules. As QT clustering has previously been implemented with MD data, we expect similar results even with highly flexible molecules that have multiple different confrontations.

Wen Kang expects to find objective functions for AMOSA whose performance is dependant on the trajectory at hand. Therefore, it is likely that these functions will be categorised into different sets with each catering for specific molecular conformations. The comparison between SOM and AMOSA is difficult to predict beforehand given that both algorithms have different strengths and weaknesses that will likely make each algorithm's usage situational depending on the trajectory at hand.

Robyn believe that both algorithms are likely to produce usable clustering in general, although it is expected that the results will vary for each trajectory that is tested. It is expected that HDBSCAN will outperform iMWK-Means when clustering trajectories of flexible molecules, as this is what has been seen in the past [17]. It would be significant to confirm this, as it would mean that a similar result had been identified when clustering both disordered proteins and carbohydrates with HDBSCAN. Additionally, it is expected that feeding the stable clusters produced by HDBSCAN into iMWK-Means will reveal higher resolution details, as this is the nature of iMWK-Means.

8.1 Success Factors

Success factors should determine whether our implementation has succeeded. As such, the implementation of the framework should allow trajectories to be read in, preprocessed, clustered, validated, and visualised. Outputs selected by the user should be produced, and the framework should be extensible regarding the choice of the algorithms used to cluster.

The framework should include a the previously clustering algorithms that must be implemented correctly and should, therefore, undergo validation tests to ensure this. As such, through the use of CVIs, algorithms should be compared to determine which may be most effective in clustering highly flexible molecules.

The clusters produced by the algorithms should allow the user to make conclusions about the dominant conformations. These clusters should be manageable in size and not overwhelming.

The overarching research question as well as the individual ones are expected to be answered by the end of the project.

9 PROJECT PLAN

9.1 Risks

We have identified numerous risks to this project (see Appendix A). Each risk has an associated probability of occurring and impact factor that we take into account. We outline possible consequences should the risk occur while also outlining strategies to monitor, mitigate and manage these risks.

9.2 Deliverables

Date Due	Deliverable	Description
12th May	Literature Review	Review current literature to establish understanding of the topic.
4th June	Project Proposal	Propose project and outline objectives of the research.
29th June	Revised Proposal	Adjust proposal based on feedback from supervisor and second reader.
10th August	Feasibility Demonstration	Demonstrate framework and basic functionality of the software.
11th September	Final Draft Submission	Complete draft of research paper for supervisor feedback.
21st September	Project Paper Final Submission	Final submission of research paper including related work, findings, analysis and conclusions.
25th September	Project Code Final Submission	Submit final framework with algorithm implementations.
5th - 9th October	Final Project Demonstration	Present project findings and demonstrate framework.
12th October	Poster Due	Compile a poster of the research.
19th October	Web Page Due	Prepare project for online repository.

9.3 Milestones

Date Due	Deliverable
12th May	Literature Review
22nd May	Project Proposal Scaffold
4th June	Project Proposal
29th June	Revised Proposal
10th July	Project Scaffold completed
17th July	Background Section completed
9th August	Framework complete for feasibility demonstration
10th August	Feasibility Demonstration
24th August	Algorithm implementation completed
25th August	Validating algorithm implementations
29th August	First implementation with Carbohydrate simulation data
7th September	Final implementation and testing
10th September	Complete analysis and write up of research paper
11th September	Final Draft Submission
21st September	Project Paper Final Submission
25th September	Project Code Final Submission
5th - 9th October	Final Project Demonstration
12th October	Webpage Due
19th October	Poster Due

9.4 Timeline (Gantt Chart)

The project has already commenced and will run until the 21st of September 2020 with a final paper submission. A few additional deliverables are required after this date, however, these do not impact the core research of the project. A Gantt chart (see Appendix B) outlines the major tasks and deliverables for this project. Each task has an associated duration and deadline. We aim to remain on schedule for the duration of the project.

