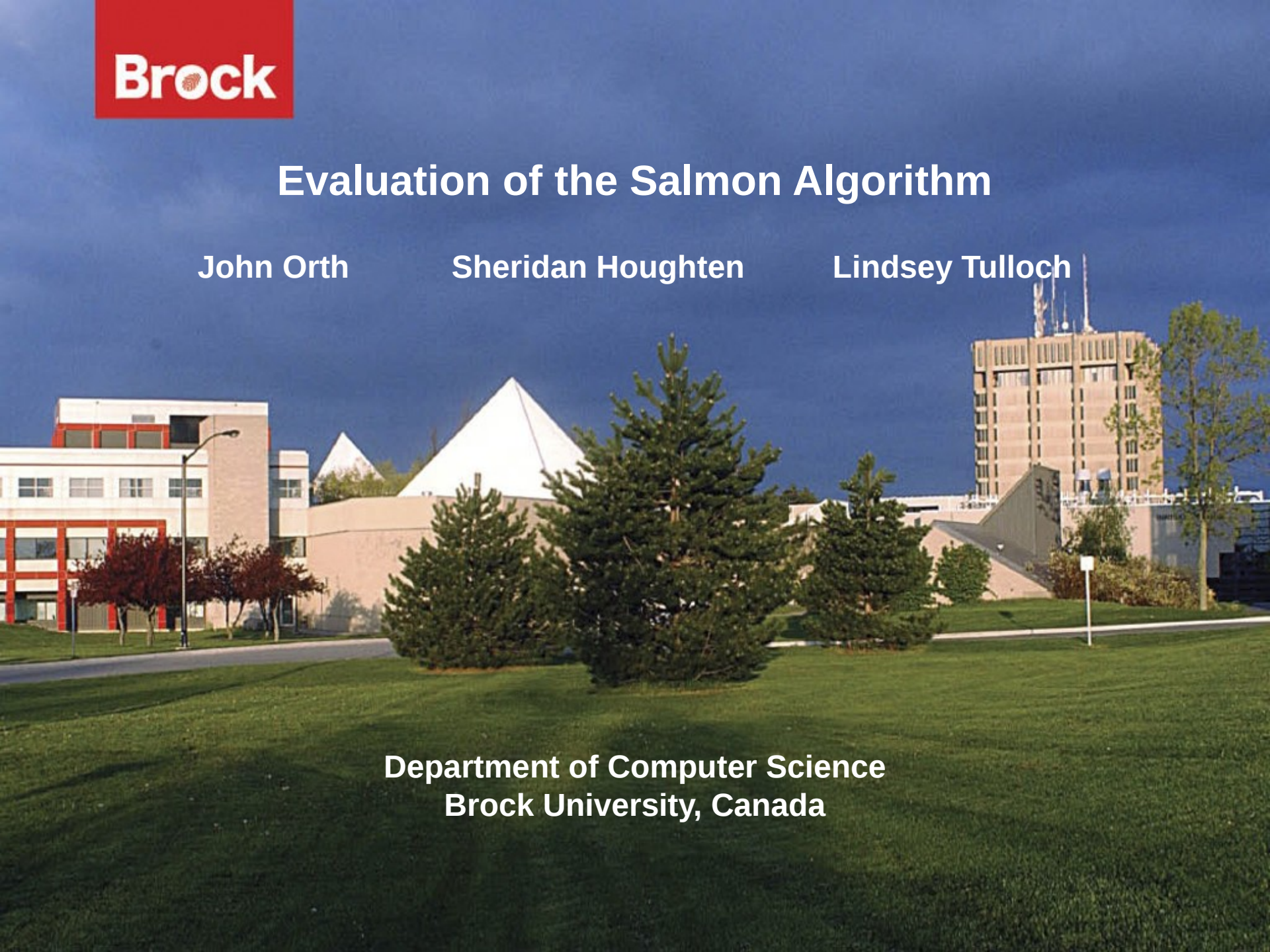


Evaluation of the Salmon Algorithm

John Orth Sheridan Houghten Lindsey Tulloch

Department of Computer Science
Brock University, Canada

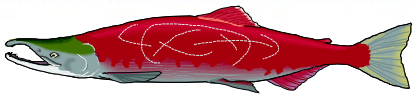
Research Motivations & Applications

1. Evaluate performance of the Salmon Algorithm on a Number of Bioinformatics Problems

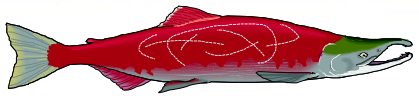
- Error Correcting Codes (can be used in sequencing applications)
- DNA Fragment Assembly Problem

2. Evaluate effectiveness of automated parameter tuning for Salmon Algorithm

- ParamILS



The Salmon Algorithm



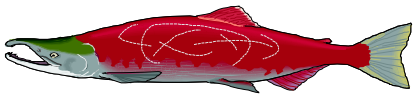
- population based search method
- inspired by salmon spawning behaviour*
- can be viewed as swimming along edges of a graph

Each salmon:

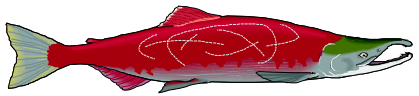
