Follow the Ants

Multiple Sequence Alignment Using Divide and Conquer + Ant Colony Optimization

Yaqin Si, Amanda Duffy, Rahul Gautum

Introduction

- MSA Importance
  - Reveals common ancestry
  - Reveals similar function based on sequence
- MSA algorithms
  - Dynamic programming
  - Progressive algorithm
  - Iterative algorithm
  - Naturally inspired algorithms

Template Paper

- “An efficient algorithm for multiple sequence alignment based on ant colony optimization and divide-and-conquer method” by Wei Liu, Ling Chen, and Juan Chen, 2007

The Algorithm:

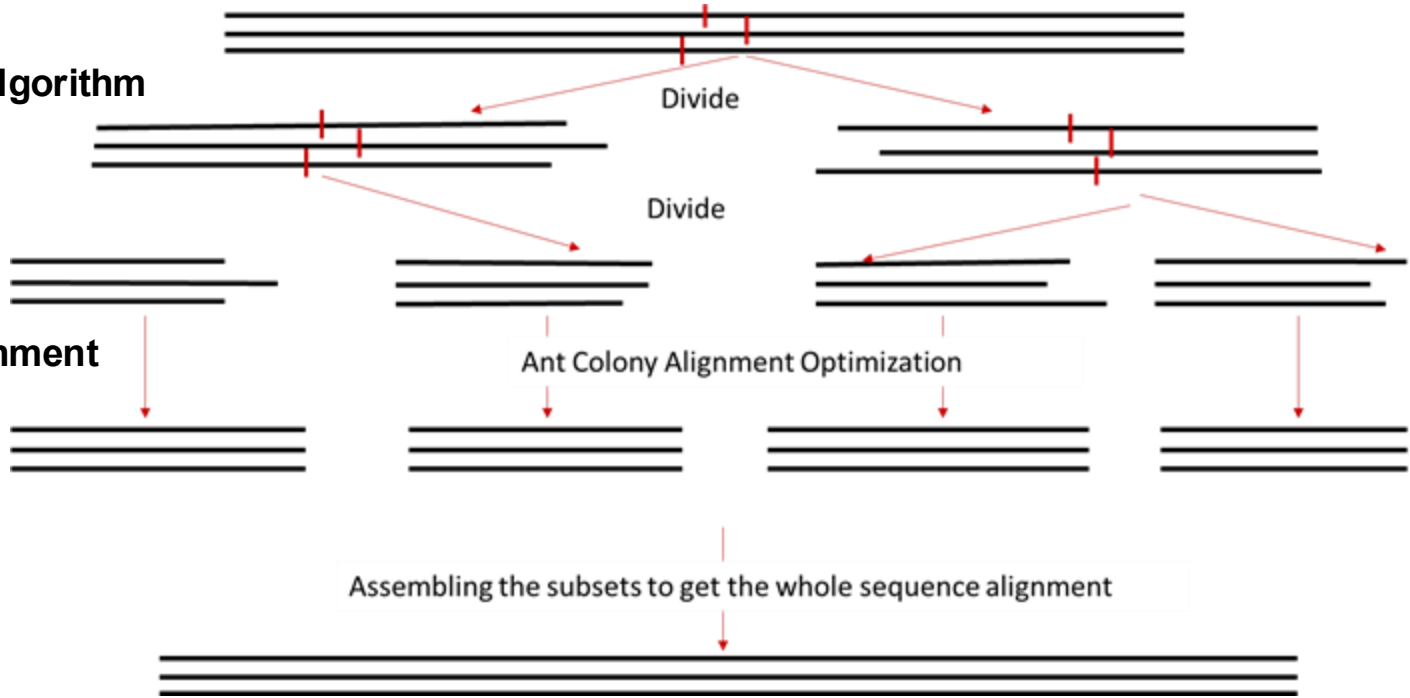
Divide and Conquer
+
Ant Colony Optimization

Overall Idea

1. Divide and Conquer
Optimized with Genetic Algorithm

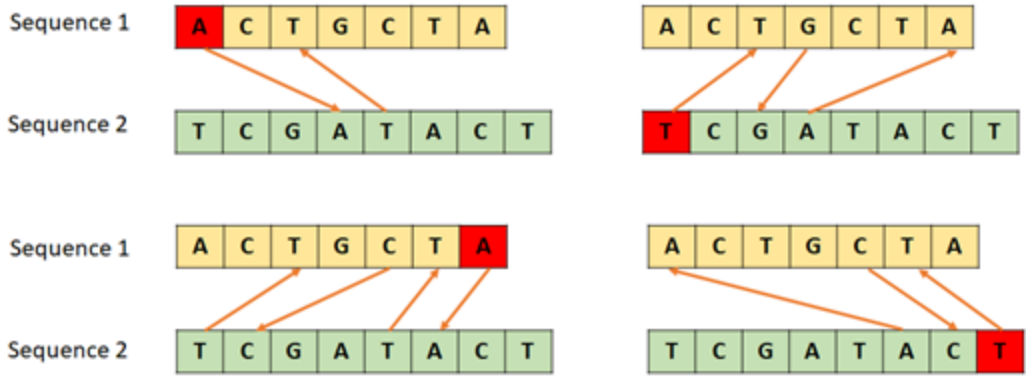
2. Multiple sequence alignment
Ant Colony Optimization

