Fuzzy Genome Sequence Assembly for Single and Environmental Genomes

Sara Nasser, Adrienne Breland, Frederick C. Harris Jr., Monica Nicolescu, and Gregory L. Vert

Department of Computer Science & Engineering, University of Nevada Reno, Reno NV 89557, USA {sara,brelanda,Fred.Harris,monica,gvert}@cse.unr.edu

Summary. Traditional methods obtain a microorganism's DNA by culturing it individually. Recent advances in genomics have lead to the procurement of DNA of more than one organism from its natural habitat. Indeed, natural microbial communities are often very complex with tens and hundreds of species. Assembling these genomes is a crucial step irrespective of the method of obtaining the DNA. This chapter presents fuzzy methods for multiple genome sequence assembly of cultured genomes (single organism) and environmental genomes (multiple organisms).

An optimal alignment of DNA genome fragments is based on several factors, such as the quality of bases and the length of overlap. Factors such as quality indicate if the data is high quality or an experimental error. We propose a sequence assembly solution based on fuzzy logic, which allows for tolerance of inexactness or errors in fragment matching and that can be used for improved assembly.

We propose fuzzy classification using modified fuzzy weighted averages to classify fragments belonging to different organisms within an environmental genome population. Our proposed approach uses DNA-based signatures such as GC content and nucleotide frequencies as features for the classification. This divide-and-conquer strategy also improves performance on larger datasets. We evaluate our method on artificially created environmental genomes to test various combinations of organisms and on an environmental genome.

1 Introduction

DNA is the building block of all life on this planet, from single cell microscopic bacteria to more advanced creatures such as humans. Twenty years after the DNA code was cracked, Frederic Sanger, a Nobel Laureate, invented the chain termination method of DNA sequencing, also known as the Dideoxy termination method or the Sanger method [39]. His research paved the way to a technique to obtain DNA sequences and to the first genome sequence assembly, Bacteriophage ϕX174 [38]. In 1990 the Human Genome Project was announced, which sought to sequence the billions of nucleotides present in human DNA and was completed in 2003, two years before its projected date.

In 1993 The Institute for Genome Research (TIGR) decided to use the TIGR EST algorithm which is based on whole-genome shotgun sequencing method to assemble a microbial genome. Thus in 1995 at TIGR, *Haemophilus influenzae* was sequenced, becoming the first genome to be sequenced entirely [11]. Shotgun sequencing was first demonstrated to close a genome in this paper. The assembly of *H. influenzae* proved the potential of shotgun sequencing and thus lead to subsequent projects that would be based on shotgun sequencing. There were two major advancements in technology that lead the to complete sequencing of the Human Genome and the *H. influenzae*: shotgun sequencing and the use of computational techniques. This brief history gives insight into the advancements that were made in sequencing using computational techniques. For more history on the evolution of DNA sequencing and timelines of genome sequencing projects, refer to [7] and [31].

A DNA strand consists of four nucleotides: Adenine(A), Cytosine(C), Guanine(G) and Thymine(T). Genome sequencing is figuring out the order of DNA nucleotides, or bases, in a genome that make up an organism's DNA. Genome sequences are large in size and can range from several million base pairs in prokaryotes to billions of base pairs in eukaryotes. For example, Wolbachia genome, a bacteria has 126 million base pairs (Mb), Arabidopsis thaliana, a plant has 120Mb, and the human genome is 3.2 billion base pairs. The whole genome cannot be sequenced all at once because available methods of DNA sequencing can handle only short stretches of DNA at a time. Although genomes vary in size from millions of nucleotides in bacteria to billions of nucleotides in humans, the chemical reactions researchers use to decode the DNA base pairs are accurate for shorted lengths [32]. Genomes are cut at random positions then cloned to obtain the smaller fragments, also known as shotgun sequences. Obtaining shotgun sequences has allowed sequencing projects to proceed at a much faster rate, thus expanding the scope of the realistic sequencing venture [31]. Sequencing DNA using the shotgun method led to the completion of several organism genomes, including human, mouse, fruit fly and several microbes, such as *Wolbachia* genome and *H. influenzae*.

Microorganisms live in communities, and their structure and behavior is influenced by their habitat. Most microorganisms genomes are known from pure cultures of organisms isolated from the environment, be it a natural organism-associated (i.e, human) or artificial system. Cultivation-based approaches miss majority of the diversity that exists however, such that development of cultivation-free methods has been implied. In the past, microbial DNA was sequenced by culturing microorganisms in a controlled environment. Cultivating these organisms did not reveal enough information about these communities of organisms. Invitro cultivation methods allow the extraction of DNA from only a limited selection of microbial species that can grow in artificial environments. These methods do little to characterize the properties of globally distributed microbes, because the vast majority of them have not been cultured.

New techniques in genomic sciences have emerged that allow an organism to be studied in its natural habitat as part of a community. Research has broadened from studying single species to understanding microbial systems and their adaptations to natural environments. These techniques have been achieved by developing methods that can sequence mixed DNA directly from environmental samples [2, 36].

Whole-genome shotgun sequencing of environmental DNA gained attention as a powerful method for revealing genomic sequences from various organisms in natural environments [2, 41]. An organism's DNA was not only sliced into small fragments but also mixed with other organisms' DNA fragments, thus creating a huge population of fragments, initial efforts were with long fragments ranging from 40Kb - 150Kb in fosmid or BAC libraries. Even though DNA fragments from diverse populations can be gathered together at the same time, they need to be assembled in order for us to make meaningful conclusions.

There are several approaches that are designed for examining a single organism, but there is a need for tools that are specific for community-level analysis. In this chapter, we propose methods of sequence classification and assembly for a metagenomic population. Sequence classification is a process of grouping genome fragments into classes based on their similarities. The proposed method aims to use an approximate method based on fuzzy logic to classify genome fragments into groups and then perform assembly.

The rest of the chapter is structured as follows: Sect. 2 presents background information on computational biology and DNA sequence assembly, including a survey of the related literature. Sect. 3 presents the fuzzy solution for sequence assembly. Sect. 4 presents taxonomical classification methods for metagenome fragments. Improvements achieved due to signatures and a new technique of fuzzy classification, in addition to results attained, are included in Sect. 3 and Sect. 4. Conclusions and a look into future directions are presented in Sect. 5.

2 Background

2.1 Genome Sequence Assembly

Several concepts and terms from genomic sciences that are used this in chapter are informally defined below. There are several books on computational biology that provide detailed explanations of the terms listed below [1, 50].

Definition 1. *Base Pair: Two nucleotides on a paired double-helix-structured DNA strand. These two nucleotides are complements of each other.*

Definition 2. *DNA Fragment/Read: A section of the genome sequence of nucleotides that forms a DNA strand.*

Definition 3. *Contig (Contiguous Sequence): A consensus sequence created by overlapping two or more sub-sequences or fragments.*

Definition 4. *Nucleotide Frequencies: The measure of occurrences of nucleotide pairs of a specified length.*

Definition 5. *Metagenome: Genome sequences containing an unspecified number of microbial organisms directly obtained from the natural habitat.*

The problem of sequence assembly is acquiring data and assembling the DNA fragments or sequences into an entire genome sequence. Available chemical technologies for sequences produce short fragments of DNA sequences (40 Kbp -1000 Kbp) depending on the technology. Sequencing machines cannot read entire DNA, and can only work on small stretches at a time. There are two important aspects to understanding the problems that arise in genome assembly: the genome is cut into smaller portions, and fragments or sequences are cut at random positions. To obtain the original sequence these fragments need to be combined by determining overlaps between fragments. Thus, portions of the fragments need to appear more than once. Multiple copies of original sequences are made to ensure that the entire sequence is covered. This process is generally referred to as coverage of nX, where n is the number of copies and X is the sequence. Coverage of 8X or 10X is widely accepted and it has been shown it is sufficient to reconstruct the entire sequence. Thus for a genome sequence of length 4(million)MB, if the sequence fragments of length around 500 bp are generated we need 80,000 sequences.

