Stockholms universitet

Choosing Representative Gene Transcripts

Hur man väljer Representativa Gentranskript

Ram Ali

Handledare: Lars Arvestad
Examinator: Marc Hellmuth
Inlämningsdatum: 2022-05-24

Kandidatuppsats i datalogi
Bachelor Thesis in Computer Science

Abstract

It is known that a single gene can give rise to not only one but several different alternative protein variants through alternative splicing. It can be troublesome to compare a protein sequence with other protein sequences if it is one of many alternative sequences produced by a single gene. In such a case the gene's longest sequence is usually selected as its representative sequence. We suggest an algorithm that selects, for each input gene, a representative sequence based on the sequence's similarity to the other genes' sequences. The algorithm is tested on simulated data created with the *SimSpliceEvol* program, and the output representatives are compared with the longest sequences by creating a multiple sequence alignment (MSA) for both of them respectively. The MSAs are scored with the sum-of-pairs method. In 69.2 percent of 321 cases, the output representatives' MSAs scored higher than the longest sequences'. However, since the algorithm frequently selected the longest sequences as representatives, choosing the longest sequence as a gene's representative may not be the worst choice most of the time. This project may provide a new perspective of selecting a gene's representative sequence with this approach if it has not been explored already.

Sammanfattning

Det är redan känt att en enda gen kan ge upphov till flera olika alternativa protein varianter genom alternativ splitsning. Det kan vara besvärligt att försöka att jämföra en protein sekvens med andra protein sekvenser om den är en av flera alternativa sekvenser som är producerad av en enda gen. I ett sådant fall brukar genens längsta sekvens väljas som dess representativa sekvens. Vi föreslår en algoritm som väljer, för varenda gen som finns i indatan, en representativ sekvens beroende på sekvensens likhet med de andra genernas sekvenser. Testning för algoritmen görs på simulerad data som är skapade med hjälp av *SimSpliceEvol* programmet. Utdatan, alltså de representativa sekvenser, av algoritmen jämförs med de längsta sekvenserna genom att en *multiple sequence alignment* (MSA) skapas för de representativa och de längsta sekvenserna vardera, och båda *alignments* får poäng genom *sum-of-pairs* metoden. I 69.2 procent av 321 fall har de representativa sekvenserna fått högre poäng än de längsta sekvenserna. Dock, eftersom algoritmen valde de längsta sekvenser som representativa sekvenser för en stor andel av testerna kan det verka som att valet av den längsta sekvensen som en gens representativa inte är det värsta valet i de flesta fallen. Detta projekt kan ge ett nytt perspektiv på val av en gens representativa sekvens med hjälp av våra metod om det inte redan hade utforskats.

Contents

1 Introduction

1.1 Sequence alignment

All organisms have cells which each contain a DNA molecule, including ours [1]. A common task in bioinformatics is to compare two or more sequences of DNA, RNA, or protein. A DNA molecule contains information for building and maintaining an organism, and the information is stored by a sequence made up of the four chemical bases along the molecule. The four bases are adenine (A), guanine (G), cytosine (C) and thymine (T). An RNA molecule is similar to a DNA molecule in that their sequences are made up of the same bases except for thymine, which is replaced by uracil (U). RNA molecules are used for different tasks within the cell, one of which is to transfer information from the DNA to the production of a protein [2]. Proteins perform most of the tasks in a cell and are vital for the function and regulation of an organism's body [3]. Before a specific protein is produced, a gene's sequence, i.e. a subsequence of the DNA that contains the information for building the protein, will first be transcribed into an RNA molecule [4]. The RNA's sequence of base pairs will then be translated to a sequence of amino acids, which is what makes up the protein. Sequence alignments are used in order to study the similarity between sequences for various reasons, one of which is phylogenetic analysis, which involves studying and comparing species, genes or proteins, and estimating the evolutionary relationships between them [5].

An alignment of two sequences, called a pairwise alignment, is done by placing the sequences in rows on top of one another and having identical symbols placed in the same column while having nonidentical symbols either placed in the same column or one of them placed against a gap in the other sequence [6]. For example, in the alignment in Figure 1, the last 'G' symbol of the second sequence is placed against a gap, '-', in the first sequence. Meanwhile, in the alignment in Figure 2, the two symbols 'G' and 'C' are placed in the same column as a mismatch, marked here with a '.' instead of a '|' which otherwise marks a match. Sequences being aligned will have the same length in the resulting alignment.

```
AGCUUUCAAGCUAA-
||    ||||||||
AG----CAAGCUAAG
```

Figure 1: Pairwise alignment of RNA sequences 'AGCUUUCAAGCUAA' and 'AGCAAGCUAAG'. When aligning closely related sequences, gaps can be interpreted as evolutionary events, such as mutations. Here, for example, either the first sequence gained the 'CUUU' segment or the second sequence lost it since both sequences evolved from their common ancestor.

```
AGCUUUCACAGCUAA
||    ||.||||||
AG----CAGAGCUAA
```

Figure 2: Pairwise alignment of RNA sequences 'AGCUUUCACAGCUAA' and 'AGCAGAGCUAA'.

To find the optimal alignment one would have to first define a scoring scheme, which gives a score value to every possible pair of symbols that can occur in an alignment column of a pairwise alignment. For an alignment of two sequences for example, one could have a scoring scheme that yields 1 point for a column with a matching pair of symbols, -1 points for a column with a gap symbol, and -2 points for a column with a mismatch. The optimal alignment in this case would be the alignment that scores the highest with this scoring scheme.

