Cloud-POA: A Cloud-Based Map Only Implementation of PO-MSA on Amazon Multi-node EC2 Hadoop Cluster

Paper Id- 284, ICCIT-2017, Dhaka, Bangladesh

Nafis Neehal, Dewan Ziaul Karim, Ashraful Islam

Lecturer, Department of CSE,
Daffodil International University,
Dhaka, Bangladesh.

Celebrating 125 Years
of Engineering the Future

Outline

Overview

- MSA Algorithm implemented – Partial Order based Multiple Sequence Alignment (PO-MSA)[1]

- Platform – Implemented on Amazon EC2 Multi-node Hadoop Cluster (10 nodes)

- Framework – Mapreduce (Map only job)

- Contribution- Better performance (runtime), Scalable and Distributed, Generate consensus sequence[2]

Problem Statement

- Sequence Alignment
    - Seq1 = AAGGT, Seq2 = AGG
    - Match = +2, Mismatch = -1, Gap = -2

A_1	A_2	A_3
AAGGT AGG_ _	AAGGT _AGG_	AAGGT _ _AGG
Score:-1	Score:2	Score:-4

Problem Statement (Cont.)

Pairwise Alignment	Progressive MSA	Iterative MSA
- No evolutionary relationship - No phylogenetic tree - Dynamic Approach increases time complexity	- No error correction - Information loss (up to 28%)[3]	- Iteration limited to a few number of times - Increases time complexity but also increases accuracy up to 5-10%[3]

TARGET - Optimize any MSA Algorithm to gain the optimal performance (Reduced Runtime + Increased Accuracy)

Literature Review

Typical MSA in terms of Quality

- MSA can reduce to consensus (1D Profile), but not vice-versa
- Information Loss Happens
- Degeneracy in MSA to Consensus mapping
- Degeneracy in representation of G/I format in RC-MCA

Typical MSA in terms of Performance

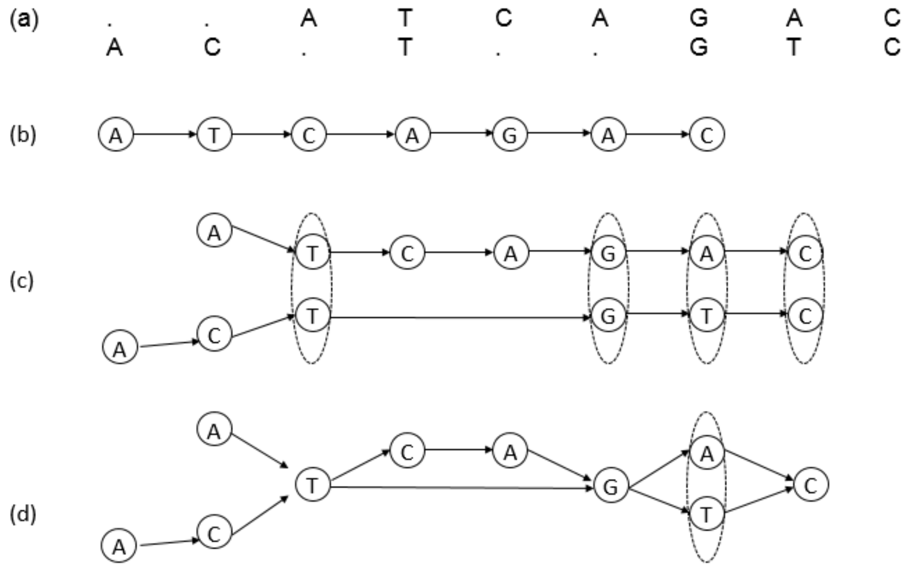
- Most Popular MSAs are ClustalW, Dialign-TX, MAFFT, MUSCLE, POA, Probalign, Probcons, T-Coffee[4]
- Probcons, T-Coffee, Probalign, MAFFT were more accurate, but consumed huge amount of time and memory
- ClustalW, Dialign and MUSCLE were faster and less memory consuming, but less accurate

Why Choose PO-MSA?

