CSA-X: Modularized Constrained Multiple Sequence Alignment

T.M. Rezwanul Islam(✉) and Ian McQuillan

Department of Computer Science,
University of Saskatchewan, Saskatoon, SK, Canada
rezwanul.islam@usask.ca, mcquillan@cs.usask.ca

Abstract. Imposing constraints that influence multiple sequence alignment (MSA) algorithms can often produce more biologically meaningful alignments. In this paper, a modularized program of constrained multiple sequence alignment (CMSA) called CSA-X is created that accepts constraints in the form of regular expressions. It uses arbitrary underlying MSA programs to generate alignments, and is therefore modular. The accuracy of CSA-X with different underlying MSA algorithms is compared, and also with another CMSA program called RE-MuSiC that similarly uses regular expressions for constraints. A technique is also developed to test the accuracies of CMSA algorithms with regular expression constraints using the BAliBASE 3.0 benchmark database. For verification, ProbCons and T-Coffee are used as the underlying MSA programs in CSA-X, and the accuracy of the alignments are measured in terms of Q score and TC score. Based on the results presented herein, CSA-X significantly outperforms RE-MuSiC. On average, CSA-X used with constraints that were algorithmically created from the least conserved regions of the correct alignments achieves results that are 17.65% higher for Q score, and 23.7% higher for TC score compared to RE-MuSiC. Further, CSA-X with ProbCons (CSA-PC) achieves a higher score in over 97.9% of the cases for Q score, and over 96.4% of the cases for TC score. It also shows that the use of regular expression constraints, if chosen well, created from accurate knowledge regarding a lesser conserved region can improve alignment accuracy. Statistical significance is measured using the Wilcoxon rank-sum test and Wilcoxon signed-rank test. An open source implementation of CSA-X is also provided.

Keyword: Multiple sequence alignment

1 Introduction

Multiple sequence alignment (MSA) is a fundamental tool towards many objectives, such as phylogenetic studies, computational biology, prediction of functional residues, and protein structure prediction [17]. A large number of MSA programs have been developed, and Pais, FSM et al. [16] recently surveyed such programs in terms of accuracy and computational time. Indeed, accuracy is

D. Figueiredo et al. (Eds.): AlCoB 2017, LNBI 10252, pp. 143–154, 2017.
DOI: 10.1007/978-3-319-58163-7_10

particularly important for MSA algorithms, especially within modern computer bioinformatics pipelines, where less accurate alignments cause negative downstream effects with amplification of errors [12]. Most of the state-of-the-art multiple sequence alignment programs such as ProbCons [7], T-Coffee [15], MAFFT [11], and ClustalW [20] are fully automated, with a limited number of changeable parameters.

But often, users have additional information that could affect the alignments such as, active site residues, intramolecular disulphide bonds, enzyme activities, and conserved motifs [19]. Hence, having a program that can use additional information, either manually created, or automatically determined from additional annotations, can improve the accuracy of alignments. Constrained multiple sequence alignment (CMSA) [18] is an extension of the MSA problem [4] that allows users to use knowledge regarding the sequences involved, in the form of constraints, with a view to achieving more biologically meaningful alignments. For example, Du and Lin [8] showed that ClustalW [20], does not align common patterns and similar structures found in sequences consistently. Because of this reason, Tsai et al. [19] proposed MuSiC, a web server that allowed constrained alignment of sequences. But many biologically important motifs, such as those listed as regular expressions in the PROSITE [10] database cannot be formulated into constraints according to the convention followed by MuSiC [5]. To solve this issue, Arslan [2], and Chung et al. [6] introduced alignment algorithms that accept regular expression constraints, and enforce that segments that match the regular expression must align. Then, Chung et al. proposed RE-MuSiC [5], an extension to their previous work [6] to support multiple sequences and multiple constraints. In that work, they used sequence motifs found in PROSITE as regular expression constraints to improve the quality of alignments. However, there are some limitations of RE-MuSiC, as it does not allow the use of quantification operators such as Kleene star (*), Kleene plus (+) in regular expression constraints, and thus only a subset of regular expressions can be used as input. Arslan [3] also proposed sequence alignment programs guided by Context Free Grammars (CFG) only limited to pairwise sequence alignment. Morgenstern et al. [14] developed DIALIGN, a web server, that can accept user defined anchor points as constraints. It is common to use a benchmark database, such as BALiBASE to evaluate MSA algorithms [9,11,15], however no such technique is available for CMSA with regular expression constraints, and therefore, no such comparison is available for use with RE-MuSiC.