Following the sequencing process, an assembler pieces together the many overlapping bases and reconstructs the original sequence [32]. The process explained above is known as the whole-genome shotgun method. There are three main steps involved in the assembly of sequences. The first step, Sequencing, breaks the genomic DNA into fragments by sonication, a technique which uses high-frequency sound waves to make random cuts in DNA molecules [4]. In the assembly phase the sequences are combined to form contiguous sequences. The final phase is finishing, in this phase contigs are joined by closing physical gaps. Closing is a time consuming process, which can be improved by using more than one clone libraries. Clone libraries are prepared using different vectors. As different vectors clone sequences differently, using more than one vector can help improve coverage. Fragments that could not be cloned by one vector could be cloned by the other. Thus gaps could be reduced as overall coverage increases when sequences are generated using different vectors.

Sequencing of an organism's DNA is a labor-intensive task, made possible by recent advances in automated DNA sequencing technology.Even though automated DNA sequencing technology made it possible to sequence genomes, several other problems exist. The sequence read from a machine is not always 100% correct; it may contain experimental errors. The process of acquisition of genome sequence data may lead to the insertion of certain discrepancies in the sequences, known as base-calling errors. The actual DNA sequence is read

as a frequency signal, which is converted to represent the character sequence representing the four nucleotides as A, C, G and T. PHRED is a popular tool for reading signals and assigning quality scores [10]. In this scoring technique each nucleotide is given a score based on the strength of the base, a high score implying a higher probability of the base being correct. Low scores are assigned to bases that have less probabilty of being true. Sequences at the ends tend to have weak signals that make it difficult to identify them. Thus the base is assigned to the closest match and marked as a low quality base.

Base-calling errors create additional problems during assembly. These differences are categorized into three groups: insertions, deletions (indels) and replacements. Another well known problem with the sequences is that they contain repetitive sections also known as repeats. All the parameters mentioned above make assembly an approximation problem.

The most popular approach to DNA fragment assembly has been to iteratively find the best overlap between all fragment pairs until an acceptable final layout is determined. If enough fragments are sequenced and their sampling is sufficiently random across the source, the process should determine the source by finding sequence overlaps among the bases of fragments that were sampled from overlapping stretches [22]. In current genome sequencing tasks, the number of fragments is usually numerous, and the degree of computation required increases exponentially. Being essentially an NP-hard problem, many different approaches with varied parameters and matching schemes have been explored to save computation time.

The earliest approach to find solutions using the shotgun sequence approach was to find the shortest common superstring from a set of sequences. Current approaches use pairwise sequence alignment as a method and instead of obtaining the shortest superstring, the longest common substring is used. To obtain the common substrings of two sequences, we are required to consider all possible substrings of the given sequences. The substring with the longest overlap is known as the longest common sequence (LCS). Finding the LCS for all possible sequences is an NP-hard problem. Thus, a brute-force approach is not feasible. Dynamic programming solves problems by combining the solutions to subproblems to reduce the runtime of algorithms containing overlapping subproblems and optimal substructures [9]. Using dynamic programming, we can find a polynomial-time solution for the LCS problem. Therefore, dynamic-programming-based approaches are the most routinely used approaches in sequence assembly and alignment.

Other techniques for finding the LCS include suffix trees and greedy approaches. A suffix tree is a data structure that uses suffix information for fast processing of string problems. A suffix tree can be constructed in linear time using the Ukkonen algorithm [47]. Even though suffix trees are a linear answer to sequence comparison problems, they are not good at storing and handling large datasets. Greedy algorithms are shown to be much faster than traditional dynamic programming in the presence of sequencing errors [54]. Greedy paradigms, applied in popular assemblers such as TIGR [42], Phrap [13], and

CAP3 [15], are relatively easy to implement, but they are inherently local in nature and ignore long-range relationships between reads that could be useful in detecting and resolving repeats [32]. Greedy and hill-climbing approaches generally find a local optimal and thus the global solution could be missed. Additionally, these algorithms work with specific kinds of errors and cannot be generalized; they also become difficult to implement on larger datasets.

Unlike greedy approaches the overlap–layout–consensus mechanism considers all possible solutions before selecting the consensus overlap. An application of graph theory is found in [18] in which fragment reads are represented by nodes and an overlap between two fragments is represented by an edge. Paths are constructed through the graph such that each path forms a contig. The paths are then cleaned by resolving and removing any problems such as intersecting paths, and consensus sequences are constructed following the paths. One of the major problems of this approach to DNA sequence assembly is the extensive computation requirement. Fragment assembly performed with the overlap–layout–consensus approach becomes inefficient with an increasing number of fragments.

Even with the algorithmic improvements, additional reductions to the search space in fragment assembly problems are routinely employed. Preassembly clustering of fragments may be viewed as a more structured form of fragment thinning before alignment comparisons are made. Clustering is a process of grouping objects into like groups based on some measure of similarity. Clustering or classification can be achieved by several techniques such as K-means and artificial neural networks. A divide-and-conquer strategy for sequence assembly based on average mutual information is described in [29].

2.2 Environmental Genomics

Molecular biology has impacted microbiology by shifting the focus away from clonal isolates and toward the estimated 99% of microbial species that cannot currently be cultivated [6, 16, 34]. As an illustration, traditional culture and PCR-based techniques showed a bias of *Firmicutes* and *Bacteroides* as the most abundant microbial groups in the human gastrointestinal (GI) tract. Metagenomic sampling has revealed that *Actinobacteria* and *Archaea* are actually most prolific [21].

Metagenomic data can be ecosystem or organism associated: ecosystem associated metagenome contains DNA of microbes obtained from an environmental sample and an organism associated metagenome contains DNA from organisms. For example, the Sargasso sea project, an ecosystem associated metagenomes contains microbial samples collected through the filtering of sea water. These samples contained large amounts of novel genetic information, including 148 new bacterial phylotypes, 1.2 million new genes, and 782 new rhodopsin-like photoreceptors [48]. A similar metagenomic project giving new insight into naturally existing bacterial systems was the sampling biofilm from of an underground acid mine drainage [46]. Because this sample was from a

system with low complexity, almost all DNA from present species were completely reconstructed, allowing the examination of strain differences and naturally forming lineages. It also enabled access to the full gene complement for at least two species, providing detailed information such as metabolic pathways and heavy metal resistance. Soil samples and the mouse GI tract are some other published metagenomic projects [36, 45].

Closely related organisms can contain remarkable genomic diversity, as was shown for some bacteria [49]. These variations, even though few, can result in different metabolic characteristics. Extracting these variations is one of the key ideas to further processing of the metagenomes. The genomic diversity between metagenome samples is extracted and used as a marker to separate data phylogenetically.

2.3 Phylogenetic Classification Using Signatures

The metagenomic approach of acquiring DNA fragments often lacks suitable phylogenetic marker genes, rendering the identification of clones that are likely to originate from the same genome difficult or impossible [44]. Separating the fragments in a metagenomic sample and reconstructing them is a complex process. Identification of certain features can distinguish one genome from another in some circumstances. Organisms within a metagenome population can belong to different ranks in the taxonomy, for example they could belong to different domains or could be from the same species. For example, the acidmine drainage data consists samples that belong to archeal and bacterial domains. A metagenome population can contain a large number of organisms that could either be very diverse, that is not closely related or it could be constrained to strains or species that are closely related. Thus complexity of a metagenome can also dictate the classification accuracy. Metagenome complexity can be measured with three different parameters: taxonomic relation, evenness, and richness. Visualization these is complex for example there are several ranks within taxonomy. This subsection describes the DNA signatures that can be employed in identification of differences within a metagenome.

DNA signatures are specific patterns that are observed within a DNA strand. These patterns can be observed in specific regions such as coding regions or can be observed throughout a genome. There have been several studies on the patterns found in DNA sequences. Biological sequences contain patterns that can lead to discoveries about the sequences. Two kinds of signatures are important to our discussion: GC content and oligonucleotide frequencies. Oligonucleotide composition within a genome contains bias, making certain patterns appear several times within the genome. These oligonucleotide usage patterns are known to be species-specific [17].