One can align sequences either globally or locally. Global alignment is when the sequences are aligned in their entirety, resulting in an alignment containing all the symbols from the sequences, like the ones in Figures 1 and 2 [6]. On the other hand, local alignment is when segments of the sequences with the highest density of matches are aligned, resulting in an alignment that can consist of several subalignments of subsequences, and can thus leave out other segments of the sequences. The alignment in Figure 3 is an example of a local alignment, in which only the segments of the sequences that are most alike are aligned. The dashes here indicate both gaps and symbols of the sequences that are not included in the alignment.

```
GCACTTAAGTA
   ||.|| |
---CTGAA-T-
```

Figure 3: A local alignment of RNA sequences 'GCACTTAAGTA' and 'CTGAATG'.

Local alignment is more suitable for finding conserved patterns in sequences [6]. It is also more appropriate if, for example, the sequences to be aligned are nucleotide sequences that contain genes, but the exact boundaries of the genes are not known, and so one would not want to include segments that are actually outside the genes' boundaries in the alignment [7]. On the other hand, global alignment is more suitable for cases like finding mutations in closely related gene or protein sequences, where the sequences are more similar [8].

1.2 Alternative splicing

The genome of an organism provides the information for the organism to function [9]. It contains the genetic information stored in DNA molecules. Gene expression is the process where information stored in a segment, called a gene, of a DNA molecule is used to synthesize products, particularly proteins. Transcription is the first stage in this process, where the DNA sequence of the gene is transcribed into an RNA transcript. In Eukaryotes many genes give rise to RNA transcripts that have to go through what is called splicing, in which segments of an RNA transcript are removed, and the rest are spliced together to form a mature RNA transcript, ready to be translated into a protein.

The genes of eukaryotes usually contain segments of introns and exons. Introns are the regions in the gene's nucleotide sequence that are non-coding for the resulting protein, and exons contain the coding regions. When an RNA transcript goes through the splicing process, the introns are the segments that are removed while the exons are the ones that are spliced together. See Figure 4 for an example of an RNA transcript being spliced.

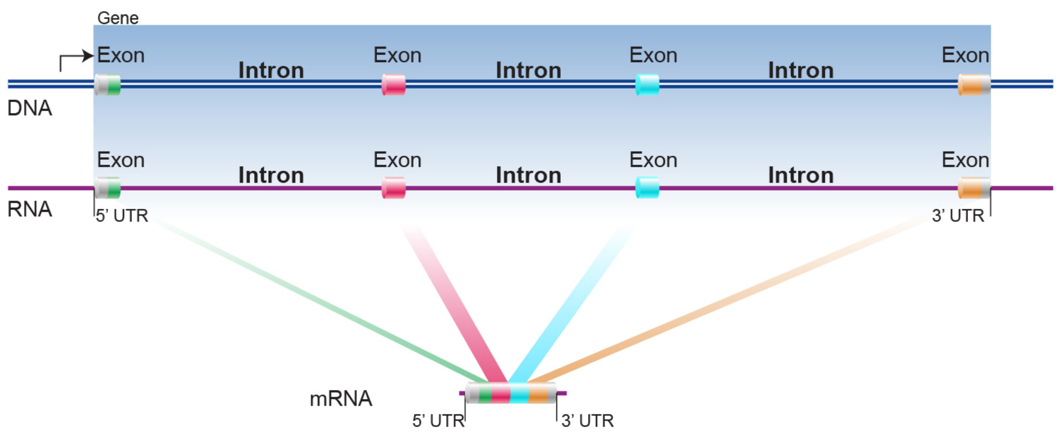

Figure 4: A gene with introns and exons. The RNA transcript of the gene goes through the splicing process and the result is a messenger RNA (mRNA), which is ready to be translated into a protein. *Courtesy: National Human Genome Research Institute* [10].

The resulting sequence of an RNA transcript that has gone through the splicing process isn't always the same though. Alternative splicing events can occur, during which introns can be retained in the resulting sequence (intron retention), or exons can be excised instead (exon skipping). The 5' and 3' splice sites of introns, which are the ends of the intron where the removal of it during splicing starts and ends, can also be altered, causing there to be alternative splice sites [11].

These alternative splicing events can give rise to several alternative RNA transcripts, and in turn, to several protein variants from a single gene. Since these resulting proteins differ in their amino acid sequences, so too may their functions [12]. A study done by Pan et al. suggests that around 95% of human genes that contain multiple exons produce transcripts that undergo alternative splicing [13].

1.3 Sequence alignment programs

There are several available programs for both pairwise sequence alignment, that is the alignment of two sequences, and multiple sequence alignment (MSA), the alignment of three or more sequences. Although there are a multitude of these programs, they do not take into account alternative transcripts if one would want to align RNA transcripts or protein sequences. If one would want to, for example, align the protein sequences from a set of genes, and some of those genes happen to be able to produce several alternative protein sequences, one might have to resolve this issue by selecting a representative sequence for each of those genes before the alignment.

The *RefSeq Select* dataset provides representative transcripts for protein-coding genes based on multiple selection criteria, including the transcripts' prior use in clinical databases [14]. The *RefSeq Select* dataset is currently available for human and mouse genes, and is planned to provide sets for other eukaryotes. This unburdens scientists of choosing representative transcripts but currently only for human and mouse genes. In the case where one has to manually choose, the longest transcript is usually chosen. This is not always the best choice, especially if the longest transcript is the longest simply because it accidentally contains introns, which are non-coding for the protein. It could therefore be beneficial to have another method of selecting representative transcripts in case there are no such datasets for the genes being studied.

1.4 Suggested algorithm

We suggest and test an algorithm that takes as input a set of genes, each having one or more transcripts, and the algorithm will select a single transcript as a representative for each gene. The selection of the representative transcript of a gene is based on the transcript's sequence similarity with the other genes' transcripts. The algorithm, using pairwise alignment to calculate the similarity between two sequences, selects the transcript with the highest sequence similarity with the other genes' transcripts.