Here a new program, CSA-X is developed that also accepts arbitrary regular expression constraints (including quantifiers), and creates a multiple sequence alignment that forces sections to align that match the entire regular expression. Furthermore, it is also possible to enforce with an extended regular expression syntax that certain sections that match part of a regular expression must align. CSA-X is a modularized program that uses an underlying MSA program, and because of this reason, it is possible to replace the underlying MSA program with another, perhaps improved or tailored program. In addition, this study compares the performance of CSA-X, RE-MuSiC, and 'X', where 'X' is the

underlying MSA program in the proposed tool, with respect to the BALiBASE 3.0 [21] benchmark database. This involves the creation of a new technique to compare CMSA algorithms by creating regular expressions algorithmically using the BALiBASE alignments. This assesses their accuracy in terms of Q score and TC score [9], and measures statistical significance of the results. Furthermore, it also shows that if constraints are chosen appropriately, such as from knowledge regarding lesser conserved regions, CSA-X can give better results than the underlying MSA algorithm.

2 Methods

An open source implementation of CSA-X has been made available [1], which will be described. Arbitrary MSA implementation can be used with it. CSA-X accepts constraints in the form of regular expressions using the PERL regular expression syntax. However, the symbol # can be optionally placed in multiple spots in the regular expression, and it has special meaning providing guidance by which sequences are aligned. Next the constraints are defined.

Definition 1. *Hash-augmented regular expressions are defined inductively:*

- *every PERL regular expression is a hash-augmented regular expression,*
- *if R and S are two hash-augmented regular expressions, then $R\#S$ is a hash-augmented regular expression.*

From this definition, it is implied that every hash-augmented regular expression can be written in the following form, for some $n \geq 1$: $R_1\#R_2\#\ldots\#R_n$, when $R_1, \ldots, R_n$ are regular expressions. Intuitively, the MSA generated will align the parts of each sequence that match $R_1, R_2, \ldots, R_n$, and enforce that the parts that match each R_i, for $1 \leq i \leq n$ are aligned. Hence, the # symbols provide additional control by giving information regarding the residues or nucleotides to align. If the # symbols are not used, then CSA-X constructs the best alignment of the entire parts matching the entire regular expression.

In the case of hash-augmented regular expressions, between every two # symbols, it must be a syntactically correct regular expression. For example, `(AC#TT)C#A` is not a valid CSA-X hash-augmented regular expression because the left side of the first # symbol contains `'(AC'` and right side contains `'TT)C'` which are not regular expressions.

If a hash-augmented regular expression matches multiple sequences, then each match must have the same number of hash symbols since # symbols cannot (by definition) be placed inside any quantifier, such as * (which could match i times within one sequence, but j times within another, where $i \neq j$).

Consider, an input of N sequences to align $S_1, S_2, S_3, \ldots, S_N$, $N \geq 2$, and a hash-augmented regular expression $R = R_1\#R_2\#R_3\#\ldots\#R_m$, where $m \geq 1$. The precise process that CSA-X uses to generate such an alignment can be described using the following high-level steps:

1. CSA-X attempts to match R to each sequence S_i of the N input sequences, for each i, $1 \leq i \leq N$. If the regular expression matches exactly once in each sequence S_i, then CSA-X determines a list of positions (for each i, $1 \leq i \leq N$) $l_i^0, l_i^1, l_i^2, \ldots, l_i^m$, where $0 \leq l_i^0 \leq l_i^1 \leq l_i^2 \leq \ldots \leq l_i^m \leq |S_i|+1$, whereby regular expression R_j matches between positions l_i^{j-1} and $l_i^j - 1$, for each j, $1 \leq j \leq m$ (if $l_i^{j-1} = l_i^j$ then R_j matches the empty string). If CSA-X does not find any regular expression matches on the input sequences or it finds matches for a strict subset of sequences in the dataset, then it returns the alignment of the input dataset using the underlying MSA program without using the regular expression.
2. CSA-X generates alignments for each of the matched sections of the sequences using the underlying alignment algorithm. That is, it aligns the subwords $S_1(1, l_1^0 - 1), \ldots, S_N(1, l_N^0 - 1)$, then aligns subwords $S_1(l_1^{j-1}, l_1^j - 1), \ldots, S_N(l_N^{j-1}, l_N^j - 1)$ for every j, $1 \leq j \leq m$, and then aligns subwords $S_1(l_1^{m+1}, |S_1|), \ldots, S_N(l_N^{m+1}, |S_N|)$. Then it concatenates each of these alignments together in order. As the constraints in CSA-X are specified using a hash-augmented regular expression, it generates alignments by decoding information from the specified constraints.
3. If CSA-X finds multiple regular expression matches on a single sequence, then it generates all possible combinations of the matched-segment datasets by selecting each regular expression match of the sequence separately, and determines the alignment that has the highest sum-of-pairs score.

Intuitively, the formalism of step 2 means that CSA-X identifies the segments that match the entire expression R for each sequence S_i in step 1. In addition, at the same time on each of the matched segments, it also identifies the subsections that match the sub-patterns $R_1, R_2, R_3, \ldots, R_m$ consecutively in the hash-augmented regular expression. Then, CSA-X aligns each matching sub-pattern separately (including the parts that match before the first matching sub-pattern, and the parts that match after the final sub-pattern has ended), using the underlying MSA program X to generate alignments, and then it merges the generated alignments together to produce a complete alignment.

In step 3, suppose a hash-augmented regular expression R, matches the sequence S_t at two spots. If the rest of the sequences match the hash-augmented regular expression exactly at one spot, then CSA-X would create two alignments, one where the matching occurs between the first matching part of S_t, and the other one where the matching occurs with the second matching part of S_t. Then the algorithm determines the alignment that has the highest sum-of-pairs score, and returns the alignment with the highest score.

It should be noted that, multiple regular expressions R and S can be used as input by joining them with quantifiers such as $R.^*S$, where "." represents any character match, and "*" is Kleene star. Alignment can be further influenced through # symbols, e.g. $R\#.^*\#S$.

Example 2. Conserved motifs for different protein sequences are listed in the PROSITE database in the form of regular expressions, which can be used as

constraints to improve the biological accuracy of the alignments in different CMSA programs. For example, the TATA-binding protein plays a vital role in the activation of eukaryotic genes. PROSITE (PDOC00303) lists the consensus for the signature pattern of the TATA-binding protein as follows (using a slightly different regular expression syntax).

Y-x-[PK]-x(2)-[IF]-x(2)-[LIVM](2)-x-[KRH]-x(3)-P-[RKQ]-x(3)-L-[LIVM]-F-x-[STN]-G-[KR]-[LIVMA]-x(3)-G-[TAGL]-[KR]-x(7)-[AGCS]-x(7)-[LIVMF].

For the alignment of different TATA box proteins, the above mentioned consensus pattern can be used as a constraint. For instance, if one would like to align TATA box proteins found in *Homo sapiens* (gb AAI09054.1), *Rattus rattus* (gb AAH16476.1), and microorganism *Halobacterium salinarum* (emb CAA63691.1) using CSA-X, then the format of this consensus pattern can be as follows:

Y.[PK]..[IF]..[LIVM]{2}.[KRH]...P[RKQ]...L[LIVM]F.[STN]G[KR][LIVMA]...G[TAGL][KR].....[AGCS].....[LIVMF].

It is also possible to simplify this further using quantifiers, or to add hash symbols to force sections of the regular expression to align; for instance to align sections of the sequences that match `Y.[PK]..[IF]..[LIVM]+.[KRH]`, then `...P[RKQ]...L[LIVM]F.[STN]G[KR][LIVMA]`, then `...G[TAGL][KR].....[AGCS]..... [LIVMF]` the hash augmented regular expression could be used. Omitting the # symbols would not necessarily align the three parts separately on all sequences.