The four nucleotides of a DNA strand (A, C, G, and T) have hydrogen bonds between them. The nucleotide A bonds specifically with T and the nucleotide C bonds with G. AT pairs have two hydrogen bonds and GC pairs have three hydrogen bonds, making the GC bond thermostable. Thus, the GC content

in an organism can sometimes be used to determine certain characteristics about that organism. Organisms are generally biased in the distribution of A, C, G and T. Certain organisms contain higher percentages of GC and are thus known as GC rich, while some other organisms are dominated by AT and are known as AT rich. This fundamental property of organisms can be used in separating one organism from another.

Another signature that has been used frequently for analysis of genome sequences is the oligonucleotide frequencies, which is a measure of the occurrence of words of fixed sizes in the genomic sequence. Oligonucleotides are short sequences of nucleotides generally of length less than 20. Nucleotide frequencies have been extensively used for grouping species or for differentiation of species. Specific details about obtaining nucleotide frequencies will be covered in Sect. 4.

2.4 Fuzzy Logic

The concept of fuzzy logic and approximate reasoning was introduced in 1975 [53]. Fuzzy logic formalizes an intuitive theory based on human approximation, which is by definition imprecise or vague. Fuzzy set theory allows classification of entities into categories by establishing degrees of weak or strong membership. A fuzzy set F is given by

$$
\begin{aligned}
&F = \{\mu_F(x) \mid x \in X, \mu_F(x) \in [0,1]\} \\
&where: \\
&x = \textit{ a given element} \\
&\mu_F(x) = \textit{ fuzzy set membership function} \\
&X = \textit{ the Universe of Discourse}
\end{aligned}
$$

The fuzzy set membership function, $\mu_F(x)$, returns a membership value between 0 and 1(inclusive) that signifies the degree of membership.

Fuzzy logic has been used in several engineering applications. Fuzzy approaches to bioinformatics have been explored to some extent. Even though the application of fuzzy logic is not widely used, it has begun to gain popularity. An application to ontology similarity using fuzzy logic was presented in [52]. Fuzzy logic also been applied to classification problems in computational biology. A modified fuzzy K-means clustering was used to identify overlapping clusters of yeast genes [12]. A model for creating fuzzy set theory for nucleotides was proposed by Sadegh-Zadeh [37]. In this model a fuzzy polynucleotide space is made to measure the degree of difference and similarity between sequences of nucleic acids. Alignment of sequences has different specifications, and thus, alignment tools are not suitable for assembly purposes. Assembly of sequences is influenced by several factors besides the sequence chain. Therefore, there is a need for an approximation method that takes into consideration all the factors for assembling sequences. Specific fuzzy applications for sequence assembly and classification will be covered in Sects. 3 and 4.

3 Fuzzy Genome Sequence Assembly

DNA sequence assembly can be viewed as the process of finishing a puzzle, where the pieces of the puzzle are DNA subsequences. Although a puzzle has pieces that fit together well, the pieces of a DNA puzzle do not fit together precisely; the ends can be ragged and some pieces are missing, thus making it difficult, and sometimes nearly impossible, to complete the puzzle. Hence, we need methods or rules to determine optimally which piece fits with another piece. The problem of sequence assembly is one of obtaining approximate matches through consensus. A consensus sequence is constructed through approximate matches by following an overlap and consensus scheme [30] as illustrated in Fig. 1.

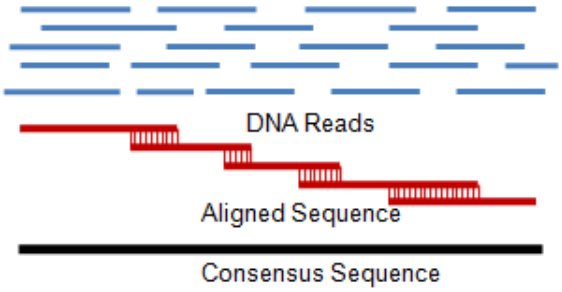

Fig. 1. Whole Genome Sequencing Process of Creating Contigs from Fragment Reads

In current genome sequencing tasks, the number of fragments is usually large, and the degree of computation required increases exponentially. Being essentially an NP-hard problem, many different approaches with varied parameters and matching schemes have been explored which can, among other things, save computation time. Finding the longest common subsequence (LCS) between fragments is the key to the process of sequence assembly.

In this section, an approximate matching scheme based on fuzzy membership functions is presented. Several parameters are considered to create an optimal assembly. Then a divide-and-conquer strategy is presented to speed up the assembly by dividing the sequences into classes. Assembling sequences is accomplished by first grouping the sequences into clusters so that sequences in a cluster have high similarity with one another and sequences between two clusters are less similar.

3.1 Previous Techniques

Dynamic programming has been extensively used to determine the LCS as it reduces the NP-hard problem to a time complexity of $\Theta(n^2)$. The method

is simple and useful in finding the LCS that may have mismatches or gaps. The Smith-Waterman algorithm, an application of dynamic programming to find the LCS for multiple sequence alignment, is one of the most prominent algorithms used in sequence assembly programs. The algorithm gained popularity because it reduces the number of searches required; more details of the algorithm can be found in [40]. Most of the earlier assemblers have crisp bounds and do not adapt to the datasets. For example, a dataset can contain all, or a significant number of, low quality reads. Some of the assemblers clip low quality regions, which will result in, most of the regions getting clipped and thus, not used in assembly. However, if the assembler can adapt itself and allow a new threshold for low quality, this problem can be avoided. Due to its applicability to problems that do not require hard solutions, fuzzy logic has been widely used in various fields to provide flexibility to classical algorithms. Thus, approximate sequence assembly is a good candidate for fuzzy logic. In the next subsection a non-greedy approach is presented, based on approximate sequence matching using fuzzy logic.

3.2 Sequence Assembly

The sequence assembly problem is tackled using two different approaches: the first module performs fuzzy sequence assembly, and the second module performs a fuzzy divide-and-conquer strategy for assembly. The divide-and-conquer strategy uses a fuzzy membership function to divide genome sequences into groups, reducing the number of comparisons and performing meaningful assembly. The fuzzy functions used in this subsection are a modified version of the fuzzy genome sequencing assembler described in [24].

Longest Common Subsequence with Fuzzy Logic

Sequence assembly requires creating contigs from fragment reads. The longest subsequence with fewest insertions or deletions (indels) is ideal. Since an exhaustive search is not applicable for this problem, a time constraint is also placed on the solution. One of the problems with existing techniques for sequence assembly is that they have crisp bounds. The user has to specify the parameters for the program, such as minimum score and minimum match. Almost all existing techniques provide user-defined thresholds; the user generally runs the program several times to obtain optimal results. In such cases it is better to determine empirically the ideal cut-off point or the threshold. For example, assume that a cutoff value for the maximum gap allowed is 30 bases and that there are fairly large numbers of sequences with gaps of 31 and 32. Due to the fact that these techniques allow for crisp matches only, these potentially important sequences would be excluded. Alternatively, we can represent a gap of 30 and lower with a fuzzy confidence value of 1, which is for crisp results. Sequences with gaps that are very close to 30, like 31, can

have a fuzzy value of 0.98. In this case, the user does not have to preprocess the data, change parameters and run the program several times.

The approach starts by acquiring the LCS of given sequence fragments using dynamic programming (details of the fuzzy LCS technique can be found in [24]). The optimal subsequence can be a perfect match, or the user may choose to tolerate indels. These criteria can depend on the user, the source of the data or the quality of the data. There are several factors that determine if two subsequences have an optimal overlap. We propose a method in which we select multiple subsequences and then, based on fuzzy parameters, select the optimal solution. The novelty of our method is that it uses more parameters of the sequence besides the length of overlap, and we believe that these parameters can lead to a better sequence. The sequence satisfying the aggregate overall requirement is selected. The process starts with LCS and selecting all the subsequences that satisfy the minimum length required as given in (1). The threshold is a function of the length of the LCS.

$$length \geq threshold, \quad threshold = f(length(LCS)) \tag{1}$$

In (1) *length* is the size of overlap, and *threshold* determines the minimum length required. The selection of the optimal subsequence is done using fuzzy similarity measures in constant time; therefore, the complexity of the algorithm is same as the complexity of dynamic programming, which is $\Theta(mn)$ for any two subsequences of length m and n. After the LCS is obtained we need to determine the other factors that influence assembly. The following subsection lists the descriptions along with the characteristic functions for each of the parameters.