- contains two lists:
- a tabu list: vertices in the current path under construction
- a memory list: copy of the parent's completed tabu lists

- attracted to water

*The behaviour of the software salmon is idealized. We make no claim that real salmon behave in exactly this fashion.



The Salmon Algorithm



Tunable Parameters:

1) Initial water level Multiplier IM

2) Population Size

3) Memory Probability φ

4) Reproduction Fraction σ

5) Roulette Selection Exponent α

6) Number of elite salmon

The probability p_n of selecting $item_n$ using roulette selection is given by:

$$p_n = \frac{(waterLevel_n)^\alpha}{\sum_{x \in Candidates} (waterLevel_x)^\alpha}$$

Error Correcting Codes

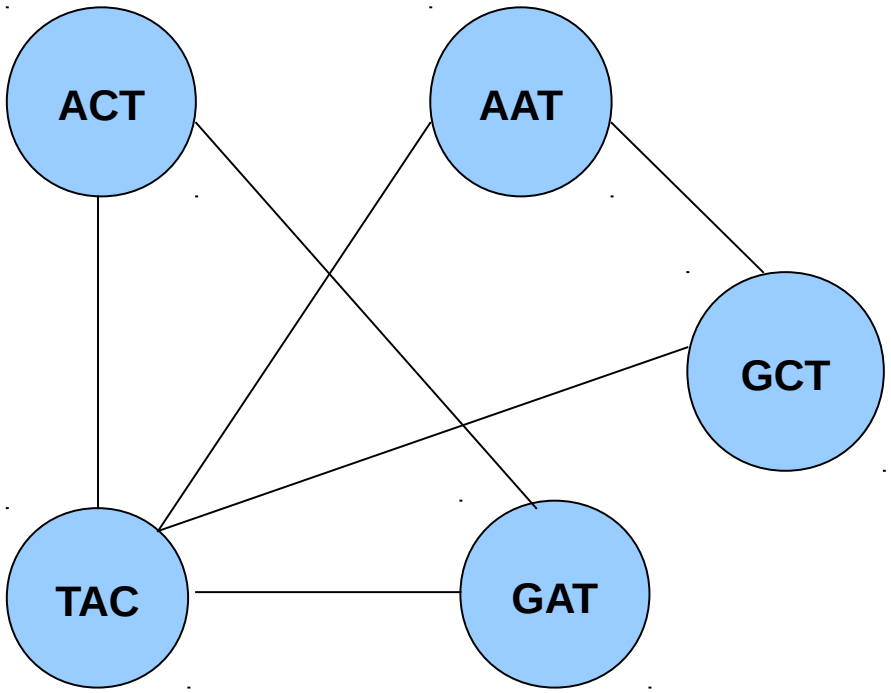
- A **set of words** that all have a certain **minimum distance** from each other.