Y.[PK]..[IF]..[LIVM]+.[KRH]#...P[RKQ]...L[LIVM]F.[STN]G[KR][LIVMA]#...G[TAGL][KR].....[AGCS].....[LIVMF].

Figure 1 shows the partial alignment generated by CSA-X, where the region identified by the regular expression is aligned in columns (highlighted). For this alignment, ProbCons is used as the underlying alignment tool.

3 Method of Assessments

Since CSA-X is a modular tool, the underlying MSA program can be changed to obtain different alignments, and in some sense, different customized tools. The study conducted by Pais, FSM et al. [16] showed that ProbCons, T-Coffee, Probalign, and MAFFT achieve higher accuracy than other MSA tools considered. Therefore, for this assessment, ProbCons and T-Coffee are used as the underlying MSA algorithms in CSA-X (although other programs can be used with CSA-X as well, these are the only two used for the purposes of assessment). Whenever CSA-X uses ProbCons, it is referred to as CSA-PC, and for T-Coffee, it is called CSA-TCOF. It is common for a benchmark database, such as BALiBASE 3.0 to be used to assess alignments and algorithms. Each set of sequences in such a database also has an alignment that is thought of as "correct" (based on additional knowledge such as protein structure). This database is

```
           201                                                  250
gi80478871 ........................YNPKRFAAVIMRIREPRTTALIFSS
gi16741283 MRIREPRTTALIFSSGKMVCTGAKSYEPELFPGLIYRMIKPRIVLLIFVS
gi1070345  ........................YNPEDFPGVVYRLQEPKSATLIFRS

           251                                                  300
gi80478871 GKMVCTGAKSEEQSRLAARKYARVVQKLGFPAK.FLDFKIQNMVGSCDVK
gi16741283 GKVVLTGAKVRAEIYEAFENIYPIL.........................
gi1070345  GKVVCTGAKSVDDVHEALGIVFGDIRELGIDVTSNPPIEVQNIVSSASLE

           301                                                  350
gi80478871 FPIRLEGLVLTH.QQFSSYEPELFPGLIYRMIKPRIVLLIFVSGKVVLTG
gi16741283 ..................................................
gi1070345  QSLNLNAIAIGLGLEQIEYEPEQFPGLVYRLDDPDVVVLLFGSGKLVITG
```

Fig. 1. CSA-X partial alignment of TATA box proteins. CSA-X partial alignment of TATA box proteins, where the highlighted regions indicate the sections matched by the regular expression constraint.

used here. However, RE-MuSiC generates erroneous alignments for a portion of the datasets in the BAliBASE 3.0 benchmark database, where the length of the sequences are not equal (sometimes the resulting alignments produced contain wildcard characters). Hence, the *working database* for this study is defined as being created from BAliBASE 3.0 including those datasets for which RE-MuSiC produces non-erroneous alignments for the purposes of comparison. BAliBASE 3.0 is classified into several groups; namely RV11, RV12, RV20, RV30, RV40, and RV50. In the working database for this study, there are 76 datasets from RV11, 84 datasets from RV12, 6 datasets from RV20, 6 datasets from RV30, 17 datasets from RV40 and 11 datasets from RV50; in total 200 datasets from BAliBASE 3.0 out of a total of 386 datasets. Out of these 200 datasets, 98 datasets contain short truncated sequences. To compare the performance of CSA-X with RE-MuSiC and other programs, this working database is used in this study (we will additionally consider the difference in results between programs when including those datasets not in the working dataset).

As BALiBASE does not contain regular expression constraints, a new technique must be developed for comparison of CMSA algorithms. To identify the effects of constraints on generated alignments, two sets of regular expression constraints are created to use for assessment. Indeed, the correct alignments from the BAliBASE 3.0 benchmark database are used to algorithmically create accurate regular expressions. One set of regular expression constraints are created from the "most conserved" region of the correct alignments. Another set is constructed from the "least conserved" region of the correct alignments. All of these constraints are automatically generated using a Perl script, which uses reference alignment files from BAliBASE 3.0, and identifies the most conserved regions and the least conserved regions for the alignments and generates the

regular expression constraints. These same sets of constraints are used to compare CSA-X and RE-MuSiC.