Fuzzy Similarity Measures

Fuzzy similarity measures and the concept created by this research are an important step in creating a contig from two subsequences or finding an overlap between two sequences. The following subsections describe the fuzzy functions utilized in our approach for assembly.

Length of Overlap

The first similarity measure is the length of the match or length of overlap μ_{lo}, which includes indels and replacements. A higher overlap is better because it generates a longer contig; thus, this function aims at maximizing overlap. The membership function for this measure is defined as:

$$\mu_{lo}(s1, s2) = \begin{cases} 1, if|overlap(s1, s2)| = max|overlap(s1, s2)| \\ 0, if|overlap(s1, s2)| = 0 \\ |overlap(s1, s2)|/max|overlap(s1, s2)|, otherwise \end{cases}$$

Here, $|overlap(s1, s2)|$ is the length of overlap of sequences *s1* and *s2*. Given sequences *s1*, *s2* where no overlap occurs, the possibility of similarity does not exist.

Confidence

The confidence μ_{qs} for each contig is defined as a measurement of the quality of the contributing base pairs [10]. A strong signal indicates a correct read or less chance of an experimental error. Every base involved in the contig has a quality score, and the entire sequence can be a mix of low and high quality bases. The confidence of a contig is the aggregate quality score of its contributing bases. For simplicity, the sum of weighted average quality scores is the confidence of the contig. The weight can be calculated as shown in (2). The bases with high quality are assigned a weight of 1. The bases that are of lower quality are given weights between 0 and 1, based on the cut-off value.

$$\mu_i = \begin{cases} 1, if \quad q_i \geq \delta \\ 0, if \quad q_i = 0 \\ (q_i - min_{qs})/(max_{qs} - min_{qs}), otherwise \end{cases} \tag{2}$$

In (2), δ is the threshold as explained earlier, generally specified by the user, and min_{qs} and max_{qs} are the minimum and maximum values for quality. The minimum and maximum values are obtained from the quality scoring algorithm. The equation below describes the membership function:

$$w_{qs} = \frac{\sum_{i=0}^{n} w_i q_i}{n} \tag{3}$$

In (3), μ_{qs} is the quality score for the overall overlap region, w_i is the weight used to standardize the quality scores, n is the number of bases, and q_i is the quality score of an individual base.

Gap Penalty

Gaps refer to regions of a sequence that are missing. These are divided into three categories: Inserts, Deletes, and Replacements. Affine gap penalty can be calculated as given in (4):

$$GapPen = GapOpening + Gaplength \times GapExtension \tag{4}$$

In the previous equation, *GapOpening* and *GapExtension* are scores for an opening or a continuation of a gap. The summation of (4) gives the entire gap penalty *GapPen(s1, s2)*. The membership function for gap penalty is given as follows:

$$\mu_{gp}(s1, s2) = \begin{cases} 1, if \quad GapPen(s1, s2) = 0 \\ 0, if \quad Overlap(s1, s2) \leq GapPen(s1, s2) \\ 1 - ((Overlap(s1, s2) - GapPen(s1, s2)))/(Overlap(s1, s2)), \\ \quad otherwise \end{cases}$$

Score

The score, denoted μ_{ws}, is calculated from the numbers of matching bases, indels and replacements. The score can be calculated by using different methods. For example:

$$score = n(MatchingBP) - n(Inserts) - n(Deletes) - n(Replacements)$$

Here, n refers to the count. The fuzzy membership function for the score is defined as

$$\mu_{ws}(s1, s2) = \begin{cases} 1, if\ tscore(s1, s2) = fmbp(s1, s2) \\ 0, if\ fmbp(s1, s2) \leq 0 \\ fmbp(s1, s2)/tscore(s1, s2), otherwise \end{cases} \tag{5}$$

where *fmbp (s1, s2)* is the score calculated using a scoring matrix and *tscore (s1, s2)* is score of the overlap if there were no indels or replacements. Detailed explanation of fuzzy membership functions and a sample scoring matrix can be found in [23].

Fuzzy Thresholds

Minmatch

Minmatch is the minimum number of matching bases that are required between the two sequences. It is not always possible to get a perfect overlap, and some amount of inexactness is tolerated. Therefore, we would like to have a minimum match value for the overlap sequences, which has a perfect match without any gaps. Generally, minmatch is used as a threshold to select or reject the contigs. A sigmoid membership function is used to select an optimal threshold for the minimum match required. The sigmoid function is given as

$$S(x, c) = \frac{1}{1 + e^{-(x-c)}} \tag{6}$$

In (6), x is the minmatch value selected by the user, and c is the break point that determines a transition from membership to non-membership.

Minscore

A score is calculated from the numbers of matching bases, indels and replacements as given previously in (5). Minscore is a threshold which specifies the minimum allowable score of the overlap. Minimum score is a commonly used parameter that sets a limit on the minimum score value that must be satisfied to accept a sequence as a match.

Aggregate Fuzzy Value

Once the fuzzy value for each of these parameters is calculated, it is combined into an function to determine the overall fuzzy value. To make a selection, this value needs to be defuzzified or converted to a crisp result. The aggregate fuzzy match value (AFV) acts as the defuzzification function. We employ the center of area (COA) defuzzification function that uses weighted average values of the fuzzy members. In a scenario of exact matching, perfect overlap can be defined as an overlap that satisfies the two thresholds, minmatch and minscore, is free of gaps, and satisfies the quality requirements. In a fuzzy system, this perfect match has a crisp value of 1. All matches that are closer to 1 than to 0 are known to be more similar. We define the fuzzy aggregate function in (7).

$$fa(c) = \mu_{qs} w_{qs} + \mu_{ws} w_{ws} + \mu_{gp} w_{gp} + \mu_{lo} w_{lo}, \quad (7)$$
$$where : w_{qs} + w_{ws} + w_{gp} + w_{lo} = 1$$

Each of the selected parameters has a weight w associated with them. These weights can be selected by the user to control the influence of an individual factor on the assembly. For example, to achieve a stringent assembly with the least gaps, w_{gp} can be set to a higher value. To obtain longer contigs, w_{lo} can be set to a higher value. A weight can be assigned a zero value so that the factor does not influence the assembly.

$$afv = fa(c)/m \quad (8)$$

In (8), m is the number of parameters, and *fa(c)* is given in (7). Equations (7) and (8) give the overall fuzzy function and the aggregate fuzzy function for m parameters. The subsequences that produce the highest fuzzy value for an overlap are selected as final sequences. Depending on their position as a suffix or prefix, a new contig or consensus sequence is formed.

3.3 LCS Clustering

Genome sequence assembly is a rigorous task that performs comparisons of a genome with every other genome present in the population. As discussed in the background review, there have been techniques to divide the fragments into groups. These groups are intended to be small and to have high similarity between the fragments.

In this work we perform a classification based on the AFV of the LCS. The idea of grouping based on the AFV derives from the fact that sequences that satisfy the overall requirement have higher similarity. These sequences have a higher chance of forming a consensus sequence. The process named ClusFGS is described in [25]. This technique improves the performance of assembly as shown in Fig. 2.