The reason for this way of selecting representative transcripts is because it has been shown that for a pairwise alignment of two related proteins with at least 50% residue identity, that is the percentage of matching symbols in the alignment, the functions of the proteins are likely to be significantly similar [12]. There are, of course, exceptions where similar sequences do not have similar function, and conversely, distantly related and dissimilar protein sequences that do have similar function. This can lead to erroneous assumptions about a protein's function, but we choose to base our choice of the representative transcript on this correlation between sequence similarity and function similarity. We will then compare our method of selecting representative sequences with the method of choosing the longest sequence as representatives.

2 Method

The algorithm is written in Python and uses the *Bio.pairwise2* module for creating pairwise alignments [15].

2.1 Algorithm

2.1.1 Input

The input for the algorithm is a set G of genes. The genes in turn are sets containing sequences of either only cDNA (coding DNA), only RNA or only protein residues. For example, let $g \in G$, and $g = \{s_1, s_2\}$ where s_1 and s_2 are the two sequences 'AGCUUUCAAGCUAA' and 'AGCAAGCUAAG' from Figure 1 respectively.

2.1.2 Output

The algorithm returns the set G_r, which holds, for each $g \in G$, a corresponding element g_r, which is the representative sequence for g. For example, if $h \in G$ and $h = \{s_1, s_2, s_3\}$, and s_2 is the representative sequence of h, then $h_r = s_2$.

A sequence, s, of a gene, $g \in G$, is selected as the representative, g_r, if the sum of the scores of its alignments with the sequences of all the other genes in G is the highest among the sequences in g. Hence, in the example above, s_2 was selected as h_r because its alignment scores must have been, in total, higher than both s_1's and s_3's.

2.1.3 Procedure

The algorithm iterates over all genes $g \in G$. Within each iteration the algorithm again iterates over all the sequences $s \in g$, and in each of these iterations, global pairwise alignments between s and each of the sequences of the rest of the genes in G are created and scored. Let A_g be a set which contains, for each sequence s in g, a set of alignment scores between s and the other genes' sequences in G. This is expressed mathematically as

$$A_g = \Big\{ score(s,t) : s \in g,\ \ t \in \bigcup_{g_2 \in G \setminus \{g\}} g_2 \Big\},$$

where $g \in G$, and where $score(s,t)$ is the alignment score between the sequences s and t.

The pairwise alignments will be performed and scored using the *Bio.pairwise2* Python module [15]. After all of the sequences $s \in g$ have been aligned with the other genes' sequences, the one that has the highest sum of alignment scores is selected as g's representative protein sequence. Expressed mathematically,

$$\underset{s \in g}{argmax} \sum_{g_2 \in G \setminus \{g\}} \sum_{t \in g_2} score(s,t)$$

contains, for each sequence s in g, the sum of its alignment scores. The sequence with its sum of alignment scores equal to $max(A_{g_{sum}})$ will be g's representative. After all genes $g \in G$ have been iterated over with the process just described above, we will have a representative sequence for each of them.

The pairwise alignments are scored by having every matching pair of symbols add 2 points, and every mismatching pair is penalized with -1 points. Gaps are also penalized, and we use the concept of the *affine gap penalty* [16]. The opening gap, that is the first gap symbol following an RNA residue, adds -0.5 points, and the symbols of the gap extention, that is the subsequent gap symbols following the opening gap, each adds -0.1 points. So, for example, the alignment in Figure 5 would yield a score of 8.2, since it has 10 matches, 1 mismatch, 1 opening gap and a gap extension of 3 gap symbols.

```
AGCUUUCACAGCUAA
AG----CAGAGCUAA
```

Figure 5: Global pairwise alignment of the sequences 'AGCUUUCACAGCUAA' and 'AGCAGAGCUAA'.

The reason behind this way of penalizing gaps is because of our focus on alternative transcripts. The genes' sequences can differ from each other by long subsequences because of evolutionary events such as exon gain, exon loss and exon duplication [17]. Because of this, it might be more suitable to encourage fewer but larger gaps, with several subsequent gap symbols, in the pairwise alignments rather than many seperate single gaps.

To aid in visualizing and in storing the alignment scores of a gene, the algorithm uses a representation of a matrix such as the one in Figure 6, where a column represents a sequence of the gene that the algorithm is currently calculating the scores for, and the rows of the column hold the scores of the alignments with the sequences of all of the other genes. Note that all the columns together represent the gene.

As an example, let $g_1 = \{s_{1g_1}, s_{2g_1}, s_{3g_1}, s_{4g_1}, s_{5g_1}\}$ be a gene of the input set G that we are currently calculating alignment scores for, in Figure 6 we see the alignment scores between g_1's sequences and all the other sequences in G.

	s_{1g_1}	s_{2g_1}	s_{3g_1}	s_{4g_1}	s_{5g_1}
s_{1g_2}	11	17	12	14	20
s_{2g_2}	21	13	22	15	10
...	...	...	...	...	...
s_{1g_3}	10	27	15	24	30
...	...	...	...	...	...

Figure 6: Score matrix for g_1's sequences.

After all the alignment scores have been calculated and stored in the matrix, the sum of the rows is calculated for each column. The column, which again represents a sequence, with the largest sum of rows, that is the largest sum of alignment scores, is selected as the gene's representative.

2.2 Testing and evaluation

2.2.1 SimSpliceEvol

The algorithm will be tested on simulated data provided by a program called *SimSpliceEvol* [11]. It simulates not only traditional gene sequence evolution events, such as insertion, deletion and substitution events, but also evolution events of the transcripts. The simulation of evolution of the transcripts is based on the *Christinat-Moret* model of transcript evolution, which is divided into two levels [17]. One acts on the exon-intron structure of the gene, that is which segments of the gene are exons or introns, and the other level acts on the sets of transcripts obtained from the exon-intron structure.