9.5 Resources Required

9.5.1 Software resources.

(1) VMD
(2) SciPy library
(3) Matplotlib
(4) Pre-existing algorithm implementations

9.5.2 Hardware resources.

At this stage we anticipate that personal computers will be sufficient for our project. Should further computing power be required, we will request access to the University of Cape Town HPC cluster.

9.5.3 Data resources.

We intend to make use of a variety of artificial and real datasets. Artificial datasets are used to determine whether the algorithm has been implemented correctly and test the validity of the algorithm. The datasets we aim to cluster will be Molecular Dynamic simulations of carbohydrate molecules. We aim to determine which algorithms are best suited for clustering highly flexible molecules, therefore we will make use of a variety of different carbohydrate molecules to determine an effective algorithm. In addition there is the possibility of clustering artificial MD simulation data in order to validate the algorithms.

9.6 Work Allocation

As mentioned previously each member will focus on implementing their selected algorithms. In addition each member will contribute to develop the framework. There will be various stages of collaboration throughout the project to ensure we produce a single comprehensive framework while still allowing for each member to observe results for their respective research questions. We outline the individual work allocation below.

Nicholas will implement the hierarchical and Quality Threshold algorithms while also working on the visualisation section of the framework. This will allow the user to visualize dominate confirmations with VMD.

Robyn will implement the HDBSCAN and iMWK-Means algorithms, and will focus on the sections of the framework which handle validation and output options.

Wen Kang will adapt the AMOSA and SOM algorithms while also focusing on the input aspect of the framework, ensuring that that user-defined parameters and trajectory options are adhered to.

REFERENCES

[1] Tigran M. Abramyan, James A. Snyder, Aby A. Thyparambil, Steven J. Stuart, and Robert A. Latour. 2016. Cluster analysis of molecular simulation trajectories for systems where both conformation and orientation of the sampled states are important. *Journal of Computational Chemistry* 37, 21 (2016), 1973–1982. https://doi.org/10.1002/jcc.24416

[2] Olatz Arbelaitz, Ibai Gurrutxaga, Javier Muguerza, Jesús M. Pérez, and Iñigo Perona. 2013. An extensive comparative study of cluster validity indices. *Pattern Recognition* 46, 1 (2013), 243–256. https://doi.org/10.1016/j.patcog.2012.07.021

[3] S Bandyopadhyay, S Saha, U Maulik, and K Deb. 2008. A Simulated Annealing-Based Multiobjective Optimization Algorithm: AMOSA. *IEEE Transactions on Evolutionary Computation* 12, 3 (2008), 269–283.

[4] Guillaume Bouvier, Nathan Desdouits, Mathias Ferber, Arnaud Blondel, and Michael Nilges. 2014. An automatic tool to analyze and cluster macromolecular conformations based on self-organizing maps. *Bioinformatics* 31, 9 (12 2014), 1490–1492. https://doi.org/10.1093/bioinformatics/btu849 arXiv:https://academic.oup.com/bioinformatics/article-pdf/31/9/1490/17085891/btu849.pdf

[5] Guillaume Bouvier, Nathan Desdouits, Mathias Ferber, Arnaud Blondel, and Michael Nilges. 2015. An automatic tool to analyze and cluster macromolecular conformations based on self-organizing maps. *Bioinformatics* 31, 9 (2015), 1490–1492.

[6] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. 2013. Density-Based Clustering Based on Hierarchical Density Estimates. In *Advances in Knowledge Discovery and Data Mining*, Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 160–172.

[7] Ricardo J. G. B Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. 2015. Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 10, 1 (2015), 1–51.

[8] David L. Davies and Donald W. Bouldin. 1979. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1, 2 (1979), 224–227. https://doi.org/10.1109/TPAMI.1979.4766909

[9] Renato Cordeiro de Amorim and Christian Hennig. 2015. Recovering the number of clusters in data sets with noise features using feature rescaling factors. *Information Sciences* 324 (2015), 126–145. https://doi.org/10.1016/j.ins.2015.06.039

[10] K Deb, A Pratap, S Agarwal, and T Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.