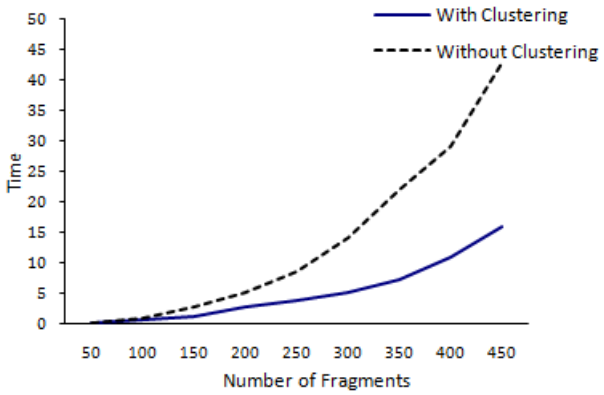

Fig. 2. Comparison of LCS with and without Clustering

3.4 Experiment and Results

The assembler was tested on artificially generated datasets and genome sequences obtained from GenBank belonging to different groups.The experiments for assembly are shown in Table 1 The results are compared with TIGR 2.0 [42]. The genomes used are listed as follows: (1) *The Wolbachia endosymbiont of the Drosophila melanogaster strain wMel 16S ribosomal RNA* gene containing 8,514 base pairs; (2) *Geobacillus thermodenitrificans NG80-2 plasmid pLW1071*, complete sequence, containing 57,693 base pairs; (3) *Yersinia pestis Pestoides F plasmid CD*, complete sequence, containing 71,507 base pairs; (4) *Arabidopsis thaliana genomic DNA, chromosome 3(ch3), BAC clone:F11I2*, geneid: F11I2.4. containing 36,034 base pairs; (5) *Ostreococcus tauri mitochondrion*, complete genome containing 44,237 base pairs; and, (6) *Phytophthora sojae mitochondrion*, complete genome containing 42,977 base pairs. All these genomes can be obtained from GenBank [26]. The total base was covered 4 times, *4X* of the original sequence. Each subsequence is in the range of 300-900 base pairs. In Table 1 , MGS = Multiple Genome Sequencing using a simple LCS implementation, TIGR = TIGR Assembler 2.0, FGS = Fuzzy Genome Sequencing. Since MGS did not perform well, we did not include it in further experiments. The next experiment was to separate two species. Sequences from two organisms was taken and mixed with each other. The input data appears as if it is from a single organism. ClusFGS algorithm is performed to group sequences from the organisms, into small classes, followed by assembly. In Table 3, ClusFGS is the method described in Subsect. 3.3 and is a modified version of FGS. Misclassification refers to the length of overall subsequences from genome 1 that were assembled incorrectly with contigs of genome 2. The results obtained in Table 1 from assembling the genome projects showed a high percentage of the genome recovered while using FGS and TIGR. Some of the small differences in results could be due to different thresholds being used. Preliminary results from Table 3 show that

Table 1. Assembly Comparisons of Different Sequences

	Percentage Genome Recovered		
Genome	MGS	TIGR	FGS
RPObc of Wolbachia genome	65%	99.6%	99.6%
Yersinia pestis Pestoides		93.9%	88.7%
Geobacillus thermodenitrificans		77.1%	91.6%
Arabidopsis thaliana ch3	56.8%	88.8%	92.135%
Ostreococcus tauri mitochondrion		77.7%	97.3%
Phytophthora sojae mitochondrion		97.7%	97.2%

Table 2. Table shows time for assembly and number of contigs obtained for Assembly of Sequences using FGS, the experiments were conducted on AMD Turion 64 X2 dual-core processor, with 4GB of RAM.

	FGS Assembly		
Genome	Time in Sec.	No. of Sequences	No. Contigs
RPObc of Wolbachia genome	146	100	15
Geobacillus thermodenitrificans	5800	650	195
Arabidopsis thaliana ch3	466	200	61
Phytophthora sojae mitochondrion	1085	300	72

Table 3. Clustering for Two Organisms with ClusFGS

Genome	Percentage Recovered	Miss-Classifications
P. sojae mitochondrion	61%	0%
G. thermodenitrificans	61.1%	0%

fuzzy classification was successful in grouping these two classes separately. The clustering classified the data into two groups without any misclassification. The clustering technique is linear, and hence, can make the assembly much faster. At this stage ClusFGS cannot recover a higher percentage of the genome because comparisons are done within a class. The performance is limited by factors such as random selection of the seeds, no interaction between classes such as reassignment of sequences, and smaller classes not being merged. This classification with some of it drawbacks is the inspiration for the new work that is presented in Sect. 4.

4 Fuzzy Classifier to Taxonomically Group DNA Fragments within a Metagenome

The metagenomic approach makes the acquisition of genomic fragments easier; nevertheless, the approach suffers from limitations. Recall from Sect. 2 that the diverse genomes acquired together may need to be separated and assembled to make meaningful conclusions. Taxonomical classification of ge-

nomic fragments is a vital problem in metagenomic approach. Because these microorganisms come from the same community, their characteristics are similar. Nevertheless, closely related bacteria can contain remarkable genomic diversity [49]. These differences can be found by analysis of features of the DNA, which we refer as DNA signature.

Pre-assembly grouping of metagenomic fragments into phylogenetic classes can lead to faster and more robust assembly by reducing the search space required to find adjacent fragment pairs, because DNA from the same organism should be classified into the same taxonomic group. The DNA signatures chosen are GC content, and tri- and tetra-nucleotide frequencies. The proposed method uses a fuzzy classifier and extracts signatures from given sequences and uses them as a feature set. The technique is verified with artificial shotgun sequences to measure correctness. The main purpose is to classify fragments of a community, which is depicted by classification of an acid mine drainage (AMD) environmental genome.

Even though studies have successfully taxonomically differentiated full genomes or fragments of sizes greater than 1,000 base pairs [20, 43, 51], there is a lack of availability of applications that classify shorter (500-900 base pairs) shotgun fragments. The proposed approach is designed with a goal of classifying shotgun fragments. Earlier techniques have focused on using a single signature for classification [51]. A combination of different signatures is proposed for the classification.

4.1 Background

Separation of domain-specific genomic fragments and reconstruction is a complex process that involves identification of certain features exhibited by entire taxonomic groups. These features are used to group the metagenomic sample into classes. The following subsection describes the DNA signatures that are employed in identification or classification of fragments.

DNA Signatures

Phylogenetically related groups of sequences show similar nucleotide frequencies either because of convergence or because they were inherited from a common ancestor [8]. For example, a study conducted on *E-coli* revealed a nonrandom utilization of codon pairs [14]. Some of the most frequent codon pairs found were: CTGGCG, CTGGCC, CTGGCA, CTGGAC, AACCCG, CTGGAA. This study and others reveal that there is a nonrandom over-representation and under-representation of certain codon pairs within a species. Oligonucleotide frequency studies with short x-x bases have reported tendencies of under- and over-representation in Xmers [5]. This study brought to attention that certain oligonucleotides are rarely observed in certain species while certain other oligonucleotides have shown their dominance in a particular species. This also shows that nucleotide composition contains bias.

The key to the classification of genomes is the presence of patterns in a sequence. Recall from Sect. 2.3, that these patterns can be specific to certain organism group. Thus, identification of these patterns can lead to the discovery of the phylogeny of a group. Moreover, the patterns can be used as signatures to distinguish one species from another. We now move our discussion to the two groups of signatures that will be utilized.

The first signature is based on GC content present in the genome. GC content is found to be variable with different organisms; this variation is viewed to be the result of variation in selection or bias in mutation [3]. For example, coding regions within a genome code for genes and are less divergent within populations. Genes represent characteristics of an organism: the physical development and phenotype of organisms can be thought of as a product of genes interacting with each other and with the environment [27]. Studies have shown that the length of the coding sequence is directly proportional to higher GC content [28], thus showing a strong correlation between GC content and gene properties. The pre-assembly of a well-known metagenomic dataset from acid mine drainage was performed by binning the fragments by their GC content [46].

The second signature that were investigated were the oligonucleotide frequencies. Nucleotide frequencies are generally taken from a group of two, three, four, five, or six nucleotides. These are known as di-, tri-, tetra-, penta- and dicodon nucleotide frequencies, respectively. These prefixes indicate the presence or absence of certain words in a genome that have been used to separate certain species. Evaluation of frequencies of fragments and their correlations based on taxonomy was performed by Teeling, *et al.* [43]. In this paper it was shown that GC content is not sufficient for separating species and tetra-nucleotide frequencies showed better differentiation of species for fragments of size 40,000 base pairs. A grouping based on nucleotide frequencies resembles the phylogenetic grouping of the representative organisms [33]. In another approach, differentiation of bacterial genomes was performed using statistical approaches for structural analysis of nucleotide sequences [35].