SimSpliceEvol takes as input a guide gene tree, such as the one in Figure 7, with branch lengths that represent the expected number of evolution events along the branches [11]. It then simulates a gene sequence and a set of alternative coding DNA sequences, which are the alternative transcripts of the gene, at the root of the tree. The simulation of both the evolution of the gene sequences and the evolution of the transcripts are applied conjointly along the branches until it produces the final gene sequences and their respective sets of alternative coding DNA sequences at the leaves of the tree. The algorithm outputs these alternative coding DNA sequences.

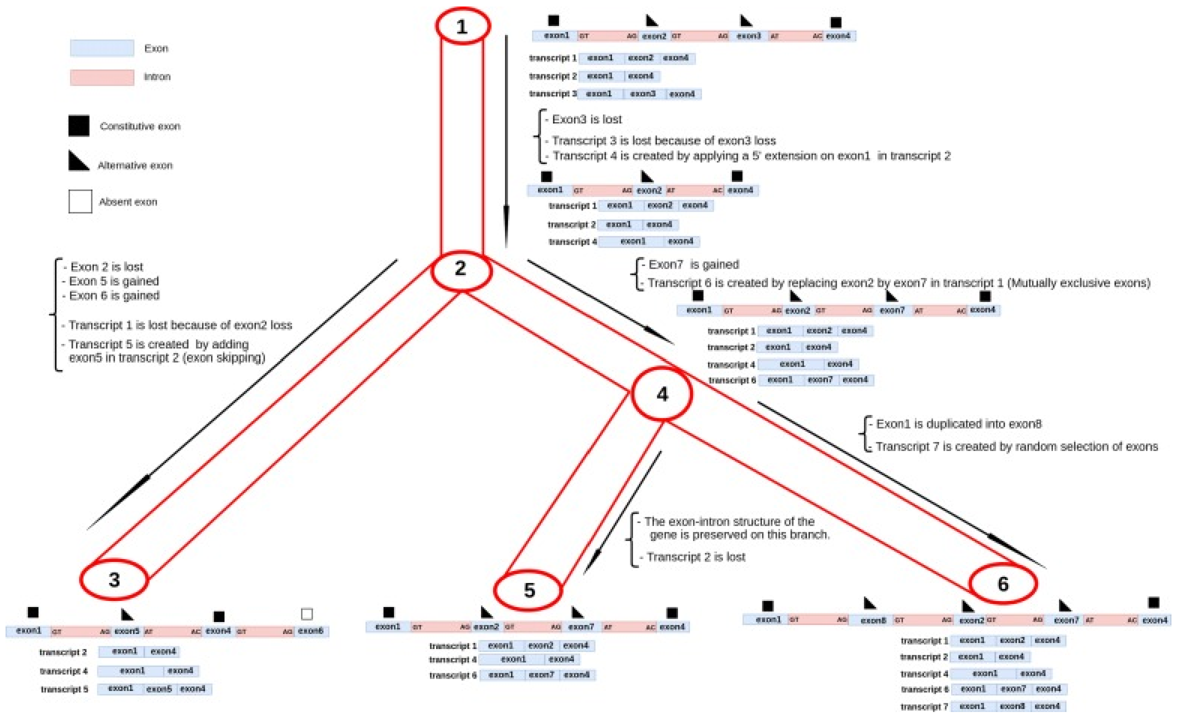
Exon
Intron
Constitutive exon
Alternative exon
Absent exon
1
2
3
4
5
6
exon1 GT AG exon2 GT AG exon3 AT AC exon4
transcript 1 exon1 exon2 exon4
transcript 2 exon1 exon4
transcript 3 exon1 exon3 exon4
- Exon3 is lost
- Transcript 3 is lost because of exon3 loss
- Transcript 4 is created by applying a 5' extension on exon1 in transcript 2
transcript 4 exon1 exon4
- Exon 2 is lost
- Exon 5 is gained
- Exon 6 is gained
- Transcript 1 is lost because of exon2 loss
- Transcript 5 is created by adding exon5 in transcript 2 (exon skipping)
- Exon7 is gained
- Transcript 6 is created by replacing exon2 by exon7 in transcript 1 (Mutually exclusive exons)
transcript 6 exon1 exon7 exon4
- Exon1 is duplicated into exon8
- Transcript 7 is created by random selection of exons
- The exon-intron structure of the gene is preserved on this branch.
- Transcript 2 is lost
transcript 5 exon1 exon5 exon4
transcript 7 exon1 exon8 exon4

Figure 7: An illustration of SimSpliceEvol's simulation of evolution events along the branches of the input guide tree. *Credits for this figure go to Esaie Kuitche, Safa Jammali, Aïda Ouangraoua and the National Center for Biotechnology Information* [11]. Creative Commons license.

The *SimSpliceEvol* program comes with three of its own already defined input guide trees [18]. They are the *small*, *medium* and *large* trees. The program will be used to create a number of simulations with a set of genes each, and the *small* tree will be used as input for the simulations. The branch lengths of the *small* tree are shorter than the *medium* and *large* ones, and using it as opposed to using the other two trees therefore results in the genes not evolving to be very distantly related. We choose to limit ourselves to using just this input tree.

Unfortunately, at the time of working on this project *SimSpliceEvol* did not work as described in the article [11]. Among other errors, the most significant was that it seemed to only output one coding DNA sequence per gene in every simulation [18]. To attempt to fix this a few modifications of the source code were done, which caused the program to now output five sequences per gene. Now, though the program may not be running properly or as the creators had probably intended, it at least produces several sequences per gene during a simulation, which we can work with. Both the original source code and the modified version of *SimSpliceEvol* are provided in this GitHub repository https://github.com/ram97-boop/Representative-Sequence.