- The distance between words is the number of changes required to transform one word into the other.

- The distance metric is defined according to the application, e.g. **Hamming distance** (substitutions only) or **Edit distance** (insertions, deletions, or substitutions).

- The determination of the largest code for a given length, distance, and alphabet is a difficult problem that may be solved with a search meta-heuristic such as a GA.

Code Clique Equivalence Example



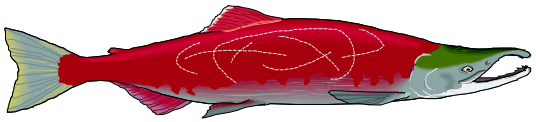
- Vertices (words) share an edge if they are at a minimum distance 2.
- In this graph there are two cliques of size 3, but none of size 4.

The Salmon Algorithm for Finding Maximum Cliques

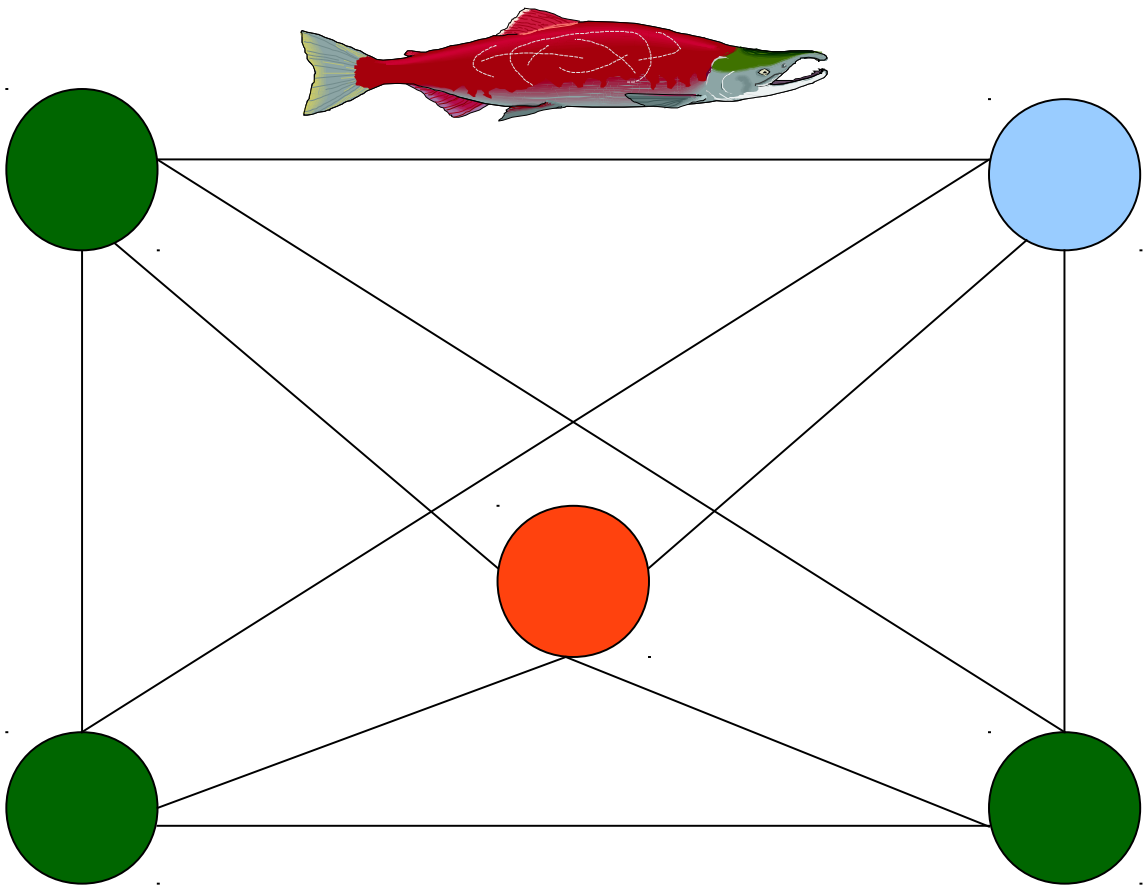
To begin the algorithm a random clique is generated and placed in each salmon's memory list.