Frequencies of larger word sizes such as tetra, penta, and hexa are considered more reliable. But obtaining enough frequencies for larger words is difficult and may not give statistically relevant results for shorter fragments. There are a total of 4,096 dicodons. A sequence of length 10,000 bp contains 1,665 dicodons because this number is less than 4,096, the sequence cannot cover the 4,096 dicodons. The same sample contains 3,332 tri-nucleotide frequencies, that can easily cover the 64 tri-nucleotides. Therefore, it is better to use tri-nucleotide frequencies in cases of fragment classification.

Even though studies have successfully taxonomically differentiated full genomes or fragments of sizes greater than 1,000 base pairs [20, 43], there is a lack of availability of applications that classify shorter (500-900bp) shotgun fragments. Our approach is designed with a goal of classifying shotgun fragments. Earlier techniques have also focused on using a single signature for classification. We propose using a combination of different signature patterns.

Clustering

K-means is an unsupervised learning algorithm to group objects into categories. The simplest K-means algorithm places N objects into K classes by using the minimum distance from the center of K to each object. In the simple K-means approach, K is fixed a priori. Clustering problems generally derive some kind of similarity between groups of objects. K-means clustering is a simple and fast approach to achieve a grouping for data. Due to its simple method of using feature vectors as seeds and the arithmetic mean as the center for the clusters, the K-means algorithm suffers from drawbacks. The simple K-means algorithm could not guarantee convergence. A modified K-means was developed that uses a weighted fuzzy average instead of the mean to get new cluster centers. Using a fuzzy weighted average instead of a simple mean improved K-means and also leads to convergence [19]. In this research, a modification of the fuzzy K-means algorithm with fuzzy weighted averages is used for fragment clustering. The algorithm is described in the next subsection.

4.2 An Overview of Our Algorithm

Fragment classification divides entire datasets into smaller categories. The classes represent two significant properties: (1) they contain fragments belonging to the same group or species present in the metagenomic data set, and (2) they have continuity and can represent local regions of the genome. The first step to classification is the identification of the signatures for each fragment. After the signatures are extracted the feature vector is initialized, and the K-means algorithm is run to create classes. The operations carried out is be described next.

GC Content

GC content is expressed as the percentage of G and C present in the fragment and is calculated as follows:

$$\frac{C+G}{A+C+G+T} \times 100$$

Certain factors need to be considered when using GC content. GC content is known to be more influential in coding regions. Shotgun fragments of metagenomes do not contain information that reveals directly whether a certain fragment contains coding regions or the percentage of fragment region that can code for a gene. Analysis of GC content revealed that it is not sufficient to obtain a classification when closely related species are present in the datasets. Thus, advanced signatures are required to obtain a good separation of groups within a metagenome.

Nucleotide Frequencies Using Markov Chain Model

Markovian models have been used in fields such as statistics, physics and queuing theory. Markov chain predictors have also been used to predict coding regions, thus finding genes. The simplest chain is the zero-order Markov chain which can be estimated from the frequencies of the individual nucleotides A, C, G, and T. The approach used to estimate the zero-order Markov chain is shown below. Consider the sequence GGATCCC, the nucleotide frequency is given by:

$$p(GGATCC) = p(G)p(G)p(A)p(T)p(C)p(C)$$

Higher order Markov chains can also be constructed using only the previous state frequencies. A maximal-order Markov chain removes biases from all the previous states and is dependent on only the past state. The tri-nucleotide and tetra-nucleotide frequencies can be calculated using a maximal-order Markov chain. Expected values are directly calculated from the observed values as shown in (9). In (9) and (10), O refers to the observed values, E is the expected value, and N_i refers to a nucleic acid base pair.

$$E(N_1N_2N_3) = \frac{O(N_1N_2)O(N_2N_3)}{O(N_2)} \tag{9}$$

$$E(N_1N_2N_3N_4) = \frac{O(N_1N_2N_3)O(N_2N_3N_4)}{O(N_2N_3)} \tag{10}$$

Fuzzy K-means Clustering

Clustering for a metagenome assembly problem has a two fold purpose: to divide the space for performance improvement and to group fragments into classes such that each class has fragments from one group. The K-means algorithm uses a set of unlabeled feature vectors and classifies them into K classes. From the set of feature vectors K of them are randomly selected as initial seeds. The feature vectors are assigned to the closest seed. The mean of features belonging to a class is taken as the new center.

Given N sequences, such that $S = \{C\}^i$,where C ={A, C, G, T}. We randomly select K sequences as the initial seeds, where K is less than the number of sequences N. The nucleotide frequencies and GC content for all sequences are calculated. These frequencies form the p features to be used in classification.

The sequence is assigned to the class that has the highest fuzzy similarity. The fuzzy similarity is calculated using a weighted fuzzy average (WFA). Let $\{x_1, ..., x_P\}$ be a set of P real numbers. The weighted fuzzy average is using the weight w_p for x_p is given as:

$$\mu^r = \sum_{p=1,P} w_p^{(r)} x_p, \;\; r \;\; = 0, 1, 2, \ldots$$

Here, x is the parameter or feature and p the number of features. The number of the iteration is given as r. The mean is obtained for all the K initial classes. The next step is to assign features to each of the classes. A feature is assigned to the closest class by computing the distance of a feature from each of the classes. Given i=0,. . . ,N and j=0,. . . ,k, the distance $d_{i,j}$ for each cluster can be calculated as follows:

$$d_{i,j} = max(\mu_j^r), for\ all\ j = 0, \ldots, k$$

Thus feature vectors are assigned to a class. Since a large number of classes were created initially, empty or small classes are eliminated. Classes that are close to each other are merged to form one class. This process is repeated until convergence by replacing the initial mean with the WFA, and feature vectors are reassigned by computing the distance. In the next subsection we show the results obtained by classification and describe the genomes used to test the approach.

4.3 Clustering *via* Feature Extraction

Artificial Metagenome

To asses the performance of fuzzy clustering on genomic sequences, experiments on artificial data were performed. In the first experiment, two genomes from different phylogenetic types are used for the first test case. These fragments are mixed with each other. Table 4 shows the results obtained after classifying these two samples. In Tables 4 and 6, GC refers to clustering with GC content, T_z refers to tri-nucleotide and TR_z refers to tetra-nucleotide frequencies using zero-order Markov chains. TR_m refers to tetra-nucleotide frequencies using a maximal-order Markov chain, T_m indicates the tri-nucleotide frequencies using a maximal-order Markov chain. Combinations of different signatures are shown by hyphenating individual frequencies. A value of NA indicates that the signatures could not separate the fragments into groups and all the data was placed into one class. In the second experiment, the dataset that was described in Sect. 3.4 was used. For this experiment we conducted not only classification but also assembly of the sequence using signature-based classification. Recall results from Table 3, that classified genome fragments using an LCS-based approach. The results of our classification and assembly are shown in Table 5. These results indicate improvement in assembly using the signature-based method, even though there are few misclassifications. The reason for the misclassifications is that ClusFGS classifies based on LCS, which considers the entire sequence for classification. Whereas signature-based classification uses signatures without creating an overlap, this also makes the approach much faster than ClusFGS.

Table 4. Separating 500 Fragments Belonging to Two Organisms Using Different Signatures

	# Fragments classified incorrectly		
Signature	Genome 1	Genome 2	Total %
GC	39	1	0.08
T_z	NA	NA	NA
TR_z	NA	NA	NA
$GC - T_z - TR_z$	18	32	0.1
T_m	7	0	0.02
TR_m	15	0	0.03
T_m- TR_m	11	3	0.028
$GC - T_m - TR_m$	5	1	0.012

Table 5. Clustering and Assembly of 800 Artificial Metagenome fragments

Genome	Percentage Recovered	Miss-Classifications
P. sojae mitochondrion	82.90%	0%
G. thermodenitrificans	94.124%	1.6%

The Acid Mine Drainage Metagenome

The AMD metagenome was obtained from Richmond Mine at Iron Mountain, CA [46]. The acid mine drainage environmental genome was shown to contain 2 major groups. We use shotgun sequences of two genomes of AMD, namely *Leptospirillum sp. Group II* (Lepto) environmental sequence and *Ferroplasma sp. Type II* (Ferrop. Type II) environmental sequence. These sets are 960,150 and 1,317,076 nucleotide base pairs respectively. The first group belongs to the bacterial genus *Leptospirillum*; the second one is an archea from the genus *ferroplasma*. Shotgun sequences of average size 700 base pairs were generated from these genomes. Fig. 3 depicts the classification results on AMD data, using GC content. A set of 3,000 samples was used for the display. The tests were conducted successfully for all 5 sub-groups in AMD.