For each simulated set of genes our algorithm will take, for each gene, the alternative coding DNA sequences as input and output a representative sequence. Two multiple sequence alignments will then be created, one with the representative sequences and the other with the longest sequences. The two multiple sequence alignments will then be scored using the sum-of-pairs method.

2.2.2 Sum of pairs

To calculate the sum-of-pairs score for a multiple sequence alignment each pair of sequences in it will be scored first, and the total score of the multiple sequence alignment will be the sum of all the sequence-pairs' scores [19]. Let us introduce some definitions before going through the process of calculating a sum-of-pairs score. Firstly is a definition of a multiple sequence alignment provided by [19].

Definition 2.1 Given k sequences, $S_1, S_2, ..., S_k$, a multiple sequence alignment (MSA) is obtained by inserting gaps in the strings to make them all the same length.

Definition 2.2 Given a sequence S of length m and an integer i such that $1 \leq i \leq m$, let $S[i]$ be the residue, or symbol, in S at the i-th position.

One way to implement or to define the calculation of the sum-of-pairs score is by looking at one column of the multiple sequence alignment at a time, from start to end, and summing the scores of all the pairs of rows in the column [19]. The sum-of-pairs score of the entire multiple sequence alignment will be the sum of all the column scores. Let us define a formula for calculating the score of a column.

Definition 2.3 Given two sequences S_1, S_2, that are in a multiple sequence alignment of length m, and given an integer i such that $1 \leq i \leq m$, let

$$score(S_1[i], S_2[i])$$

be the score of the pair of symbols $S_1[i]$ and $S_2[i]$ in the i-th column of the alignment.

Definition 2.4 Given a multiple sequence alignment of k sequences, $T_1, T_2, ..., T_k$, with length m, and given an integer i such that $1 \leq i \leq m$, let

$$SP(T_1[i], T_2[i], ..., T_k[i]) =$$

$$score(T_1[i], T_2[i]) + score(T_1[i], T_3[i]) + ... + score(T_{k-1}[i], T_k[i])$$

be the sum of the scores of all the pairs of symbols at the i-th column of the alignment.

Before scoring a column, we must assume score values for every possible pair of symbols that can exist in the alignment. For a pair of symbols a and b we choose to assume the following values:

$$score(a,b) = \begin{cases} 3 & a \text{ and } b \text{ match} \\ -1 & \text{indel} \\ -2 & a \text{ and } b \text{ mismatch} \\ 0 & a \text{ and } b \text{ gap symbols} \end{cases}$$

To clarify, a and b are a match only if they are both not gap symbols and are identical. They are a mismatch if they are both not gap symbols and are not identical. Furthermore, by indel we mean either only a or only b is a gap symbol.

Note that in a multiple sequence alignment there is no column entirely consisting of gaps. Hence, when scoring a pair of sequences we do not penalize a column of gap symbols and we ignore it to avoid extra penalization, since another sequence in the alignment will have a non-gap symbol in the same column [19]. See Figure 8 for an example of scoring a pair of sequences with our scoring function.

```
AGCTTTCACAGCTAA
AG----CAGAGCTAA
```

Figure 8: The score for this pair of sequences with our scoring function would be 24, since there are 10 matches, 1 mismatch and 4 indels.

3 Results

Using *SimSpliceEvol*, 500 simulations were run. In each of these simulations five genes were created with five alternative sequences each. For each gene in each simulation, we select a representative sequence using our algorithm and also extract the longest sequence. This resulted in two output files for each simulation, one consisting of the genes' representative sequences and the other of the genes' longest sequences.

In 179 out of the 500 simulations both of the output files were exactly the same, meaning that the algorithm selected the longest sequence as the representative for 5/5 genes. See Figure 9. In 209 simulations the longest sequence was selected as the representative for 4/5 genes. The same happened in 93 simulations for 3/5 genes, in 18 simulations for 2/5, and in the remaining one simulation for 1/5, which is difficult to see in Figure 9.

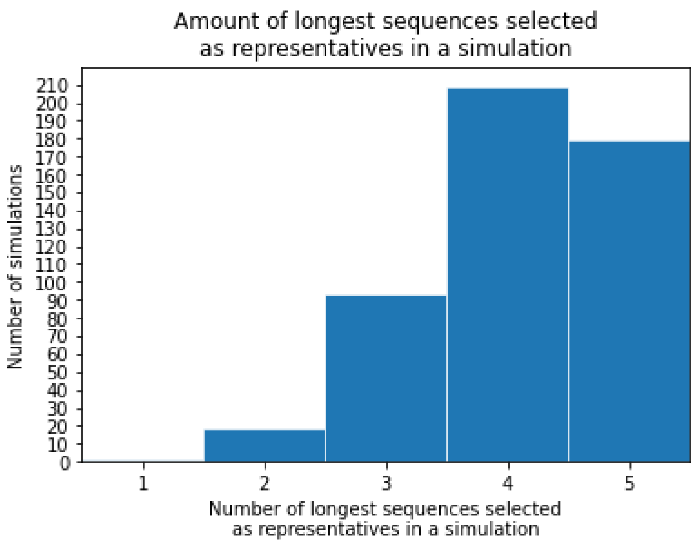

Figure 9: A bar chart of the number of simulations where the representative sequences are also the longest. The numbers on the x-axis represent the amount of longest sequences that have been selected as representatives in a simulation, so for example, the 5 on the x-axis means that all of the five genes' longest sequences are their representatives.