[11] Roy González-Alemán, David Hernández-Castillo, Julio Caballero, and Luis A Montero-Cabrera. 2019. Quality Threshold Clustering of Molecular Dynamics: A Word of Caution. *Journal of chemical information and modeling* 60, 2 (2019), 467–472.

[12] M. Halkidi and M. Vazirgiannis. 2001. Clustering validity assessment: finding the optimal partitioning of a data set. (2001), 187–194. https://doi.org/10.1109/ICDM.2001.989517

[13] L J Heyer, S Kruglyak, and S Yooseph. 1999. Exploring expression data: identification and analysis of coexpressed genes. *Genome research* 9, 11 (1999), 1106–1115.

[14] Pablo Jaskowiak, Davoud Moulavi, Antonio Furtado, Ricardo Campello, Arthur Zimek, and Jörg Sander. 2016. On strategies for building effective ensembles of relative clustering validity criteria. *Knowledge and Information Systems* 47, 2 (2016), 329–354. https://doi.org/10.1007/s10115-015-0851-6

[15] Joshua D. Knowles and David W. Corne. 2000. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation* 8, 2 (2000).

[16] Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software* 2, 11 (mar 2017). https://doi.org/10.21105/joss.00205

[17] Ryan Melvin, Ryan Godwin, Jiajie Xiao, William Thompson, Kenneth Berenhaut, and Freddie Salsbury. 2016. Uncovering Large-Scale Conformational Change in Molecular Dynamics without Prior Knowledge. *Journal of Chemical Theory and Computation* 12, 12 (2016), 6130–6146. https://doi.org/10.1021/acs.jctc.6b00757

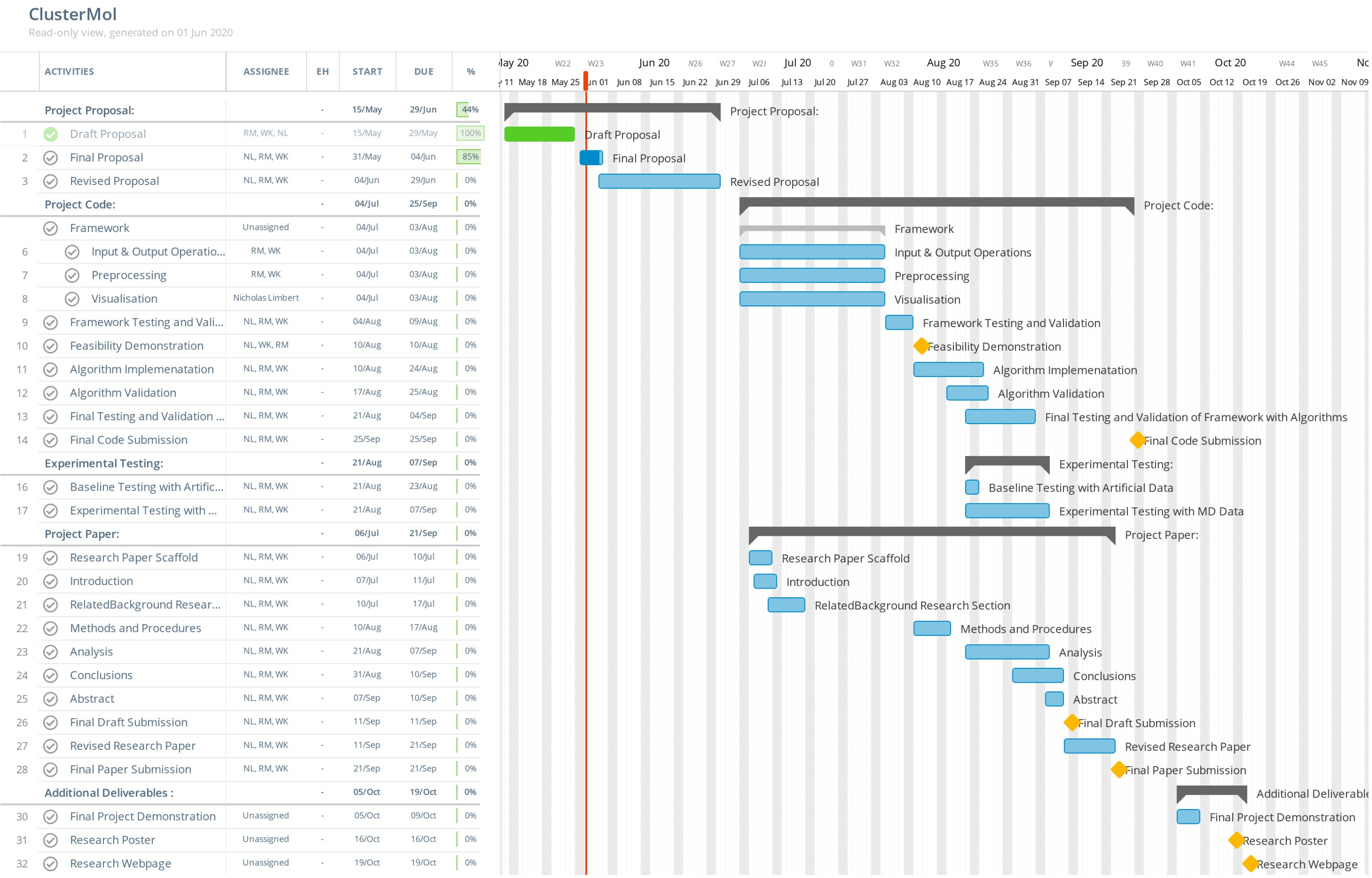
[18] Jy Shao, Sw Tanner, N. Thompson, and Te Cheatham. 2007. Clustering molecular dynamics trajectories: 1. Characterizing the performance of different clustering algorithms. *Journal Of Chemical Theory And Computation* 3, 6 (2007), 2312–2334. https://doi.org/10.1021/ct700119m