3. Assembly



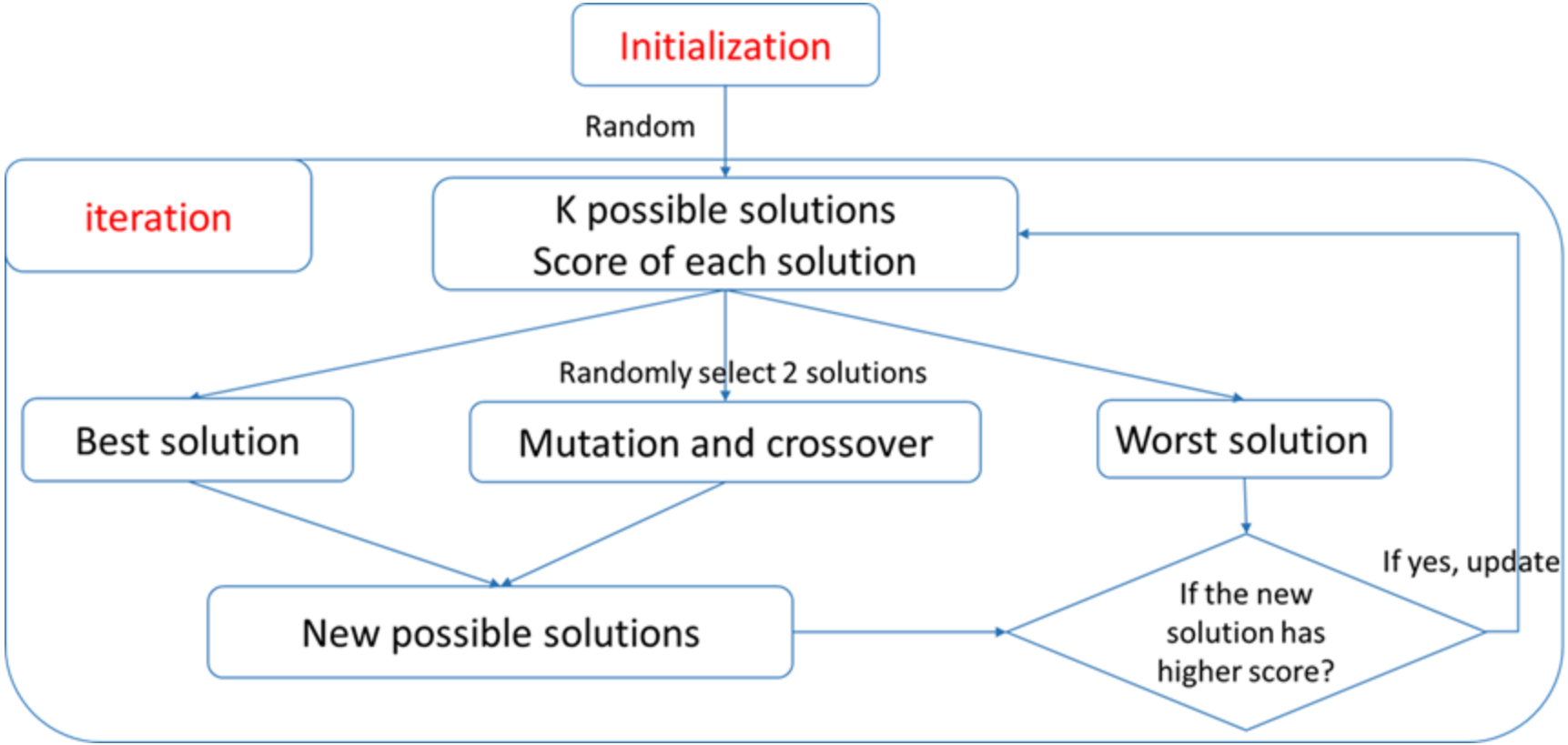
Divide and Conquer

1. Define a simplified sum of pair score



Score:
+2+3
+4+3

2. Optimize cutoff positions with Genetic Algorithm



2.. Optimize cutoff positions with Genetic Algorithm

2.1 k possible solutions for cutoff positions (random)
Find the best and worst score

2.2 Generate A new solution

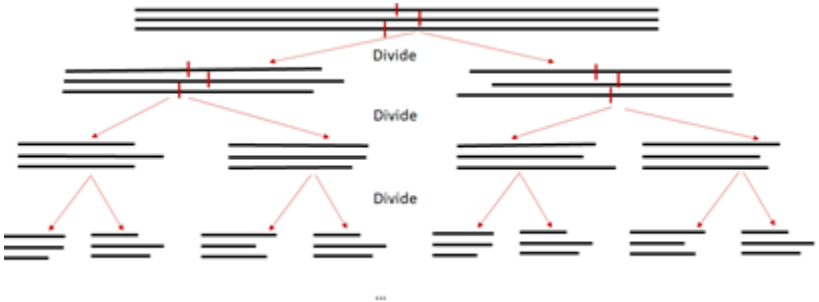
a. Borrow information from the current best solution
b. If score_new > score_worst, replace the worst solution