For comparing our method of selecting representatives with the method of selecting the longest sequences as representatives we ignore the 179 simulations where all the representative sequences were also the gene's longest and look at the remaining 321. For each of these 321 simulations two multiple sequence alignments were created, one for the representative sequences and the other for the longest sequences. The sum-of-pairs scores was then calculated for both alignments. The multiple sequence alignment of the representatives scored higher than the one of the longest sequences in 222 of the 321 simulations, which would be around 69.2 percent. The average difference in sum-of-pairs scores between the MSAs of the representatives and the longest sequences is approximately 5483.49 points for the 222 simulations where the representatives' MSAs scored higher, and approximately 2866.30 points for the remaining 99 simulations where the longest sequences' MSAs scored higher. The average difference in sequence length between the representatives and the longest sequences in the 222 simulations where the representatives' MSAs scored higher is approximately 115.08 residues, and in the remaining 99 simulations it is approximately 83.29 residues.

If we divide the simulations into cases based on the amount of longest sequences selected as representative sequences like we've done in Figure 9 then we'd get the charts shown in Figures 10, 11, 12 and 13. The charts in Figures 10 and 11 both show the average difference in sum-of-pairs score for the different amounts of longest sequences selected as representatives. Though, the first chart is exclusively for the 222 simulations where the representatives' MSAs scored higher while the second is for the other 99 simulations where the longest sequences' MSAs scored higher. The charts in Figures 12 and 13 show the average difference in length between the representative sequences and the longest sequences for the different amounts of longest sequences selected as representatives. Similarly, the first chart is exclusively for the 222 simulations where the representatives' MSAs scored higher while the second is for the other 99 simulations where they scored lower.

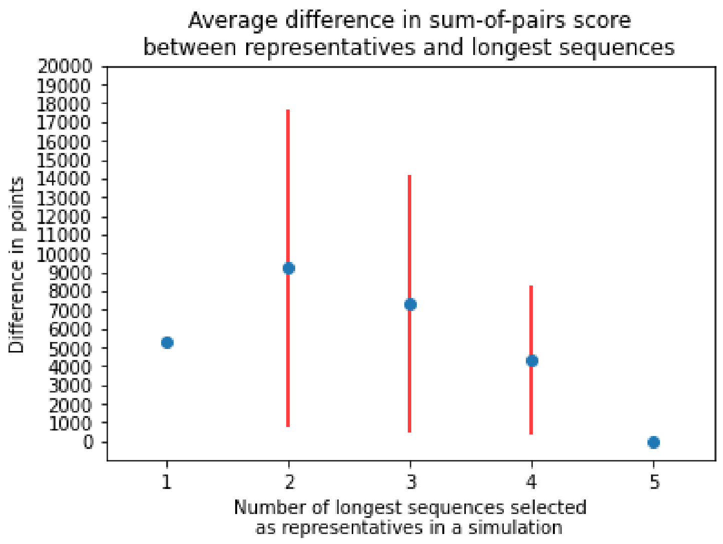

Figure 10: A bar chart of the average differences (with the standard deviations shown as red lines) in sum-of-pairs score between the MSAs of the representatives and the longest sequences for the different amounts of longest sequences selected as representatives in a simulation. Notice that in the first case the standard deviation is zero because there is only one simulation in that case and therefore only one difference in sum-of-pairs score. *Note that the averages shown here are for the 222 simulations where the representatives' MSAs scored higher than the longest sequences'.*

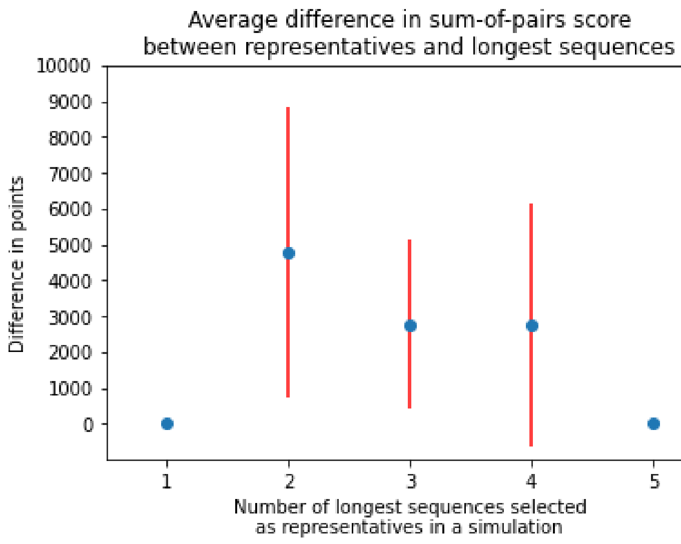

Figure 11: A bar chart of the average differences (with the standard deviations shown as red lines) in sum-of-pairs score between the MSAs of the representatives and the longest sequences for the different amounts of longest sequences selected as representatives in a simulation. *Note that the averages shown here are for the 99 simulations where the representatives' MSAs scored lower than the longest sequences'.*

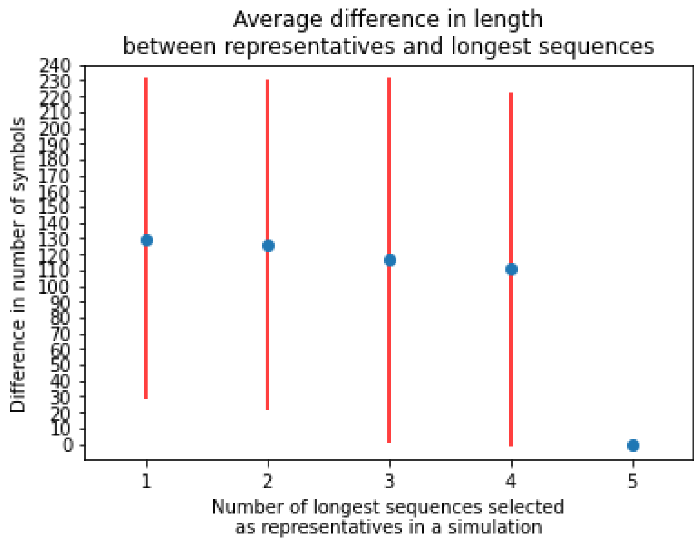

Figure 12: A bar chart of the average differences (with the standard deviations shown as red lines) in sequence length between the representatives and the longest sequences for the different amounts of longest sequences selected as representatives in a simulation. *Note that the averages shown here are for the 222 simulations where the representatives' MSAs scored higher than the longest sequences'.*