[19] Jianyin Shao, Stephen W Tanner, Nephi Thompson, and Thomas E Cheatham. 2007. Clustering Molecular Dynamics Trajectories: 1. Characterizing the Performance of Different Clustering Algorithms. *Journal of chemical theory and computation* 3, 6 (2007), 2312–2334.

A RISK MATRIX

Risk	Probability	Impact	Consequence	Monitoring	Mitigation	Management
Team member drops out of project.	2	8	Although the research questions are individual work, remaining team members have increased workload regarding the framework that they might be unable to handle.	Periodically ensure that all team members are happy to pursue the goals of the project.	Ensure that as many feasible changes are made where a team member is unhappy about a certain aspect of the project.	Reduce the scope f the framework to ensure the project is completed on time.
Framework scope creep.	3	7	Core features may not be implemented fully if additional features are given priority.	Have weekly meetings discussing the current direction of the framework.	Set firm objectives that shouldn't be deviated from.	Immediately stop working on additional features and return to core functions.
Core functions cannot be delivered on time.	3	10	Unable to submit a project report on time, leading to reduced marks or failure.	Have weekly checkups with all team members and with supervisor.	Ensure that all team members are working as required, bringing up any problems where necessary.	Consult supervisor and potentially find an alternative course of action.
Lack of communication within the team as well or with the supervisor.	1	7	Implementations might be done incorrectly or bad assumptions may be made which can lead to problems with the framework and answering of the objective questions.	Attend weekly meetings and discuss what everyone is currently working on and what barriers are being faced.	Ensure everyone is kept up-to-date with all agreements and plans.	Discuss more stringent meetings and alternative communication methods.
Team member(s) falling seriously ill due to COVID-19.	2	9	Large amounts of development time could be lost as recovery may take weeks.	Monitor one's own health and ensure stakeholders are aware of any problems.	Continue practicing social distancing and sanitise often.	Notify team and the supervisor to discuss alternative solutions should time become an issue.
Chosen algorithms to complex to implement.	5	10	Unable to perform experimental tasks and compare effectiveness on highly flexible molecules.	Test algorithms with artificial data to ensure implementations are on schedule.	Ensure we have sufficient knowledge of algorithms, and requirements.	Notify team and the supervisor to discuss alternative solutions. These include the possibility of implementing a different algorithm.
Computing power insufficient for clustering analysis	6	3	Unable to cluster data timelessly and produce results for analysis.	Ensure we understand complexity of algorithms and computing requirements.	Constantly evaluate performance.	Inform supervisor and request access to HPC cluster.
Loss of interest in project	1	4	Poor work and reduced project deliverable output.	Constantly evaluate supervisor feedback and keep on schedule.	Constantly evaluate peers performance and keep up to date with their work.	Reevaluate scope and produce minimal viable project.