2.3 Generate new solutions from Mutation and Crossover

	Cut 1	Cut 2	Cut 3	Cut ...	Cut k-1	Cut k
Seq 1	x11	x12	x13	x1j	x1(k-1)	x1k
Seq 2	x21	x22	x23	x2j	x2(k-1)	x2k
Seq 3	x31	x32	x33	x3j	x3(k-1)	x3k
Score	S_1	S_2	S_3	S_j	S_k-1	S_k

K possible cutoff positions for all sequences

3. Divide recursively



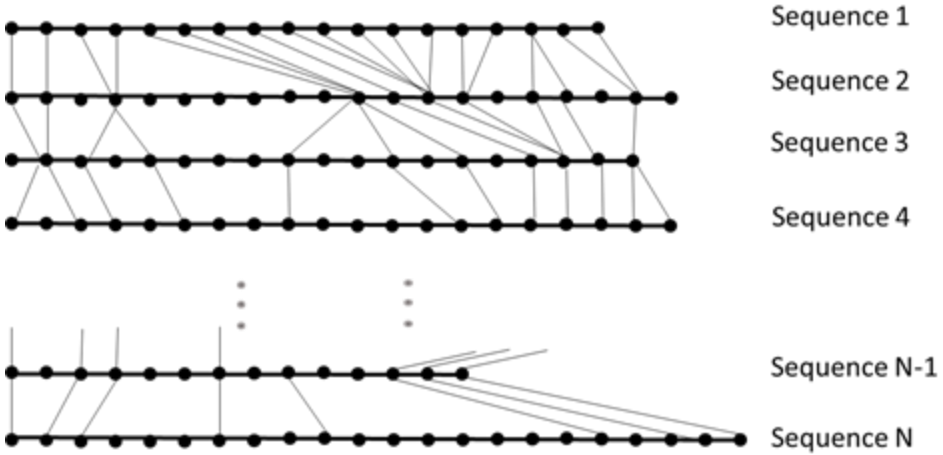
Cut p	Cut q
x1p	x1q
X2p	X2q
x3p	x3q
S_p	S_q

Cut p'	Cut q'
x1p	x1q
x2p'	x2q
x3q	x3p
S_p'	S_q'

Mutation and Crossover

Ant Colony Optimization - Forming Paths

- For every nucleotide of every sequence, form a path connecting its best matches, one from each sequence
- If a perfect match isn't found, map to either the next nucleotide or a gap
- Adjust pheromone along paths, leading to more "ants" choosing that path
- Form alignment from paths, filling in skipped nucleotides or gaps as needed
- Adjust pheromone and other parameters each iteration
- Choose best alignment over all paths
- Repeat 40 times, keeping best alignment



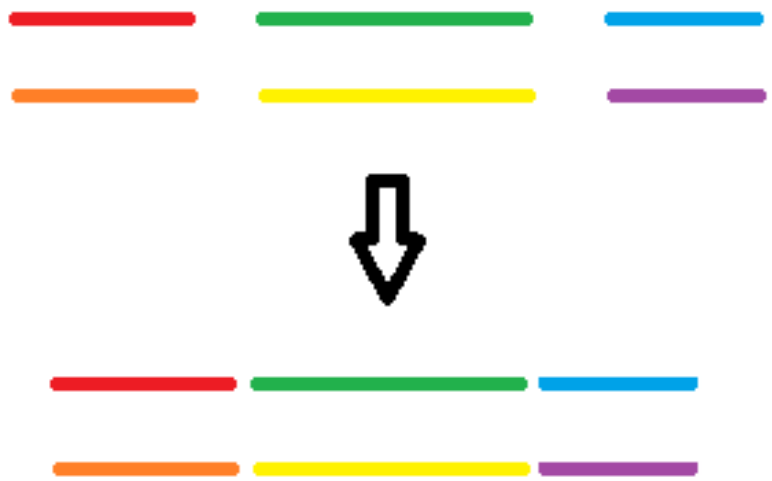
Ant Colony Optimization - Forming Alignments

- Cycle through each nucleotide in starting sequence and grab its path
- For each gap or nucleotide in path:
  - If gap, just insert a gap
  - Otherwise:
    - Fill in skipped nucleotides
    - Pad with spaces
    - Append matched nucleotide
  - Last iteration: append remaining nucleotides
- Append gaps until all subsequences are the same length

```
AGCCAA------ACG-AG-GATGGTTC
-GACAG-TGACA--GTAACG--G-TAC
AGACACCTGCT---G-AT-G--G-TAG
AGCCAA------ACG-AG-GATGGTTC
AGACACCTGCT---G-AT-GC-G-TAG
-GACAG-TGACA--GTAACG--G-TAC
```

Assembly

- We have several aligned subsequences
- Simply append them in order to form whole sequences