The idea behind this approach is to identify the effects of constraints on multiple sequence alignment. Often, expert users possess information about the sequences involved in the alignment process. They align the sequences using a MSA program, and then often adjust the alignment based on knowledge not reflected in the generated alignment. For the assessment, we do not obtain regular expression constraints from expert users, but rather, algorithmically create regular expressions from the curated alignments. Although this approach could create unrealistically good regular expressions, it is useful to use when comparing multiple algorithms that take regular expressions as constraints and to represent "accurate" knowledge being incorporated into regular expressions.

For this study, the regular expression constraints are generated to be of length 12 with a maximum of one gap per sequence, which is large enough to affect alignments, while avoiding many matches. To make a fair comparison between CSA-X and RE-MuSiC, both are tested on the same sets separately for both (most and least conserved) of regular expression constraints, and therefore, all regular expressions tested do not have the quantifiers * or + as these do not work with RE-MuSiC. Furthermore, a separate comparison is made between CSA-PC with these regular expressions and ProbCons without using any regular expressions at all (and similarly with T-Coffee) to gauge the potential improvements that using regular expressions as constraints can provide. This depends on whether the regular expressions are created from highly conserved or lesser conserved regions. Although this part of the assessment is done using the correct alignments to construct the regular expressions, it is only being used to see if regular expressions can possibly improve quality, depending on the type of regular expression. A thorough test of common regular expressions used by expert users together with a test to see if they improve alignment quality would be valuable. However, for comparing RE-MuSiC to CSA-X, such regular expressions are equally favourable to both programs, and is therefore a useful method of comparison.

3.1 Accuracy, Statistical Significance and Parameters

To measure the accuracy of considered programs in this study, two scores, Q score (Quality Score) and TC score (Total Column Score) are computed. Edgar [9] defined the Q score of an algorithm as a ratio between the number of correctly aligned pairs to the number of residue pairs in the reference alignment. This is the same as the sum-of-pairs score defined by Thompson et al. [22]. TC score is the number of correctly aligned columns, divided by the number of columns in the reference alignment (this is the same as the column score (CS) defined by Thompson et al.).

To reduce the probability that the difference is merely by chance, researchers working in the area of MSA frequently conduct statistical significance tests. In this work, Wilcoxon signed-rank test [23] and Wilcoxon rank-sum test [23] are used to measure statistical significance. If two samples are paired, Wilcoxon signed-rank test is used, otherwise, Wilcoxon rank-sum test is used.

Standalone ProbCons and T-Coffee are used with the default parameter settings (performing a comparison by systematically varying all parameters with every program is of interest but is left for future work). The same parameter settings of ProbCons and T-Coffee are used in CSA-PC and CSA-TCOF respectively. RE-MuSiC is run with the default gap extension and gap open penalty. CSA-PC, CSA-TCOF, and RE-MuSiC are provided with the equivalent set of regular expression constraints. As the format of specifying regular expression constraints in CSA-X and RE-MuSiC is different, equivalent regular expression constraint sets are used for these programs.

4 Results

For each of CSA-PC, CSA-TCOF, RE-MuSiC, T-Coffee, and ProbCons, average (AVG) and standard deviation (SD) of Q score and TC score are presented in Table 1. Among these programs CSA-PC, CSA-TCOF, and RE-MuSiC are provided with the regular expression constraints; however, T-Coffee and ProbCons are used without any constraints (as they do not take any as input).

4.1 Comparison of CSA-X with RE-MuSiC

It is observed from Table 1 that for the 200 datasets in the working database, that CSA-PC and CSA-TCOF both achieve higher accuracy compared to RE-MuSiC, using both Q score and TC score. From Table 1 it can be calculated that

Table 1. Average and standard deviation of Q score and TC score for the working database. MC (and LC respectively) represent the use of regular expression constraints identified from the most conserved region (least conserved respectively), of the correct alignments in the benchmark datasets ('−' represents a score that cannot be computed). The entries that are bold represent the highest value for each type of score and regular expression.