The results of classification using the modified K-means approach using DNA signatures is given in Table 6. It compares the classification results for the two AMD genomes. Classification was performed using different combinations of signatures, and the results are displayed in Table 6. The final classification resulted in two groups, one with fragments from *Lepto* and another with *Ferrop Type II* fragments respectively. The results indicate that frequencies obtained using maximal-order Markov chain created the better classification than zero-order Markov chain. A combination of different signatures also resulted in fewer misclassifications.

Taxonomy is a method of classifying organisms into types and further classifying types into subtypes to form a hierarchical structure. All species are classified into hierachical groups starting with *domain*, *kingdom*, *phylum*, *class*,

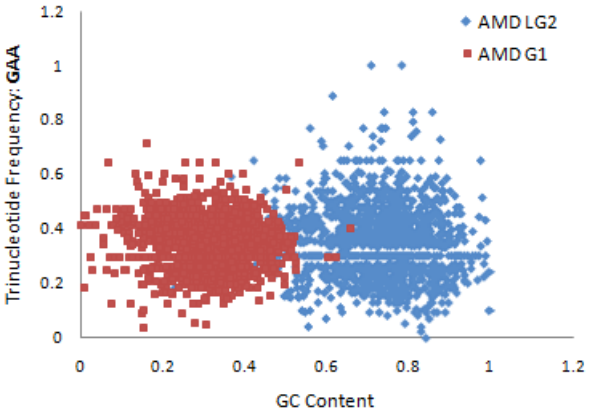

Fig. 3. Classification using GC Content and Nucleotide Frequency for Shotgun Sequence Fragments obtained from AMD G1 and AMD LG2

Table 6. Separating 20,000 Fragments from AMD into Two Classes Using Different Signatures

	# Fragments classified incorrectly		
Signature	Lepto.	Ferrop. Type II	Total %
GC	500	27	0.026
T_z	NA	NA	NA
TR_z	NA	NA	NA
$GC - T_z - TR_z$	640	27	0.033
T_m	147	16	0.0081
TR_m	170	6	0.0088
T_m- TR_m	127	10	0.0068
$GC - T_m - TR_m$	129	11	0.007

order, *family*, *genus* and *species*. Organisms that belong to different domains can have genome sequences that are different. But as we go down the hierarchy the similarities increase, therefore organisms that belong to same species are highly similar. As similarities between organisms increase it becomes difficult to cluster them. A study on the classification of fragments to identify the accuracy of the classifier can be found in [23]. Analysis of certain pairs also shows that there is over- and under-representation of certain oligonucleotide words. The results of classification indicate that at higher ranks in the taxonomy the classifier works well and the classification gradually decreases after which there is sharp increase in miss-classifications. Advanced signatures or supervised clustering can be a potential approach for organisms that are more similar.

5 Conclusions and Future Work

This body of work contributes an effective framework for assembly of sequences using fuzzy logic. The work was initiated to create an assembler that can work on metagenome fragments without pre-processing. The fuzzy assembly process can successfully assemble sequences. The functions proposed can be easily adapted in other assembly methods or techniques.

This classifier is enhanced to use DNA signatures to perform a phylogenetic classification. A fuzzy clustering algorithm is proposed to classify shotgun genome fragments into taxonomical classes. We classified fragments using different signatures and combination of signatures. We also tested the AMD metagenome and classified it into two groups of bacteria and archea. Using combination of DNA signatures also showed good classification. Results were obtained for different types of genomes sequences, thus testing a wide range of input genomes. Prior to this work, classification was performed on full genomes or fragments that were longer than 1000bp. This work shows that fragments of smaller length can also be classified into groups. We propose an unsupervised classification that requires, no training or identification of important nucleotides. A known limitation of the classification technique is that the classes have to be set by the user. If the classes are not set, the K-means algorithm determines final groups. The algorithm creates classes that are compact rather than classes that are large and dispersed. Thus fragments from one genome, can be present in more than one class, ensuring classes with minimal or no misclassifications. The technique can be improved by application of validity measures, using marker regions to identify and create groups that can represent a number of genomes within the sample.

This work opens a question of using an adaptive assembler that can adapt itself to the input to generate the best possible assembly. The concept of adaptive assembler is dependent on two factors: statistical analysis of data and the best approximation of the parameters. The assembly can be further improved by data reduction before assembly making it possible to run larger data sets at faster speeds. A parallel version of the assembler can be found faster assembly in [23]. The classification proposed can also be enhanced by generating signatures that are different from each other rather than selecting random signatures. The results indicate that we are able to group shotgun sequences from their frequencies and GC Content. Analysis of the DNA signatures can be done to find the best discriminatory pairs, enabling selection of features that suit the dataset best rather than using all available frequencies.

References

1. A. D. Baxevanis and B. F. Ouellette. *Bioinformatics : A practical guide to the analysis of genes and proteins.* John Wiley, 1^{st} edition, 2005.

2. O. Beja, M. Suzuki, E. Koonin, L. Aravind, A. Hadd, L. Nguyen, R. Villacorta, M. Amjadi, C. Garrigues, S. Jovanovich, R. Feldman, and E. DeLong. Construction and analysis of bacterial artificial chromosome libraries from a marine microbial assemblage. *Environmental Microbiology*, **2**:516–529, 2000.
3. J. Birdsell. Integrating genomics, bioinformatics and classical genetics to study the effects of recombination on genome evolution. *Molocular Biology Evolution*, **19**:1181–1197, 2002.
4. T. Brown. *Genomes*. Garland Science, 3^{rd} edition, 2006.
5. C. Burge, A. Campbell, and S. Karlin. Over- and under-representation of short oligonucleotides in DNA sequences. *Proceedings National Acaddemy of Science USA*, **89**(4):1358–1362, 1992 Feb 15.
6. K. Chen and L. Pachter. Bioinformatics for whole-genome shotgun sequencing of microbial communities. *PLoS Computational Biology*, **1**:106–112, 2005.
7. S. Choudhuri. The path from nuclein to human genome: A brief history of DNA with a note on human genome sequencing and its impact on future research in biology. *Bulletin of Science Technology Society*, **23**:360–367, 2003.
8. G. C. Conant and P. O. Lewis. Effects of nucleotide composition bias on the success of the parsimony criterion in phylogenetic inference. *Molecular Biology Evolution*, **18**:1024–1033, 2001.
9. T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. McGraw-Hill, 2^{nd} edition, 2001.
10. B. Ewing and P. Green. Basecalling of automated sequencer traces using phred. ii. error probabilities. *Genome Research*, **8**:186–194, 1998.
11. R. Fleischmann, M. Adams, O. White, R. Clayton, E. Kirkness, A. Kerlavage, C. Bult, B. Tomb, J Dougherty, and J. Merrick. Whole-genome random sequencing and assembly of Haemophilus Influenzae Rd. *Science*, **269**(5223):496–512, 1995.
12. A. P. Gasch and M. B. Eisen. Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. *Genome Biology*, **3**(11):1–22, 2002.
13. P. Green. *Documentation for Phrap*. Genome Center, University of Washington, 2006.
14. G. Gutman and G. Hatfield. Nonrandom utilization of codon pairs in Escherichia coli. *Proceedings National Academy of Science USA*, **86**:3699–3703, 1989.
15. X. Huang and A. Madan. CAP3: A DNA sequence assembly program. *Genome Research*, **9**(9):868–877, 1999.
16. P. Hugenholtz. Exploring prokaryotic diversity in the genomic era. *Genome Biology*, **3**:reviews0003.1–reviews0003.8., 2002.
17. S. Karlin, I. Ladunga, and B. Blaisdell. Heterogeneity of genomes: Measures and values. *Proceedings National Academy of Science USA*, **91**:12837–12841, 1994.
18. J. Kececioglu and E. Myers. Combinatorial algorithms for DNA sequence assembly. *Algorithmica*, **13**:7–51, 1995.
19. C. Looney. Interactive clustering and merging with a new fuzzy expected value. *Pattern Recognition*, **35**:2413–2423, November 2002.
20. A. McHardy, H. Martn, A. Tsirigos, P. Hugenholtz, and I. Rigoutsos. Accurate phylogenetic classification of variable-length DNA fragments. *Nature Methods*, **4**(1):63–72, 2007.
21. E. Mongodin, J. Emerson, and K. Nelson. Microbial metagenomics. *Genome Biology*, **6**(10):347, 2005.