Evaluation

Parameter Testing

- Unfortunately, too many to properly test all of them
- Played around with different values to get idea of optimal solutions
- Significant parameters were tested more systematically

```
# initialize parameters. Note: Paper did not give any of these values.
a = 0.1  # pheromone parameter (smaller than b and c to start)
b = 0.4  # matching location parameter
c = 0.4  # location deviation parameter
d = 0  # number of cycle before for comparing alignments
h = 30  # range of character selecting
evap1 = 0.1  # evaporation coefficient
evap2 = 0.1  # evaporation coefficient
pheromone_threshold = 0.75  # if pheromone levels pass this threshold,
va = 0.5  # velocity of adjusting a
vb = 0.5  #velocity of adjusting b
vc = 0.5  #velocity of adjusting c
ta = 0.75  # threshold of a
tb = 0.1  # threshold of b
tc = 0.1  # threshold of c
sa = 0.75  # initialized value of a
sb = 0.1  # initialized value of b
sc = 0.1  #  initialized value of c
initial_pheromone = 0.1
prob_of_space = 0.03  # probability of selecting a space
```

Performance Testing

Time and Memory Testing

- Kept number of sequences constant (4) and varied sequence length (4 - 200)
- Kept sequence length constant (40) and varied number of sequences (4 - 25)
- Measured runtime and memory usage with BenchmarkTools' @timed

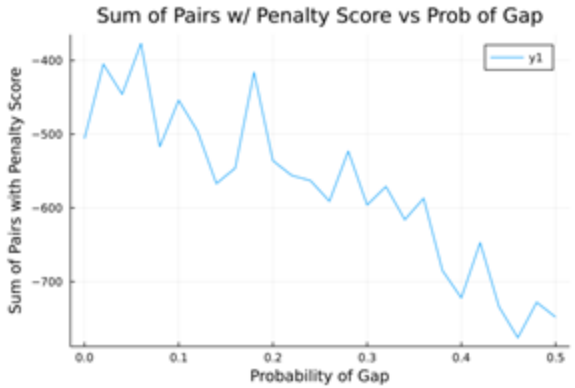
Accuracy Testing

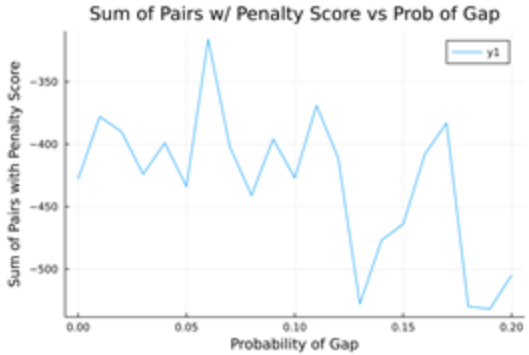
- Compared a few test sets to Rose and Clustal Omega reference scores
- Independently varied number of sequences and sequence length and plotted score next to Rose reference score

Results

Parameter: prob_of_space

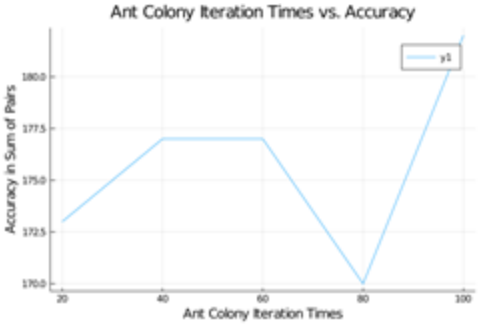
- Plotting score vs prob_of_space for 0 : 0.02 : 0.5 found that optimal probability was lower, between 0 and 0.2. Optimal probability in this graph was 0.06.
- Plotting score vs prob_of_space for 0 : 0.01 : 0.2 found optimal probability was again 0.06.

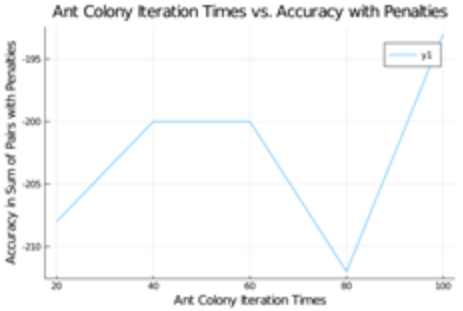


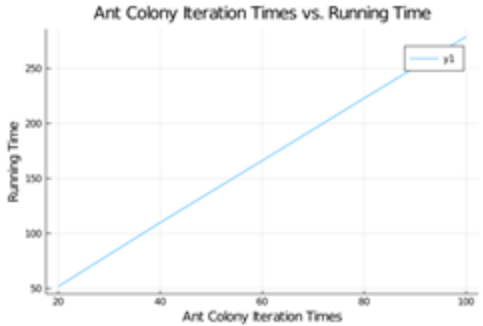


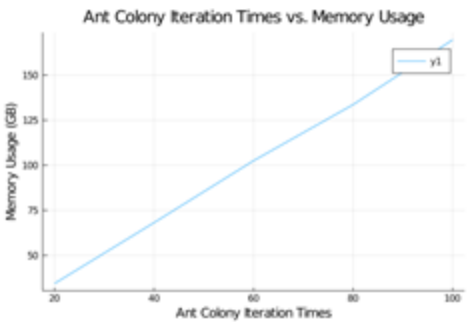
Parameter: Cyclenum

- Number of iterations of Ant Colony Optimization
- Tradeoff between accuracy and time/memory
- Score does not necessarily increase with iterations