		MC	LC	AVG (SD)
		AVG (SD)	AVG (SD)	
Q	CSA-PC	**0.868** (**0.118**)	**0.881** (**0.116**)	–
	CSA-TCOF	0.860 (0.131)	0.876 (0.124)	–
	RE-MuSiC	0.691 (0.197)	0.702 (0.220)	–
	ProbCons	–	–	0.854 (0.153)
	T-Coffee	–	–	0.846 (0.166)
TC	CSA-PC	**0.713**(**0.222**)	**0.730** (**0.244**)	–
	CSA-TCOF	0.702 (0.231)	0.718 (0.244)	–
	RE-MuSiC	0.496 (0.256)	0.487 (0.299)	–
	ProbCons	–	–	0.693
	T-Coffee	–	–	0.680

on average for Q score, CSA-PC achieves approximately 0.179 and CSA-TCOF achieves almost 0.174 higher score compared to RE-MuSiC when using constraints obtained from the least conserved (LC) region of the correct alignments respectively. For the constraints obtained from the most conserved (MC) region of the correct alignments, CSA-PC and CSA-TCOF achieve 0.176 and 0.168 higher score respectively. While for TC score, for the most conserved region regular expression constraints set, CSA-PC achieves 0.217 and CSA-TCOF achieves 0.206 higher results compared to RE-MuSiC, and the score rises by 0.217 and 0.206 respectively for CSA-PC and CSA-TCOF for the LC constraints set.

Out of 200 working datasets for Q score, CSA-PC and CSA-TCOF with LC constraints perform higher for 195 and 194 datasets respectively compared to RE-MuSiC. In addition, for TC score, CSA-PC and CSA-TCOF with LC constraints set achieves higher score in total for 185 and 184 datasets. CSA-PC and CSA-TCOF with MC constraints set achieves a higher score for Q score for 192 and 191 datasets respectively, and for TC score they achieve higher score for 186 and 180 datasets respectively compared to RE-MuSiC. In addition, if all the datasets in BAliBASE 3.0 are considered, instead of just the working datasets, and we define CSA-X as performing better for instances where RE-MuSiC is giving erroneous results, then CSA-PC (LC) gives a higher score for 381 datasets out of 386 datasets, and CSA-TCOF (LC) gives a higher score for 380 datasets out of 386 datasets.

4.2 Comparison of CSA-X with the Underlying MSA Program

From Table 1, CSA-PC and CSA-TCOF score higher overall than standalone ProbCons and T-Coffee respectively run without any constraints. For MC constraints, CSA-PC and CSA-TCOF both show 0.014 higher Q score and more than 0.02 higher TC score compared to ProbCons and T-Coffee. Further, using LC constraints, CSA-PC and CSA-TCOF achieve 0.026 and 0.029 higher Q score and 0.0378 and 0.0376 higher TC score respectively. According to Thompson et al. [22] the BAliBASE sum-of-pairs score (similar to Q score) increases if a program succeeds in aligning sequences relative to the reference alignment dataset; this means that the higher the Q score is, the better the program is at generating accurate alignments, while TC score tests how efficiently the program is aligning all the sequences. This is a more stringent criteria of measurement as a column score can become zero if a single sequence is misaligned [13]. Again, it is worth mentioning that the comparison of CSA-X to its underlying tool does not lend any evidence to the notion that CSA-X is better than its underlying algorithm. It is only examining certain types of correct information that can improve alignments. Furthermore, it is an important verification of the potential of CSA-X.

4.3 Statistical Analysis

First, the Wilcoxon rank-sum test is performed between CSA-TCOF and RE-MuSiC, and between CSA-PC and RE-MuSiC. As the outcome of CSA-TCOF

does not depend upon RE-MuSiC, the Wilcoxon rank-sum test is chosen. Second, each are compared to their underlying algorithm with both the most conserved and least conserved regular expressions. Since the outcome of CSA-TCOF and CSA-PC depends upon T-Coffee and ProbCons respectively, the Wilcoxon signed-rank test is selected to test if there is significant difference between these programs. Table 2 shows the results of these tests. For both the tests, the null hypothesis is that there is no significant difference between the two samples. If the test rejects the null hypothesis then it means that there is a significant difference between the two samples. For this test, a 5% significance level is used, which means that if the p-value is less than 0.05 then the null hypothesis is rejected. For the Wilcoxon rank-sum test, all the p-values are significantly less than 0.05. Hence, the null hypothesis is rejected, and it is determined that the results of CSA-PC and CSA-TCOF are significantly different compared to RE-MuSiC, and the differences are not by chance. However, for the Wilcoxon signed-rank test, the results are not significantly different for CSA-TCOF and T-Coffee if CSA-TCOF uses the most conserved (MC) regular expression constraints set, as with CSA-PC. This is because ProbCons and T-Coffee both are able to successfully align the most conserved region without the explicit constraints. But the situation changes if CSA-TCOF and CSA-PC uses the least conserved (LC)