PO-MSA in terms of Quality	PO-MSA in terms of Performance
- Creates a unique MSA-Consensus mapping so that MSA can be reconstructed from Consensus - Eliminates degeneracy of MSA	- POA runtime is moderate, accuracy is excellent - Runtime can be decreased radically by a parallel implementation - Has a scalable structure in order to deploy for aligning large number sequences

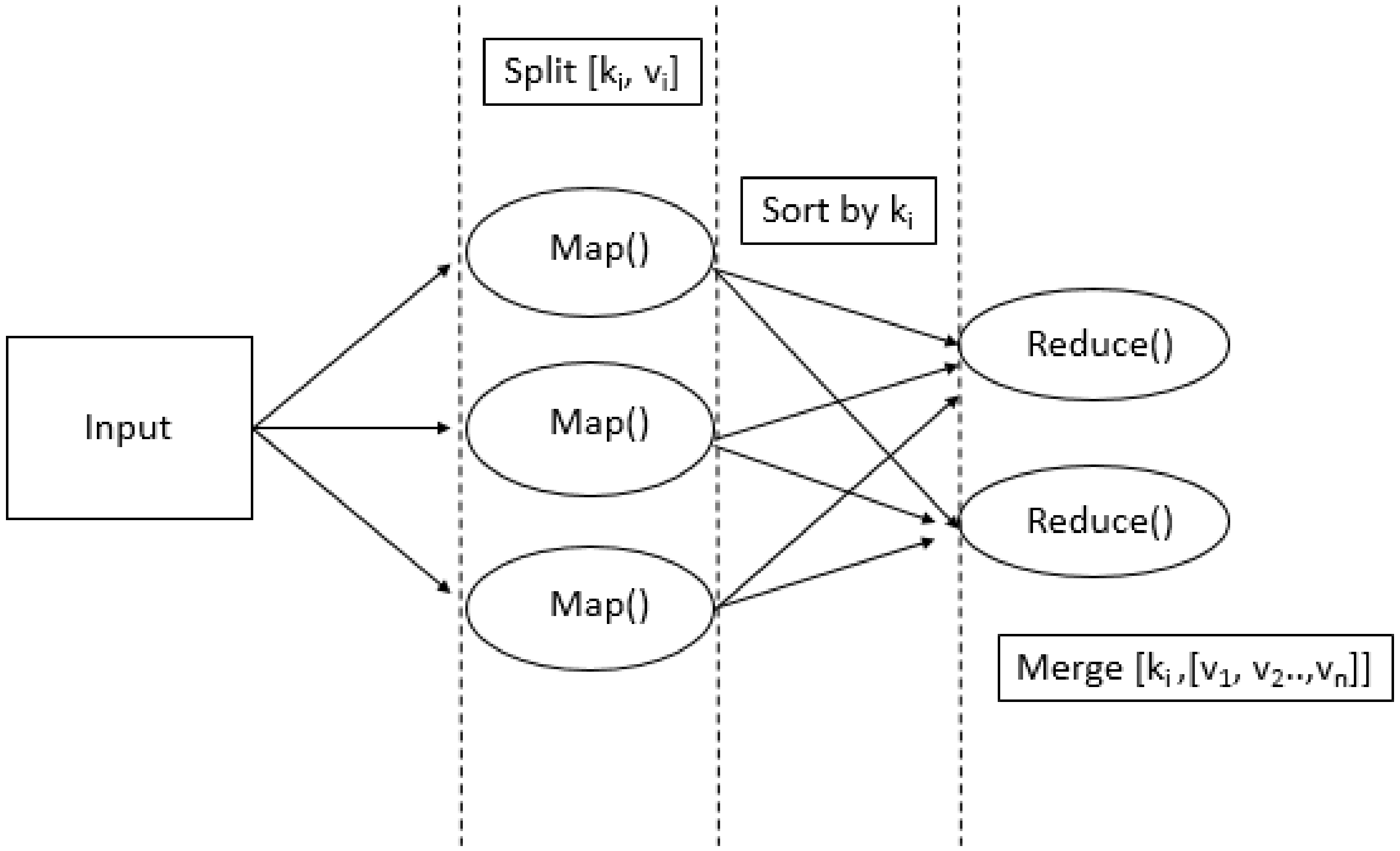
PO-MSA Structure

- Align new sequence to a DAG
- Multiple successors/ predecessors
- Align using all previous scores
- Insert node if not aligned with anything
- Insert aligned sequence and process new sequence again