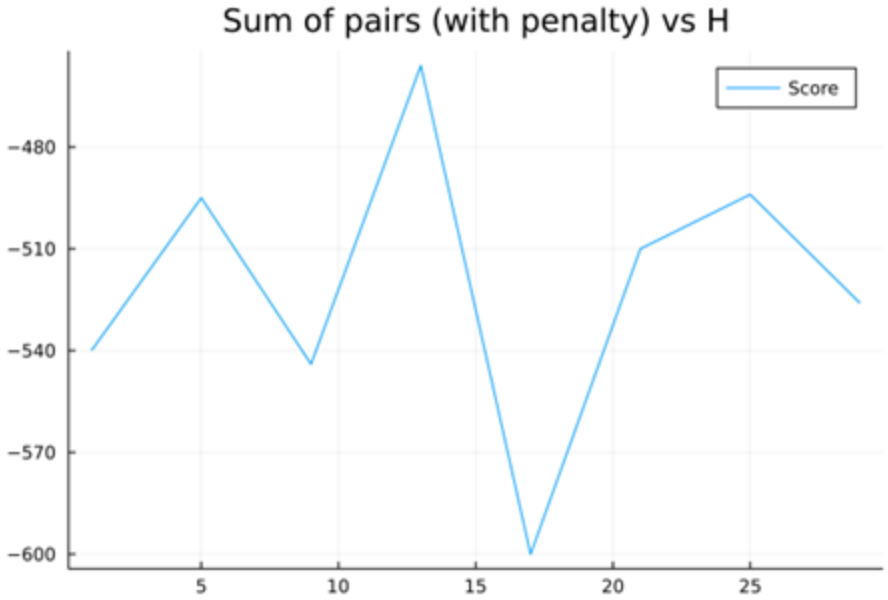






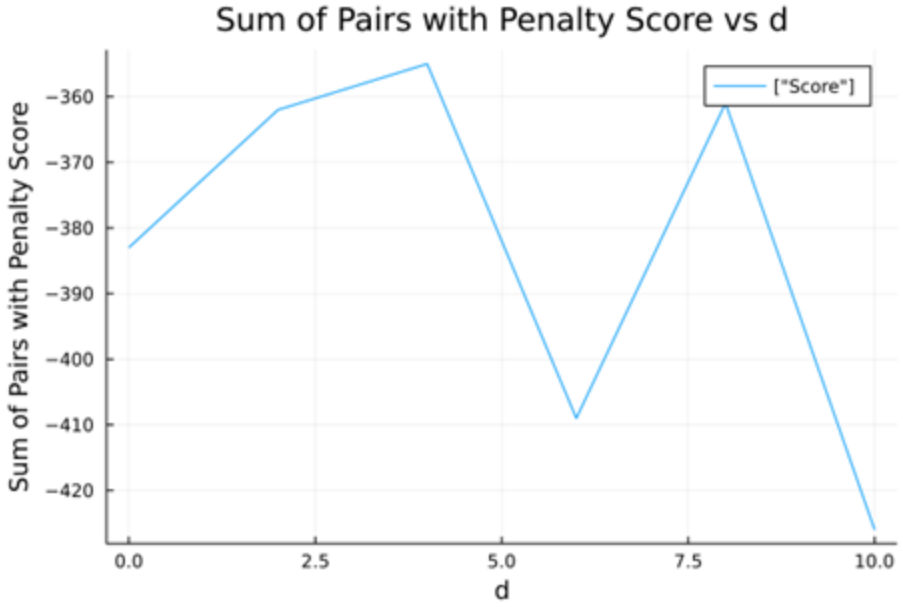
Parameter: h

- The number of potential matching nucleotides for each ant to consider
- Did not see a trend
- Needs to be repeated on longer sequences