- Vertices with more water have a higher probability of being selected.

- The salmon then begins traversing the vertices in its memory list, as shown on the next slide.

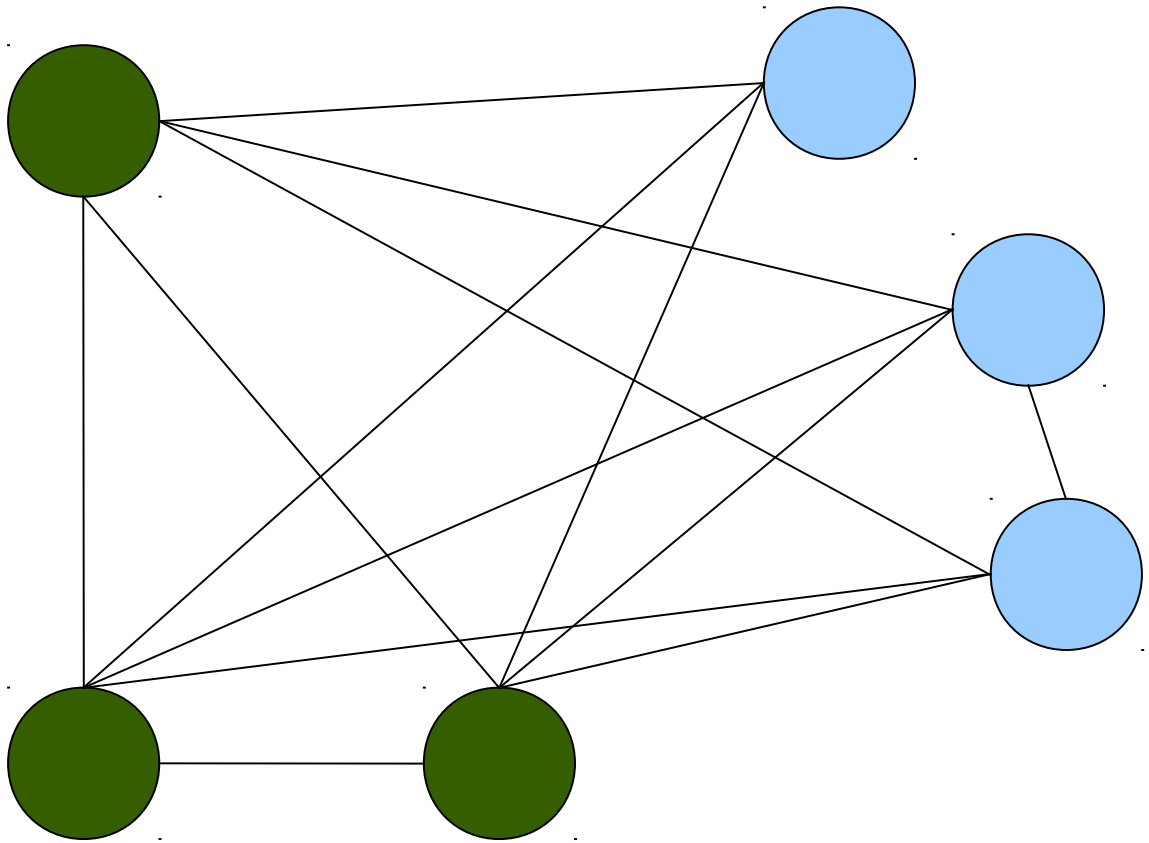


1. Traverse the Memory Vertices



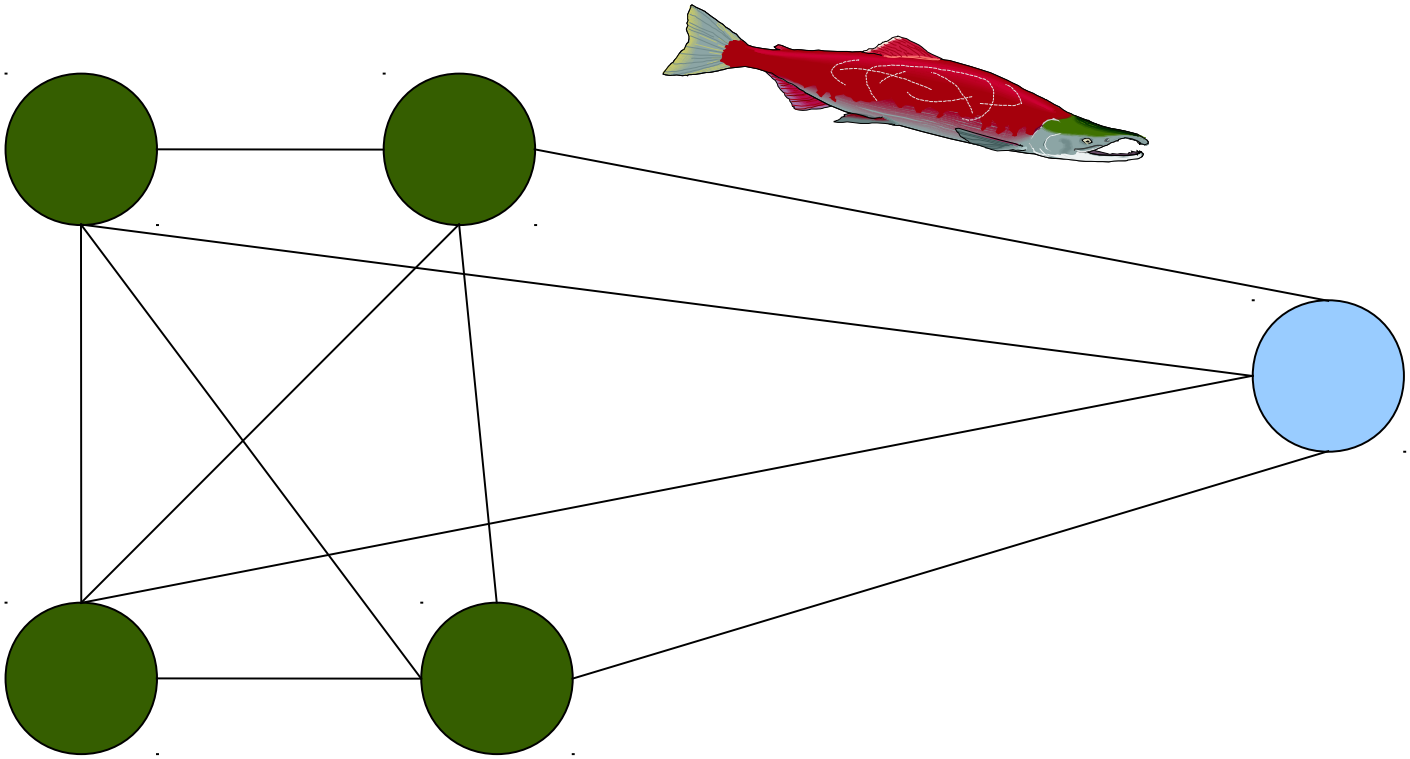
- The salmon swims to each vertex in its memory list, adding each to its tabu list with probability ϕ.
- Green have been added.
- Red have not been added.
- Blue are not yet traversed.

2. Construct the Candidates List



- After the memory list has been traversed, the candidates list is created.
- Candidates (blue) are those vertices that share an edge with all vertices in the clique (green).
- Candidates may or may not share an edge with each other.