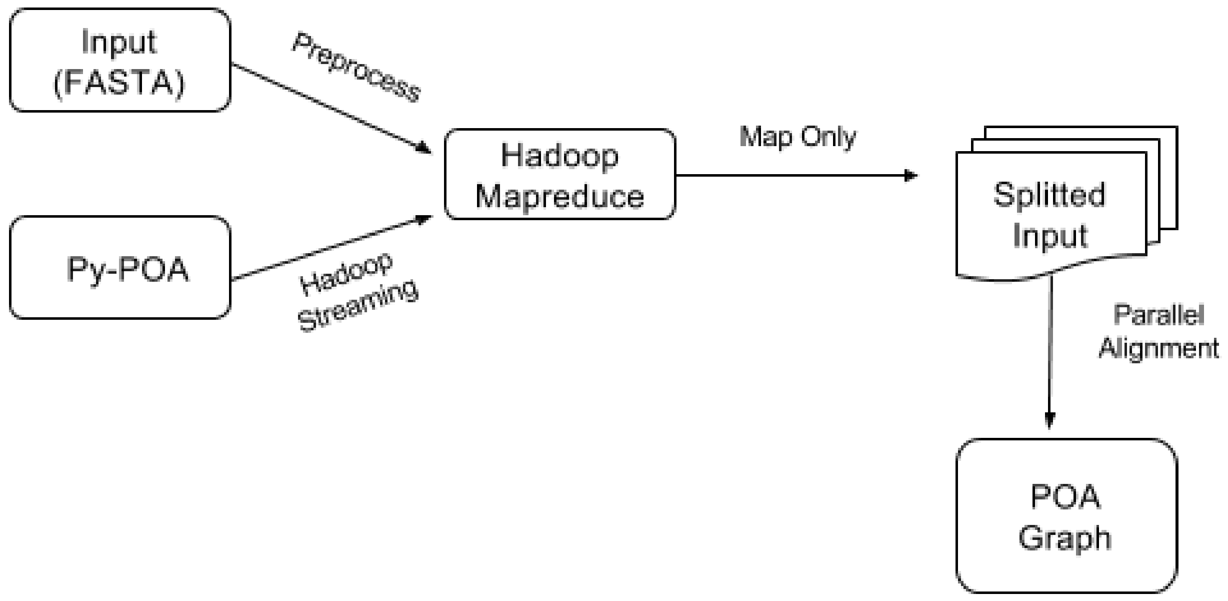


IEEE Celebrating 125 Years of Engineering the Future

Hadoop's Mapreduce



Proposed Framework

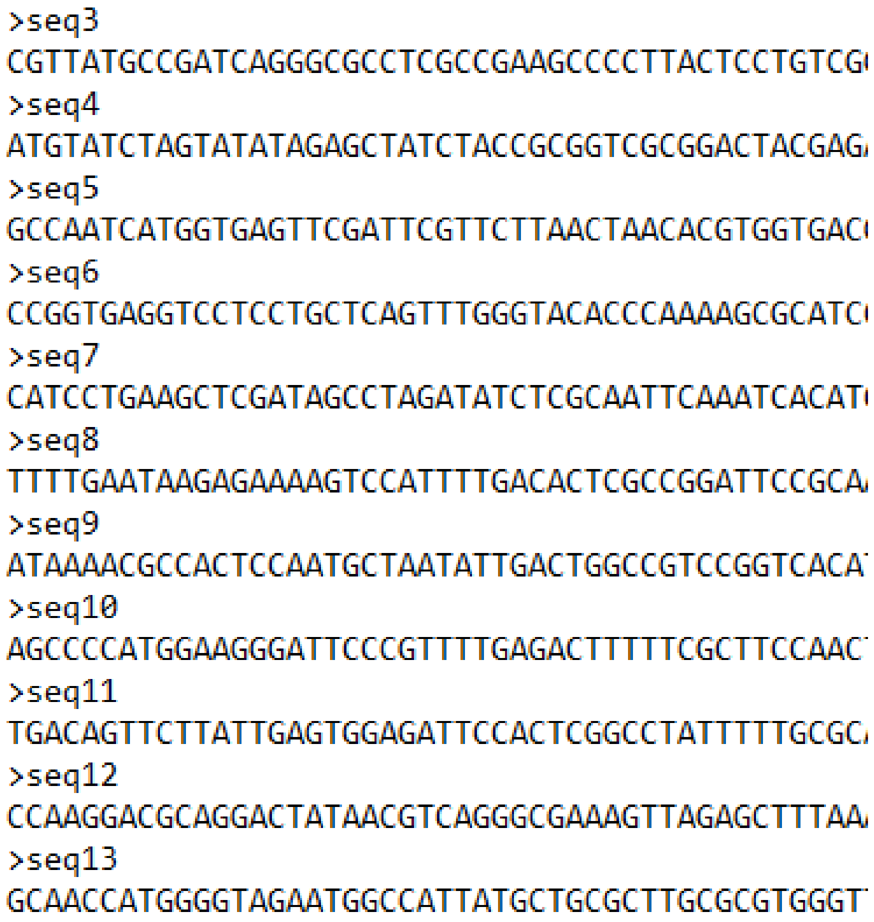


Framework Configuration

- Python based POA Implementation
- Amazon EC2 based Multi Node Hadoop Cluster
- (10 Nodes - 2M, 8S)
- EC2 Master Instance Configuration - t2.micro type, Linux (Ubuntu 16.04) Image, 4GB Ram, 16 GB SSD
- EC2 Slave Instance Configuration - t2.micro type, Linux (Ubuntu 16.04) Image, 2GB Ram, 4 GB SSD
- Hadoop Streaming for Python Wrapper
- Hadoop’s Mapreduce Framework (Map Only Job)
- PuTTY for SSH Tunnelling and WinSCP for File Transfer

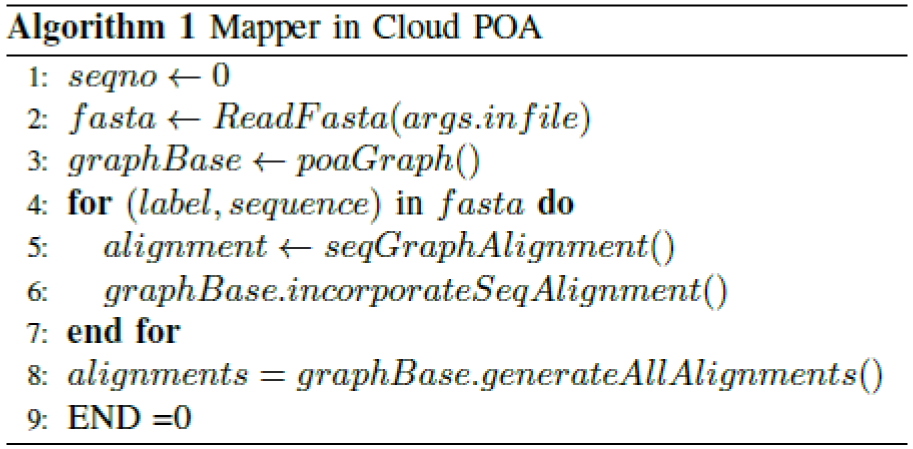
Input Preprocessing

- Generally contains sequences 150-200 BPs long (short-read sequences)