Parameter: d

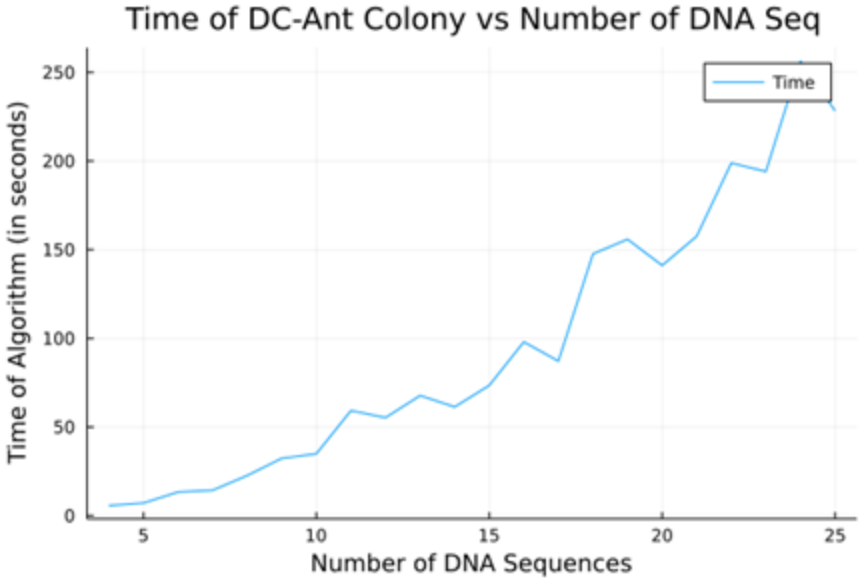
- After each cycle, compare best score to worst score from last d cycles. If worse, partially evaporate pheromone.
- No definite trend but seemed to do better for smaller values, peaking at 4
- Need to repeat for different test sets

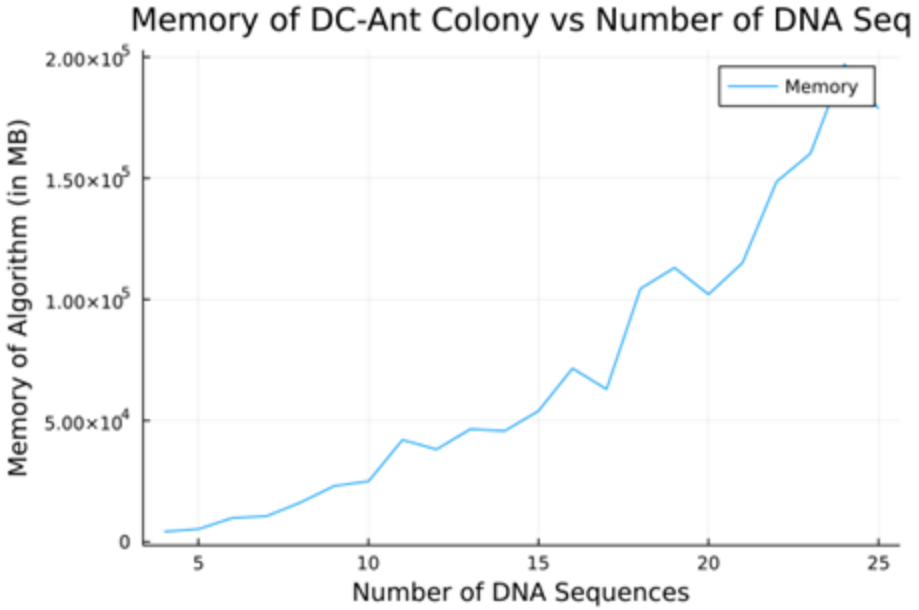


Accuracy Comparison to Rose and Clustal Omega

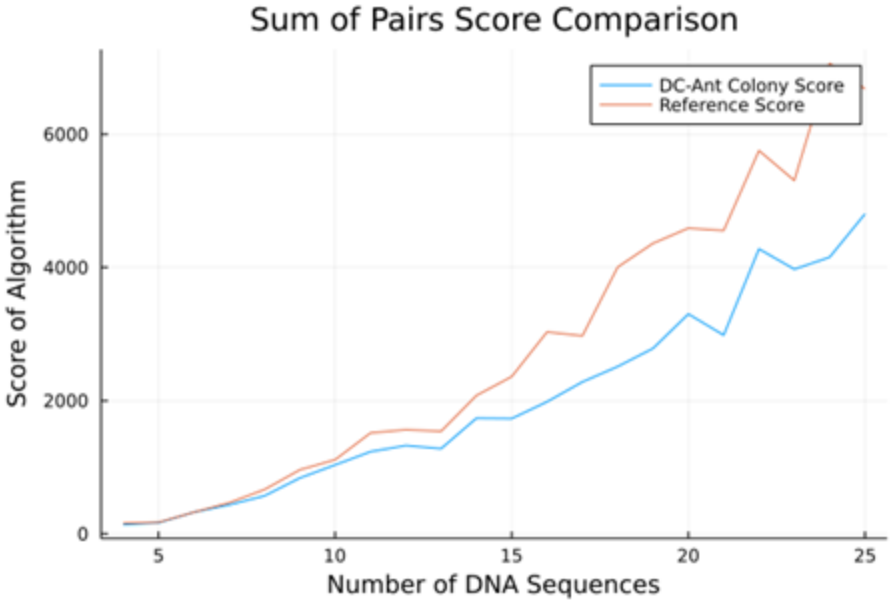
			Sum of Pairs Score			Sum of Pairs with Penalty Score (+1,-1,-2)		
Test Case	Number of Sequences	Length of Sequence	DC-Ant	Rose	Clustal Omega	DC-Ant	Rose	Clustal Omega
Test1	4	90	249	284	249	-437	-278	-256
Test2	5	220	784	1144	916	-2280	-2297	-1741
Test3	6	50	548	588	483	3	237	175
Test4	8	250	1921	2216	2270	-9957	-6370	-5814
Test5	9	150	1384	1473	1724	-7736	-22970	-6127
Test6	9	400	3352	5391	4442	-15730	-31242	-11822
Test7	10	300	3852	6441	5167	-20270	-23619	-9649
Test8	10	100	2495	3225	2709	-6011	-4818	-1873