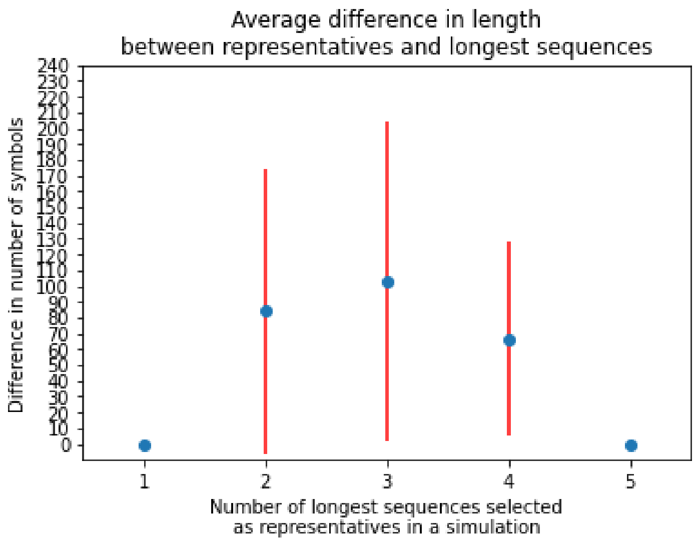

Figure 13: A bar chart of the average differences (with the standard deviations shown as red lines) in sequence length between the representatives and the longest sequences for the different amounts of longest sequences selected as representatives in a simulation. *Note that the averages shown here are for the 99 simulations where the representatives' MSAs scored lower than the longest sequences*'.

4 Discussion

From Figure 9 we see that the representative sequences selected by our algorithm tend to also be the longest sequences. In all of the 500 simulations there is at least one gene that had its longest sequence selected as its representative. In 388 out of the 500 simulations either four or five genes had their longest sequences selected as their representatives, which makes up 77.6 percent of the simulations. This could suggest that selecting the longest sequence as the representative is maybe not the worst choice most of the time. Especially if a retained intron is what causes a sequence to be the longest, since intron retention is the rarest form of alternative splicing [20]. On the other hand, the algorithm did select representatives whose MSAs scored higher than the longest sequences' MSAs for the majority of the simulations where at least one gene's longest sequence was not selected as its representative. It may also be notable that the average difference in scores were larger in those simulations than the average difference in the simulations where the longest sequences' MSAs scored higher, meaning that in these simulations, the scores of the representatives' MSAs were not far off from the longest sequences' MSAs'.

The algorithm seems likely to select representatives whose MSAs score equally to or higher than those of the longest sequences. However, the algorithm was tested on only coding DNA sequences. It would be interesting to see how it performs on protein sequences for example, since there are only four nucleotides in a DNA or RNA sequence and there are 28 amino acids that could make up a protein.

Caution has to be taken around the testing and evaluation of our algorithm. Firstly, it was tested on simulated data created using the *SimSpliceEvol* program. It does not simulate certain occurances, like e.g. the evolution of splice sites and motif conservation, i.e. the conservation of certain subsequences between diverged gene families [11]. It is therefore not certain that our algorithm will perform similarly with real data.

Secondly, the simulations were run using the *small* input guide tree provided in the *SimSpliceEvol* program, where its branch lengths were relatively short compared to the other trees. The resulting genes and their respective alternative transcripts were therefore more closely related. It is therefore uncertain if we would have similar results with different input trees.

Thirdly, the *SimSpliceEvol* program was initially not working properly and did not produce several alternative sequences per gene in a simulation like it was supposed to. While it at least produces several sequences per gene after some modifications to the source code, it probably is not doing so properly or as the creators, Kuitche, Jammali and Ouangraoua, had intended [11]. Most notable is that it now strictly produces five sequences per gene, which in turn can have affected the proper simulating of the evolutionary events occuring in them.

Fourthly, the algorithm selects a gene's representative based on the gene's sequences' similarity to the other genes' sequences in the input set. While it is shown that protein function between two proteins is more likely to be similar if their sequences are as well, there are exceptions for when this is not the case [12].

It should also be noted that the pairwise alignments in the algorithm may take a while depending on the number of genes, the number of their respective alternative sequences and their lengths in the input set. If the input set contains n genes with m alternative sequences each, then

$$\binom{n}{2} m^2 \tag{1}$$

pairwise alignments would have to be computed. For example, if $n = 4$ and $m = 5$ the five sequences of the first gene that is iterated upon will be pairwisely aligned with the sequences of the other three genes, which gives us $5(3 \times 5)$ alignments so far. Then, the second gene's five sequences will be aligned with the sequences of the remaining two genes, which gives us $5(2 \times 5)$ alignments more. Lastly, the third gene's sequences will be aligned with the last gene's sequences, which gives us $5(1 \times 5)$ more alignments. So for an input set of 4 genes with 5 sequences each there would be

$$5(3 \times 5) + 5(2 \times 5) + 5(1 \times 5)$$

$$= 5^2(1 + 2 + 3)$$

$$= 5^2 \times \frac{3(1+3)}{2}$$

$$= 5^2 \times \frac{(4-1)(1+4-1)}{2}$$

$$= 5^2 \times \frac{4^2 - 4}{2}$$

$$= m^2 \times \frac{n^2 - n}{2}$$

$$= \frac{n(n-1)}{2} \times m^2$$

$$= \frac{n!}{2!(n-2)!} \times m^2$$

$$= \binom{n}{2} m^2$$

alignments, which is equal to the previous expression (1).