22. G. Myers. Whole-genome DNA sequencing. *IEEE Computational Engineering and Science*, **1**:33–43, 1999.
23. S. Nasser. *Fuzzy Sequence Classification and Assembly of Environmental Genomes*. PhD thesis, University of Nevada Reno, 2008.
24. S. Nasser, G. Vert, M. Nicolescu, and A. Murray. Multiple sequence alignment using fuzzy logic. In *Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, volume **7**, pages 304–311, 2007.
25. S. Nasser, G. L. Vert, A. Breland, and M. Nicolescu. Fuzzy classification of genome sequences prior to assembly based on similarity measures. In *North American Fuzzy Information Processing Society*, pages 354–359, 2007.
26. NCBI. National center for biotechnology information. http://www.ncbi.nlm.nih.gov/, NIH, 2007.
27. M. A. Nowak. *Evolutionary Dynamics: Exploring the Equations of Life*. Belknap Press,, 1^{st} edition, October 2006.
28. J. Oliver and A. Marn. A relationship between GC content and coding-sequence length. *Journal of Molecular Evolution*, **43**(3):216–223, 2004.
29. H. H. Otu and K. Sayood. A divide-and-conquer approach to fragment assembly. *Bioinformatics*, **19**(1):22–29, 2003.
30. H. Peltola, H. Soderlund, and E. Ukkonen. Seqaid: A DNA sequence assembling program based on a mathematical model. *Nucleic Acids Research*, **21**(1):307–321, 1984.
31. E. Pillsbury. A history of genome sequencing. Technical report, Yale University Bioinformatics, 2001.
32. M. Pop, S. L. Salzberg, and M. Shumway. Genome sequence assembly: Algorithms and issues. *IEEE Computer*, **35**(7):47–54, July 2002.
33. D. T. Pride, R. J. Meinersmann, T. M. Wassenaar, and M. J. Blaser. Evolutionary implications of microbial genome tetranucleotide frequency biases. *Genome Research*, **13**(2):145–158, February 1, 2003.
34. M. Rappe and S. Giovannoni. The uncultured microbial majority. *Annual Reviews Microbiology*, **57**:369–394, 2003.
35. O. Reva and B. Tmmler. Differentiation of regions with atypical oligonucleotide composition in bacterial genomes. *BMC Bioinformatics*, **6**(1):251, 2005.
36. M. Rondon, P. August, A. Bettermann, S. Bradly, T. Grossman, M. Liles, K. Loiacono, B. Lynch, I. MacNeil, C. Minor, C. Tiong, M. Gilman, M. Osburne, J. Clardy, J. Handelsman, and R. Goodman. Cloning the soil metagenome: a strategy for accessing the genetic and functional diversity of uncultured microorgansims. *Applications Environmental Microbiology*, **66**:2541–2547, 2000.
37. K. Sadegh-Zadeh. Fuzzy genomes. *Artificial Intelligent Medicine*, **18**(1):1–28, 2000.
38. F. Sanger, A. Coulson, G. Hong, D. Hill, and G. Petersen. Nucleotide sequence of Bacteriophage Lambda DNA. *Journal Molecular Biology*, **162**(4):729–773, 1982.
39. F. Sanger, S. Nicklen, and C. AR. DNA sequencing with chain-terminating inhibitors. *Proceedings National Academy of Science USA*, **74**(12):5463–5467, 1977.
40. T. Smith and M. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, **147**:195–197, 1981.

41. J. Stein, T. Marsh, K. Wu, H. Shizuya, and E. DeLong. Characterization of uncultivated prokaryotes: isolation and analysis of a 40-kilobase-pair genome fragment from a planktonic marine archaeon. *Journal of Bacteriology*, **178**:591–599, 1996.
42. G. Sutton, O. White, M. Adams, , and A. Kerlavage. TIGR Assembler: A new tool for assembling large shotgun sequencing projects. *Genome Science & Technology*, **1**:9–19, 1995.
43. H. Teeling, A. Meyerdierks, M. Bauer, R. Amann, and F. Glockner. Application of tetranucleotide frequencies for the assignment of genomic fragments. *Environmental Microbiology*, **6**:938–947, 2004.
44. H. Teeling, J. Waldmann, T. Lombardot, M. Bauer, and F. O. Glockner. TETRA: A web-service and a stand-alone program for the analysis and comparison of tetranucleotide usage patterns in dna sequences. *BMC Bioinformatics*, **5**:163, 2004.
45. P. Turnbaugh, R. Ley, M. Mahowald, V. Magrini, M. ER, and G. JI. An obesity–associated gut microbiome with increased capacity for energy harvest. *Nature*, **444**(7122):1009–10, 2006.
46. G. W. Tyson, J. Chapman, P. Hugenholtz, E. E. Allen, R. J. Ram, P. M. Richardson, V. V. Solovyev, E. M. Rubin, D. S. Rokhsar, and J. F. Banfield. Community structure and metabolism through reconstruction of microbial genomes from the environment. *Nature*, **428**:37–43, 2004.
47. E. Ukkonen. On-line construction of suffix trees. *Algorithmica*, **14**(3):249–260, 1995.
48. J. C. Venter, K. Remington, J. F. Heidelberg, A. L. Halpern, D. Rusch, J. A. Eisen, D. Wu, I. Paulsen, K. E. Nelson, W. Nelson, D. E. Fouts, S. Levy, A. H. Knap, M. W. Lomas, K. Nealson, O. White, J. Peterson, J. Hoffman, R. Parsons, H. Baden-Tillson, C. Pfannkoch, Y.-H. Rogers, and H. O. Smith. Environmental genome shotgun sequencing of the sargasso sea. *Science*, **304**:66–74, 2004.
49. R. Welch, V. Burland, G. Plunkett, P. Redford, P. Roesch, D. Rasko, E. Buckles, S. Liou, A. Boutin, J. Hackett, D. Stroud, G. Mayhew, D. Rose, S. Zhou, D. Schwartz, N. Perna, H. Mobley, M. Donnenberg, and F. Blattner. Extensive mosaic structure revealed by the complete genome sequence of uropathogenic Escherichia coli. *Proceedings National Academy of Science USA*, **99**(26):17020–17024, 2002.
50. L. Wong. *The Practical Bioinformatician*. World Scientific Publishing Company, 1^{st} edition, 2004.
51. T. Woyke, H. Teeling, N. Ivanova, M. Huntemann, M. Richter, F. Gloeckner, D. Boffelli, I. Anderson, K. Barry, H. Shapiro, E. Szeto, N. Kyrpides, M. Mussmann, R. Amann, C. Bergin, C. Ruehland, E. Rubin, and N. Dubilier. Symbiosis insights through metagenomic analysis of a microbial consortium. *Nature*, **443**(7114):925–7, 2006.
52. D. Xu, R. Bondugula, M. Popescu, and J. Keller. Bioinformatics and fuzzy logic. In *IEEE International Conference on Fuzzy Systems*, pages 817–824, 2006.
53. L. A. Zadeh. *Fuzzy logic and approximate reasoning*, volume **30**. Synthese, 1975.
54. Z. Zhang, S. Schwartz, L. Wagner, and W. Miller. A greedy algorithm for aligning DNA sequences. *Journal of Computational Biology*, **7**:203–214, 2000.