- Tested with up to 100k sequences
- All sequences were DNA sequences
- Both Synthetic Data and Benchmark Data has been used
- For Benchmarking, GENIE Gene Finding Benchmark Dataset was used (793 Long Human Genes)



```
>seq3
CGTTATGCCGATCAGGGCGCCTCGCCGAAGCCCCTTACTCCTGTCG
>seq4
ATGTATCTAGTATATAGAGCTATCTACCGCGGTCGCGGACTACGAG
>seq5
GCCAATCATGGTGAGTTCGATTCGTTCTTAACTAACACGTGGTGAC
>seq6
CCGGTGAGGTCCTCCTGCTCAGTTTGGGTACACCCAAAAGCGCATC
>seq7
CATCCTGAAGCTCGATAGCCTAGATATCTCGCAATTCAAATCACAT
>seq8
TTTTGAATAAGAGAAAAGTCCATTTTGACACTCGCCGGATTCCGCA
>seq9
ATAAAACGCCACTCCAATGCTAATATTGACTGGCCGTCCGGTCACA
>seq10
AGCCCCATGGAAGGGATTCCCGTTTTGAGACTTTTTCGCTTCCAAC
>seq11
TGACAGTTCTTATTGAGTGGAGATTCCACTCGGCCTATTTTTGCGC
>seq12
CCAAGGACGCAGGACTATAACGTCAGGGCGAAAGTTAGAGCTTTAA
>seq13
GCAACCATGGGGTAGAATGGCCATTATGCTGCGCTTGCGCGTGGGT
```