For the computation of the pairwise alignments the *Bio.pairwise2* module uses a dynamic programming algorithm [15]. The documentation does not specifically mention which algorithm is implemented, but if we assume that it is the Needleman-Wunsch algorithm the time complexity for aligning two sequences with lengths p and q respectively would be $O(pq)$ [21].

5 Conclusion

Our algorithm seems to have selected representatives whose multiple sequence alignments had higher sum-of-pairs scores than those of the longest sequences. However, choosing the longest sequence as a representative may not be the worst choice most of the time since the algorithm selected the longest sequences as representatives for a large portion of the simulated data. Since the algorithm was tested on simulated data we cannot assume that it will yield similar results with real data. Furthermore, the simulated data were of genes that were closely related and the genes' alternative sequences were of coding DNA, so the testing of the algorithm was limited to these contraints. Further testing will have to be done to get a better picture of how the algorithm actually performs, especially with different data like more diverged genes for example, or with protein sequences instead of DNA or RNA etc. We might also get more accurate results with a larger set of data to test on instead of the 500 which we limited ourselves to. Lastly, because of the lengthy computation of the algorithm it may not be a viable option for one who would want to select representatives for hundreds of genes or more, each having several alternative sequences. Instead, this project may primarily provide a new perspective of selecting a gene's representative sequence with this approach if it has not already been explored.

6 References

[1] What is DNA? https://medlineplus.gov/genetics/understanding/basics/dna/. [Accessed 2022-03-29].

[2] Ribonucleic Acid (RNA). https://www.genome.gov/genetics-glossary/RNA-Ribonucleic-Acid. [Accessed 2022-03-29].

[3] What are Proteins and what do they do? https://medlineplus.gov/genetics/understanding/howgeneswork/protein/. [Accessed 2022-03-29].

[4] What is a Gene? https://medlineplus.gov/genetics/understanding/basics/gene/. [Accessed 2022-03-29].

[5] David M Hillis. Phylogenetic Analysis. *Current Biology*, 7(3):R129–R131, 1997. Elsevier.

[6] David W Mount. *Bioinformatics: Sequence and Genome Analysis*, chapter 1,3. 1 edition, 2001. Coldspring Harbor Laboratory Press.

[7] Manolis Kellis et al. Global alignment vs. Local alignment vs. Semi-global alignment. https://bio.libretexts.org/Bookshelves/Computational_Biology/Book%3A_Computational_Biology_-_Genomes_Networks_and_Evolution_(Kellis_et_al.)/03%3A_Rapid_Sequence_Alignment_and_Database_Search/3.03%3A_Global_alignment_vs._Local_alignment_vs._Semi-global_alignment. [Accessed 2021-11-10].

[8] Dannie Durand. Sequence Alignment. https://www.cs.cmu.edu/~durand/03-711/2015/Lectures/PW_sequence_alignment_2015.pdf, 2015. [Accessed 2021-11-15].

[9] Genome. https://www.nature.com/scitable/definition/genome-43/. [Accessed 2022-04-05].

[10] Intron. https://www.genome.gov/genetics-glossary/Intron. [Accessed 2022-04-05].

[11] Esaie Kuitche, Safa Jammali, and Aïda Ouangraoua. SimSpliceEvol: Alternative Splicing-aware Simulation of Biological Sequence Evolution. *BMC Bioinformatics*, 20(20):1–13, 2019. BioMed Central.

[12] Vineet Sangar, Daniel J Blankenberg, Naomi Altman, and Arthur M Lesk. Quantitative Sequence-function Relationships in Proteins Based on Gene Ontology. *BMC Bioinformatics*, 8(1):1–15, 2007. BioMed Central.

[13] Qun Pan, Ofer Shai, Leo J Lee, Brendan J Frey, and Benjamin J Blencowe. Deep Surveying of Alternative Splicing Complexity in the Human Transcriptome by High-throughput Sequencing. *Nature Genetics*, 40(12):1413–1415, 2008. Nature Publishing Group.

[14] NCBI RefSeq Select. https://www.ncbi.nlm.nih.gov/refseq/refseq_select/. [Accessed 2021-11-16].

[15] Bio.pairwise2 Module. https://biopython.org/docs/1.75/api/Bio.pairwise2.html. [Accessed 2021-10-28].

[16] Gap Penalty. https://www.bionity.com/en/encyclopedia/Gap_penalty.html. [Accessed 2022-02-19].

[17] Yann Christinat and Bernard ME Moret. Inferring Transcript Phylogenies. In *BMC Bioinformatics*, volume 13, pages 1–14, 2012. BioMed Central.

[18] SimSpliceEvol GitHub Repository. https://github.com/UdeS-CoBIUS/SimSpliceEvol. [Accessed 2022-02-28].

[19] Mona Singh. Multiple Sequnce Alignments 1 [Lecture Notes]. https://www.cs.princeton.edu/~mona/Lecture/msa1.pdf, 2000. Princeton University.

[20] Eddo Kim, Alon Magen, and Gil Ast. Different Levels of Alternative Splicing Among Eukaryotes. *Nucleic Acids Research*, 35(1):125–131, 2007. Oxford University Press.

[21] Sorin Istrail. Sequence Alignment and the Needleman-Wunsch Algorithm [Lecture Notes]. http://cs.brown.edu/courses/cs181/lectures/notes-alignment/19-9-17/notes.pdf, 2017. Brown University.