B GANTT CHART



ClusterMol

Read-only view, generated on 01 Jun 2020

	ACTIVITIES	ASSIGNEE	EH	START	DUE	%
	Project Proposal:		-	15/May	29/Jun	44%
1	Draft Proposal	RM, WK, NL	-	15/May	29/May	100%
2	Final Proposal	NL, RM, WK	-	31/May	04/Jun	85%
3	Revised Proposal	NL, RM, WK	-	04/Jun	29/Jun	0%
	Project Code:		-	04/Jul	25/Sep	0%
	Framework	Unassigned	-	04/Jul	03/Aug	0%
6	Input & Output Operatio...	RM, WK	-	04/Jul	03/Aug	0%
7	Preprocessing	RM, WK	-	04/Jul	03/Aug	0%
8	Visualisation	Nicholas Limbert	-	04/Jul	03/Aug	0%
9	Framework Testing and Vali...	NL, RM, WK	-	04/Aug	09/Aug	0%
10	Feasibility Demonstration	NL, WK, RM	-	10/Aug	10/Aug	0%
11	Algorithm Implemenatation	NL, RM, WK	-	10/Aug	24/Aug	0%
12	Algorithm Validation	NL, RM, WK	-	17/Aug	25/Aug	0%
13	Final Testing and Validation ...	NL, RM, WK	-	21/Aug	04/Sep	0%
14	Final Code Submission	NL, RM, WK	-	25/Sep	25/Sep	0%
	Experimental Testing:		-	21/Aug	07/Sep	0%
16	Baseline Testing with Artific...	NL, RM, WK	-	21/Aug	23/Aug	0%
17	Experimental Testing with ...	NL, RM, WK	-	21/Aug	07/Sep	0%
	Project Paper:		-	06/Jul	21/Sep	0%
19	Research Paper Scaffold	NL, RM, WK	-	06/Jul	10/Jul	0%
20	Introduction	NL, RM, WK	-	07/Jul	11/Jul	0%
21	RelatedBackground Resear...	NL, RM, WK	-	10/Jul	17/Jul	0%
22	Methods and Procedures	NL, RM, WK	-	10/Aug	17/Aug	0%
23	Analysis	NL, RM, WK	-	21/Aug	07/Sep	0%
24	Conclusions	NL, RM, WK	-	31/Aug	10/Sep	0%
25	Abstract	NL, RM, WK	-	07/Sep	10/Sep	0%
26	Final Draft Submission	NL, RM, WK	-	11/Sep	11/Sep	0%
27	Revised Research Paper	NL, RM, WK	-	11/Sep	21/Sep	0%
28	Final Paper Submission	NL, RM, WK	-	21/Sep	21/Sep	0%
	Additional Deliverables :		-	05/Oct	19/Oct	0%
30	Final Project Demonstration	Unassigned	-	05/Oct	09/Oct	0%
31	Research Poster	Unassigned	-	16/Oct	16/Oct	0%
32	Research Webpage	Unassigned	-	19/Oct	19/Oct	0%