Table 2. Wilcoxon rank-sum and Wilcoxon signed-rank test results.

Wilcoxon rank-sum test			
Constraints	Programs	Scores	P-value (Significant)
MC	CSA-TCOF and RE-MuSiC	Q	<2.2e-16 (yes)
		TC	2.89e-15 (yes)
	CSA-PC and RE-MuSiC	Q	<2.2e-16 (yes)
		TC	<2.2e-16 (yes)
LC	CSA-TCOF and RE-MuSiC	Q	<2.2e-16 (yes)
		TC	6.66e-16 (yes)
	CSA-PC and RE-MuSiC	Q	<2.2e-16 (yes)
		TC	<2.2e-16 (yes)
Wilcoxon signed-rank test			
Constraints	Programs	Scores	P-value (Significant)
MC	CSA-TCOF and T-Coffee	Q	<0.6529 (no)
		TC	0.1579 (no)
	CSA-PC and ProbCons	Q	<0.0911 (no)
		TC	0.0201 (yes)
LC	CSA-TCOF and T-Coffee	Q	<8.18e-10 (yes)
		TC	3.09e-08 (yes)
	CSA-PC and ProbCons	Q	<2.50e-10 (yes)
		TC	<1.10e-08 (yes)

regular expression constraints set, and it is observed that there is significant difference in the results of CSA-TCOF and CSA-PC with T-Coffee and ProbCons respectively if they are supplied with LC constraints set.

Although, constraints chosen from the most or least conserved region are not necessarily realistic in terms of regular expression constraints chosen by either an expert user, or created from a database of additional information, using constraints created from the correct alignment does have the advantage of capturing some piece of information the user may know to be true, in a situation where a standalone alignment program is not giving the desired results. And indeed, constraints chosen from the most conserved region do not seem to help significantly versus not using any constraint, however constraints chosen from the least conserved region do help versus not using any constraints.

5 Conclusions

The constrained multiple sequence alignment program, CSA-X, allows the user to specify regular expression constraints for the multiple sequence alignment, and the resulting alignment enforces that specific sections matching the regular expression gets aligned. This can improve the accuracy and biological significance of the generated alignments, as functional and structural information regarding the sequences can be expressed using regular expression syntax. However, a more systematic study of regular expression constraints from expert users and other sources is left as future work.

In this research work, based on the average accuracy scores from the benchmarking analysis and the statistical significance testing, it is shown that CSA-X framework with ProbCons and T-Coffee (known as CSA-PC and CSA-TCOF respectively) generates more accurate alignments compared to RE-MuSiC—the only other implemented CMSA algorithm that uses regular expression constraints. Furthermore, it is also shown that if good regular expression constraints are chosen from the least conserved portion of the correct alignments, then the results of CSA-X are significantly better than the underlying MSA program. Finally, CSA-X is a modularized tool, and it allows the user to change the underlying multiple sequence alignment program if more efficient programs become available, or a specialized program is required. An open source implementation is also available [1].