Py-POA Implementation



Algorithm 1 Mapper in Cloud POA

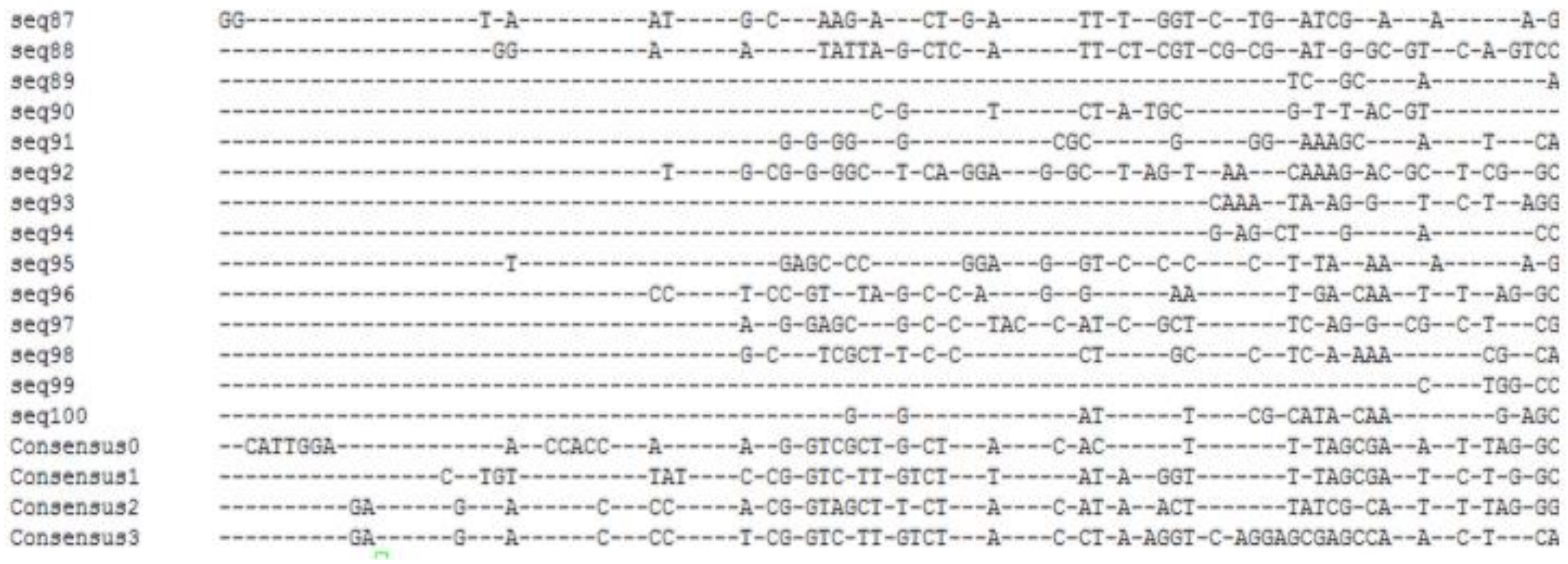
```
1: $seqno \leftarrow 0$
2: $fasta \leftarrow ReadFasta(args.infile)$
3: $graphBase \leftarrow poaGraph()$
4: for $(label, sequence)$ in $fasta$ do
5:     $alignment \leftarrow seqGraphAlignment()$
6:     $graphBase.incorporateSeqAlignment()$
7: end for
8: $alignments = graphBase.generateAllAlignments()$
9: END =0
```