3. Add Candidates to Clique



- The salmon selects vertices from the candidates list by using roulette selection on water level values.
- After each selection the list is updated.
- The process continues until no candidates remain.

4. Update Water Levels
   Each salmon adds an amount of water to each vertex in its clique equal to the size of its clique.

5. Save Elite
   The best salmon is saved to an elite salmon.

6. Produce Children.
   The best σ salmon (σ is a fraction 1/n where n is an integer) produce children. The parents then 'die' and the cycle repeats to step 1.

Optimum Parameter Values from previous study on edit codes

- ϕ (memory retention): .8
- σ (survival rate): .5
- IM (initial water level multiplier): 1 to 10 times best known*
- number of elite: 0
- Population: 100
- α (roulette exponent): variable

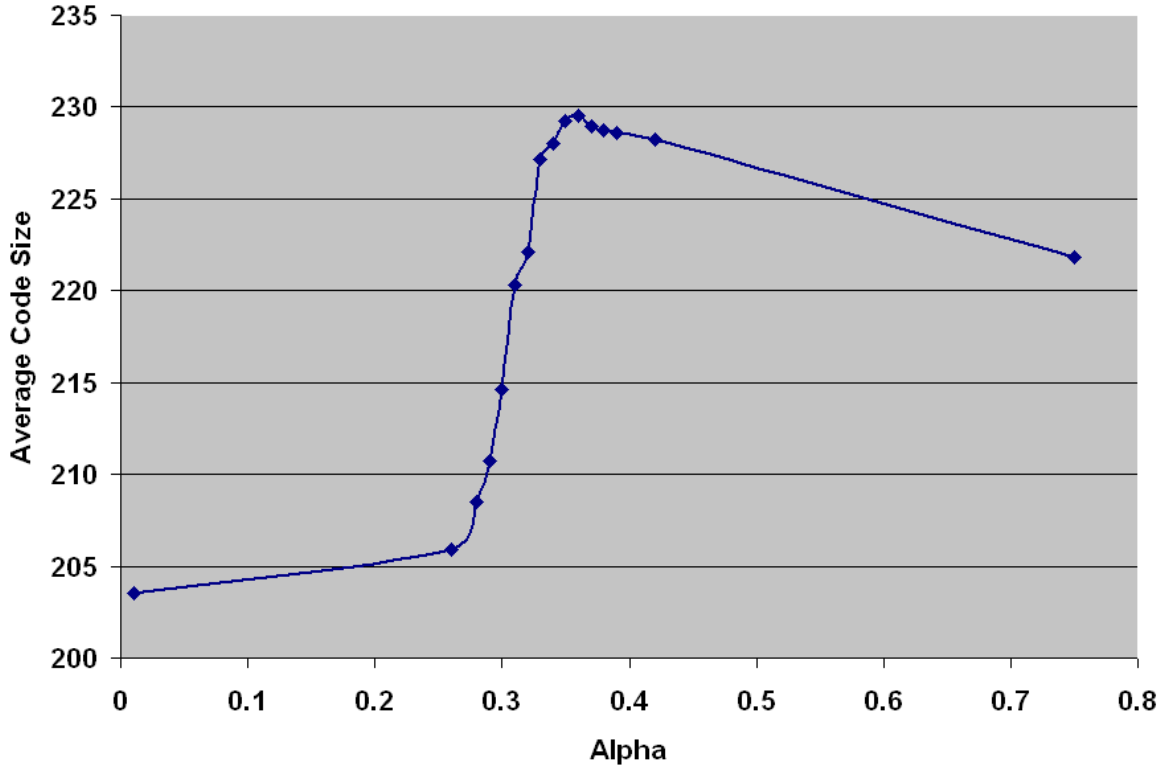
- These were used as a starting point in the current study on Hamming codes

***best known is the largest clique known for a given data set.**

Salmon Algorithm Ternary Hamming Distance Code Results

Code	$(5,3)_3$	$(6,3)_3$	$(7,3)_3$	$(8,3)_3$
Best Known	18	38	99	252
Salmon Best	18	38	99	236