References

1. CSA-X. https://bitbucket.org/RezwanIslam/csa-x/downloads. Accessed 28 Jan 2017
2. Arslan, A.N.: Regular expression constrained sequence alignment. J. Discrete Algorithms **5**(4), 647–661 (2007)
3. Arslan, A.N.: Sequence alignment guided by common motifs described by context free grammars. In: Biotechnology and Bioinformatics Symposium (BIOT) (2007)
4. Carrillo, H., Lipman, D.: The multiple sequence alignment problem in biology. SIAM J. Appl. Math. **48**(5), 1073–1082 (1988)

5. Chung, Y.S., Lee, W.H., Tang, C.Y., Lu, C.L.: RE-MuSiC: A tool for multiple sequence alignment with regular expression constraints. Nucleic Acids Res. **35**(suppl 2), W639–W644 (2007)
6. Chung, Y.-S., Lu, C.L., Tang, C.Y.: Efficient algorithms for regular expression constrained sequence alignment. In: Lewenstein, M., Valiente, G. (eds.) CPM 2006. LNCS, vol. 4009, pp. 389–400. Springer, Heidelberg (2006). doi:10.1007/11780441_35
7. Do, C.B., Mahabhashyam, M.S., Brudno, M., Batzoglou, S.: ProbCons: probabilistic consistency-based multiple sequence alignment. Genome Res. **15**(2), 330–340 (2005)
8. Du, Z., Lin, F.: Pattern-constrained multiple polypeptide sequence alignment. Comput. Biol. Chem. **29**(4), 303–307 (2005)
9. Edgar, R.C.: MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Res. **32**(5), 1792–1797 (2004)
10. Hulo, N., Bairoch, A., Bulliard, V., Cerutti, L., De Castro, E., Langendijk-Genevaux, P.S., Pagni, M., Sigrist, C.J.: The PROSITE database. Nucleic Acids Res. **34**(suppl 1), D227–D230 (2006)
11. Katoh, K., Misawa, K., Kuma, K., Miyata, T.: MAFFT: a novel method for rapid multiple sequence alignment based on fast fourier transform. Nucleic Acids Res. **30**(14), 3059–3066 (2002)
12. Kumar, S., Filipski, A.: Multiple sequence alignment: in pursuit of homologous DNA positions. Genome Res. **17**(2), 127–135 (2007)
13. Lassmann, T., Sonnhammer, E.L.: Kalign — an accurate and fast multiple sequence alignment algorithm. BMC Bioinformatics **6**(1), 298 (2005)
14. Morgenstern, B., Werner, N., Prohaska, S.J., Steinkamp, R., Schneider, I., Subramanian, A.R., Stadler, P.F., Weyer-Menkhoff, J.: Multiple sequence alignment with user-defined constraints at GOBICS. Bioinformatics **21**(7), 1271–1273 (2005)
15. Notredame, C., Higgins, D.G., Heringa, J.: T-Coffee: a novel method for fast and accurate multiple sequence alignment. J. Mol. Biol. **302**(1), 205–217 (2000)
16. Pais, F.S.-M., de Ruy, P., Oliveira, G., Coimbra, R.S.: Assessing the efficiency of multiple sequence alignment programs. Algorithms Mol. Biol. **9**(1), 1–4 (2014). BioMed Central
17. Papadopoulos, J.S., Agarwala, R.: COBALT: constraint-based alignment tool for multiple protein sequences. Bioinformatics **23**(9), 1073–1079 (2007)
18. Tang, C.Y., Lu, C.L., Chang, M.D.T., Tsai, Y.T., Sun, Y.J., Chao, K.M., Chang, J.M., Chiou, Y.H., Wu, C.M., Chang, H.T., Chou, W.I.: Constrained multiple sequence alignment tool development and its application to RNase family alignment. J. Bioinform. Comput. Biol. **1**(02), 267–287 (2003)
19. Te Tsai, Y., Huang, Y.P., Yu, C.T., Lu, C.L.: MuSiC: a tool for multiple sequence alignment with constraints. Bioinformatics **20**(14), 2309–2311 (2004)
20. Thompson, J.D., Higgins, D.G., Gibson, T.J.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Res. **22**(22), 4673–4680 (1994)
21. Thompson, J.D., Koehl, P., Ripp, R., Poch, O.: BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark. Proteins: Struct., Funct., Bioinf. **61**(1), 127–136 (2005)
22. Thompson, J.D., Plewniak, F., Poch, O.: A comprehensive comparison of multiple sequence alignment programs. Nucleic Acids Res. **27**(13), 2682–2690 (1999)
23. Triola, M.M., Triola, M.F.: Biostatistics for the Biological and Health Sciences. Pearson Addison-Wesley, Boston (2006)