•Simplefasta - Preprocessing FASTA

•Poagraph - Generate POA Graph

•Seqgraphalignment - Align and Insert new sequence to the Graph

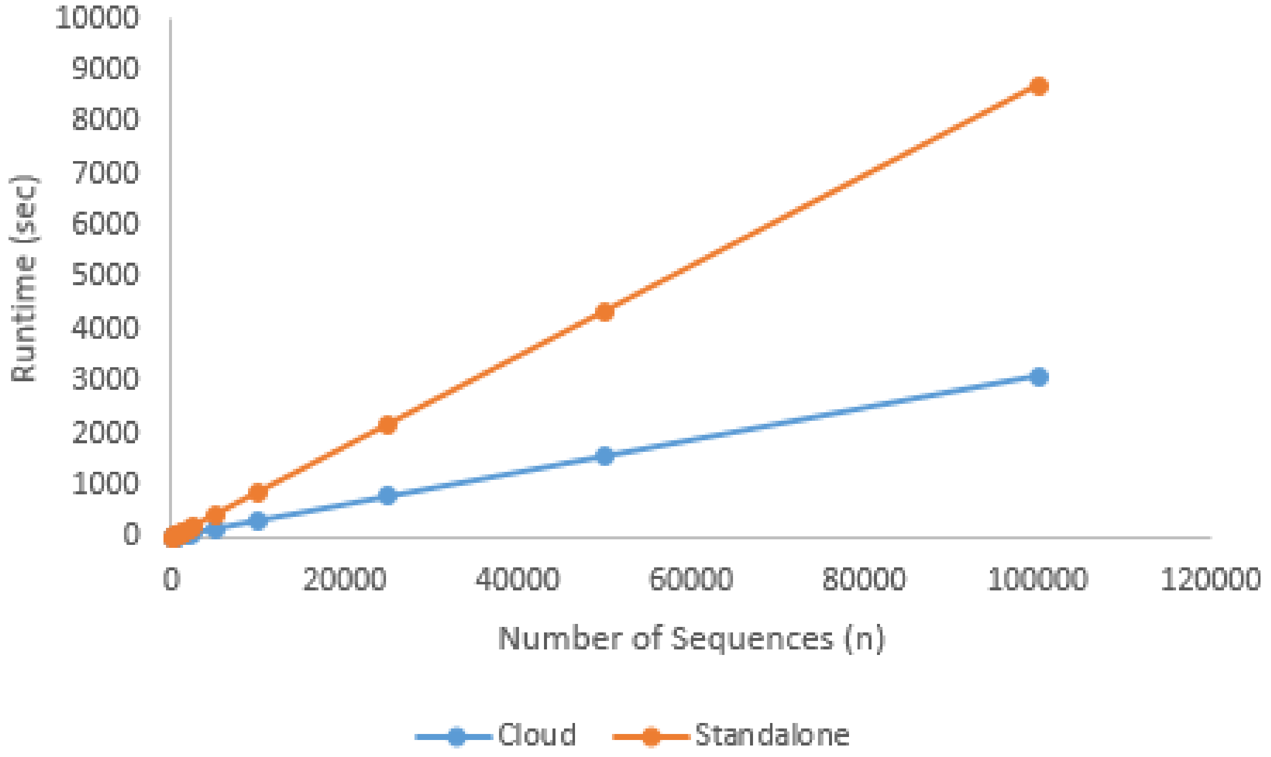
Final Result



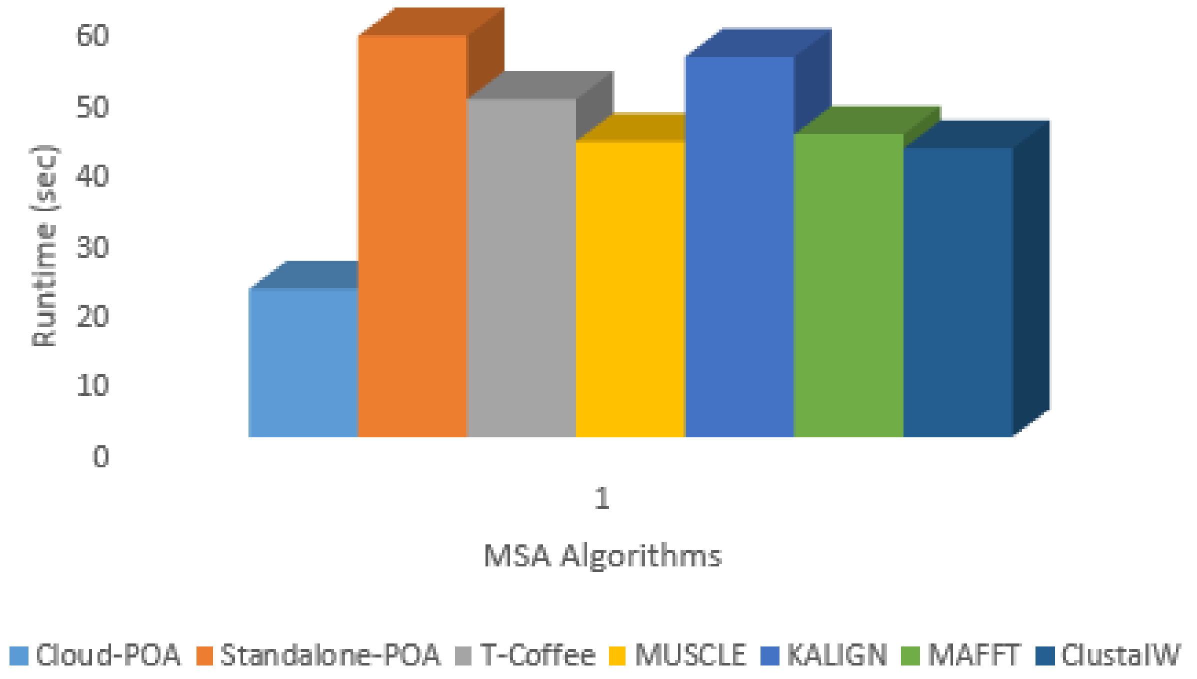
```
seq87           GG------------------T-A----------AT-----G-C---AAG-A---CT-G-A------TT-T--GGT-C--TG--ATCG--A---A------A-G
seq88           ---------------------GG----------A------A-----TATTA-G-CTC--A------TT-CT-CGT-CG-CG--AT-G-GC-GT--C-A-GTCC
seq89           ----------------------------------------------------------------------------------TC--GC----A---------A
seq90           --------------------------------------------------C-G------T------CT-A-TGC--------G-T-T-AC-GT----------
seq91           -------------------------------------------G-G-GG---G-----------CGC------G-----GG--AAAGC----A----T---CA
seq92           ----------------------------------T-----G-CG-G-GGC--T-CA-GGA---G-GC--T-AG-T--AA---CAAAG-AC-GC--T-CG--GC
seq93           ----------------------------------------------------------------------------CAAA--TA-AG-G---T--C-T--AGG
seq94           ----------------------------------------------------------------------------G-AG-CT---G-----A--------CC
seq95           ----------------------T--------------------GAGC-CC-------GGA---G--GT-C--C-C----C--T-TA--AA---A------A-G
seq96           ---------------------------------CC-----T-CC-GT--TA-G-C-C-A----G--G------AA-------T-GA-CAA--T--T--AG-GC
seq97           ----------------------------------------A--G-GAGC---G-C-C--TAC--C-AT-C--GCT-------TC-AG-G--CG--C-T---CG
seq98           ----------------------------------------G-C---TCGCT-T-C-C---------CT-----GC----C--TC-A-AAA-------CG--CA