Ternary Hamming codes, length 8 and distance 3
Average code size vs α



The best results do not occur at the maximum but in the area around the inflection point

Covering Codes

Closely related to error-correcting codes

- Let W be the set of all words of length n:
- A covering code C of radius r is a subset of W such that for all words in W at least one word in W is at distance r or less
- Opposite to error-correcting codes, we wish to minimize the size of the cover (rather than maximize the size of the code)

- **Minimization view:** fix the number of words covered to |W| and attempt to minimize the size of the cover needed to achieve this
- **Maximization view:** fix the number of words in the cover to an amount slightly greater than the best known cover size, and attempt to cover all words. If this is obtained then repeat with a smaller cover size.
- The maximization version performed better but both versions showed extreme sensitivity to parameter choice, requiring tedious changes even to the *level of 0.0001.*

Automated Parameter Tuning with param-ILS

So. . .

Employed ParamILS, an automated parameter tuning and Algorithm configuration tool [8] [9] (Hutter, Hoos)

For an algorithm A whose parameters are to be optimized for a set of problem instances D:

- Initial incumbent is the result of A run with default parameters
- Iterative Local Search (ILS) is performed to find local optima
- Parameter configurations chosen uniformly at random
- Each local optima is compared with incumbent
- Incumbent is updated when a better result is found
- Resulting incumbent is used as the starting point for next ILS cycle

TSP and DNA Fragment Assembly

To test ParamILS' efficacy, we used the DNA Fragment Assembly problem (FAP), the process of assembling fragments generated during sequencing into contigs.

- The layout phase of FAP can be translated into a TSP problem[12] (Mallen-Fullerton,Fernandez-Anaya) with the cities representing fragments and the distances representing overlap scores.
  - Two minor changes:
  - Negate Total: Minimization to Maximization
  - Remove distance to last city since FAP is not circular (insertion of dummy city is one solution)

Param-ILS + DNA Fragment Assembly + Salmon

Procedure:

- Provide list of parameters to test
- ParamILS feeds randomly selected parameters to Salmon algorithm
- Salmon algorithm uses parameters to run DNA Fragment Assembly problem as TSP
- Results are evaluated by ParamILS, parameters that give good results are adjusted and further explored

SUMMARY OF FRAGMENT ASSEMBLY BENCHMARKS [13]

Benchmark	Coverage	Mean Frag. Length	Number of Frags	Original Seq Length
x60189 4	4	395	39	3835
x60189 5	5	286	48	3835
x60189 6	6	343	66	3835
x60189 7	7	387	68	3835
m15421 5	5	398	127	10089
m15421 6	6	350	173	10089
m15421 7	7	383	177	10089

SALMON ALGORITHM FRAGMENT ASSEMBLY BEST RESULTS

Benchmark	Best Known	Salmon Best
x60189 4	11478	11478
x60189 4	14161	14159
x60189 4	18301	18088
x60189 4	21271	21104
m15421 5	38746	38018
m15421 6	48052	45260
m15421 7	55171	51891

Conclusions

- The Salmon Algorithm performed well in general for the 3 problems studied but there are several tunable parameters
- Demonstrates the need to examine parameters thoroughly for different problems, regardless of their apparent similarity: the optimal values for covering codes and error correcting codes differed noticeably, as well as for fragment assembly
- For error-correcting codes: the algorithm possibly struggled to take advantage of mathematical optimizations that might be available in the Hamming space, which is significantly more regular than the edit space (more in the next talk...)

- An extensive study of suitable parameter setting was performed for the 2 code problems
- For the fragment assembly problem, automated parameter tuning was studied.

Future Work

- Extending to the larger fragment assembly benchmarks
- Incorporating specialized methods e.g. mathematical optimization to help reduce search space first
- Investigate ways to reduce memory usage
- Investigating other options for parameter optimization (ParamILS was chosen primarily due to convenience)


Brock
Thank you.