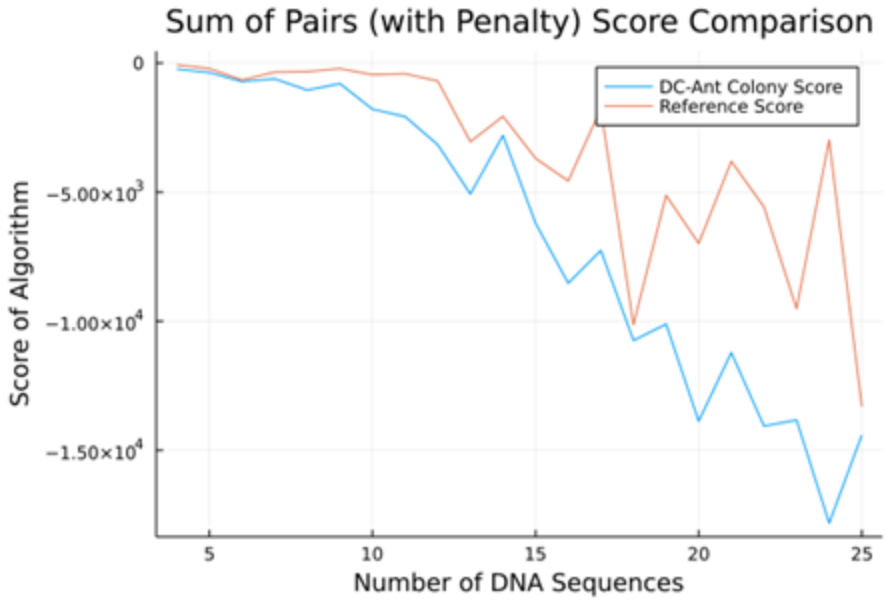
Time and Memory vs Number of Sequences



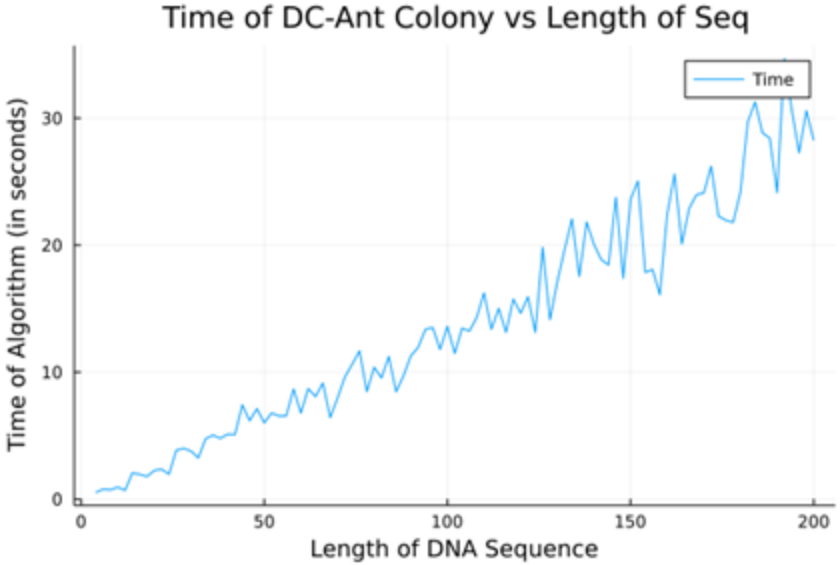


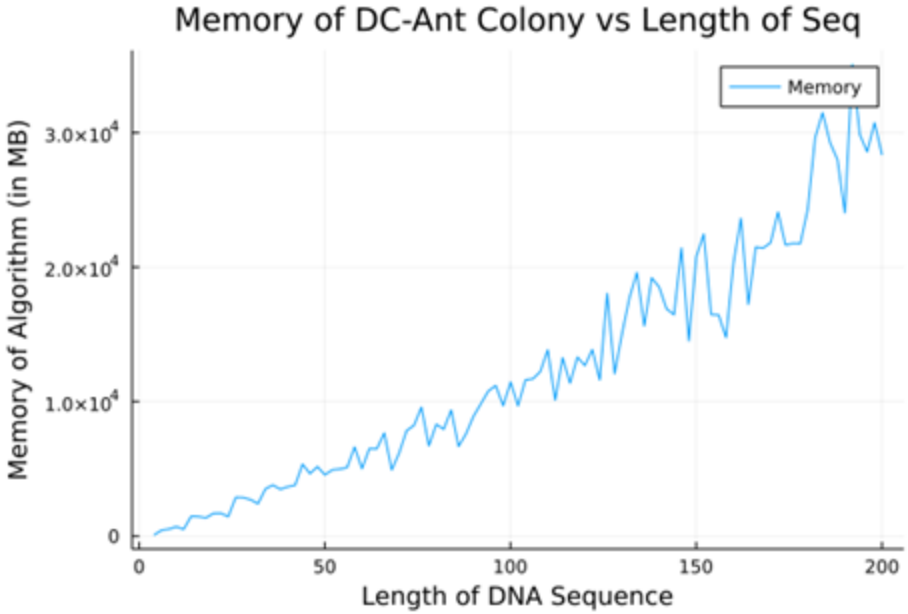
Accuracy vs Number of Sequences



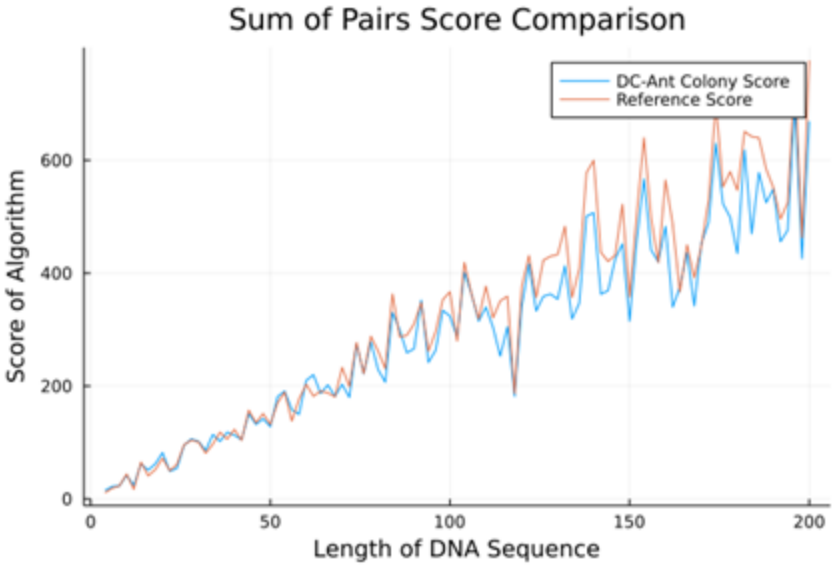


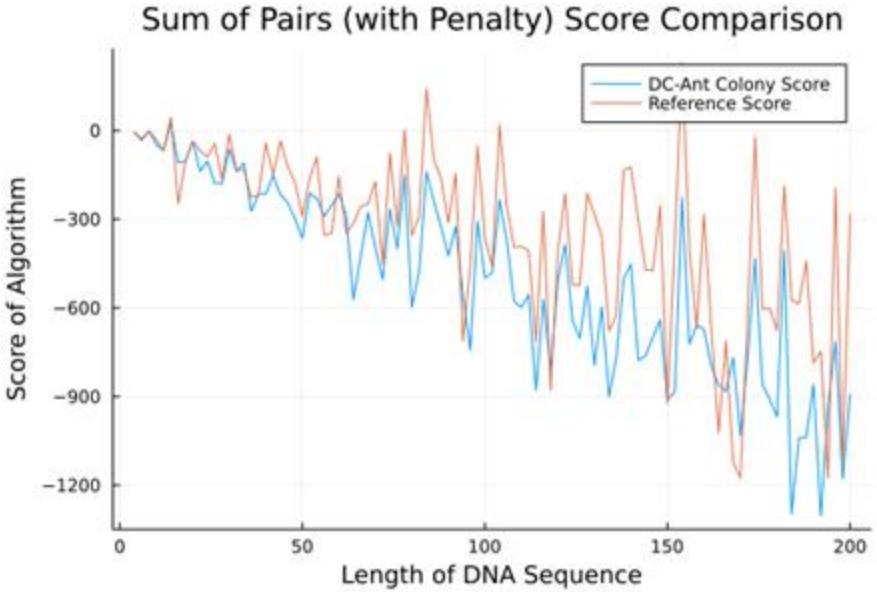
Time and Memory vs Length of Sequences





Accuracy vs Length of Sequences





Discussion

Performance

- Comparable scores to Rose and Clustal Omega
  - Always worse than Clustal Omega
  - Sometimes better than Rose, sometimes worse
- Slow for many sequences ($O(N^2)$) and long sequences ($O(len * log(len))$)
- Tradeoff between accuracy and speed/memory
- Paper ignores values of crossovers and cycles for complexity but not correct as these are in the range of 10^3

Possible Improvements

Improving Code

- Lots of parameters to tweak
  - Initial pheromone levels
  - Importance of pheromone, matching location, location deviation
  - Velocity of adjusting pheromone, matching location, location deviation
  - Number of iterations
  - Scores/penalties for matches, mismatches, gaps
- Improve alignment algorithm
- More complex reassembly to avoid large strings of gaps

Improving Experiments

- Systematically test more parameters
- Test time/memory vs number of sequences for more than one constant sequence length
- Test time/memory vs sequence length for more than one constant number of sequences
- Test parameters on more than one set of sequences to avoid optimizing on a single set

Q and A