seq99           --------------------------------------------------------------------------------------------C----TGG-CC
seq100          ------------------------------------------------G---G-------------AT------T----CG-CATA-CAA--------G-AGC
Consensus0      --CATTGGA-------------A--CCACC---A------A--G-GTCGCT-G-CT---A----C-AC------T-------T-TAGCGA--A--T-TAG-GC
Consensus1      -----------------C--TGT----------TAT----C-CG-GTC-TT-GTCT---T------AT-A--GGT-------T-TAGCGA--T--C-T-G-GC
Consensus2      ----------GA------G---A------C---CC-----A-CG-GTAGCT-T-CT---A----C-AT-A--ACT-------TATCG-CA--T--T-TAG-GG
Consensus3      ----------GA------G---A------C---CC-----T-CG-GTC-TT-GTCT---A----C-CT-A-AGGT-C-AGGAGCGAGCCA--A--C-T---CA
```

IEEE
Celebrating 125 Years
of Engineering the Future

Performance Evaluation



Performance Evaluation



Performance on GENIE Benchmarking Dataset

Future Work

- Build guide tree for establishing sequencing order
- Aligning two PO-MSA Graph with each other
- Make some optimization to further improve the alignment quality

References

1. Lee et. al (2002), Multiple Sequence Alignment using Partial Order Graphs. Bioinformatics, Vol 18, Issue 3.

2. Lee et. al (2003),Generating Consensus Sequence from Partial Order Multiple Sequence Alignment Graphs. Bioinformatics, Vol 12, Issue 4.

3. Driscoll et. al (2013), An Overview of Multiple Sequence Alignments and Cloud Computing in Bioinformatics. ISRN Biomathematics.

4. Pais et. al (2014), Assessing the Efficiency of Multiple Sequence Alignment programs. AMB.

Affiliation and Acknowledgement

Bangladesh University of Engineering and Technology

THANK YOU

Celebrating 125 Years
of Engineering the Future