Evolutionary Search Techniques for the Lyndon Factorization of Biosequences

Workshop on Evolutionary Computation for Permutation Problems@GECCO 2019

Amanda Clare, Jacqueline W. Daykin, Thomas Mills, Christine Zarges

Department of Computer Science
Aberystwyth University
Aberystwyth, Wales, UK

✉ c.zarges@aber.ac.uk

July 13, 2019

Overview

1. The Problem
2. The Algorithm
3. Results
4. Conclusions

Motivation: Stringology Meets Bioinformatics

Goal

Investigate structures in strings and permutations of the string alphabet with application to factoring genomes for sequence alignment.

Notation and Terminology

- Σ: an ordered alphabet
- *word*: finite sequence of symbols over Σ
- π: permutation defining the ordering of the alphabet

Typical Alphabets

- Standard English alphabet (26 letters)
- DNA alphabet (4 letters)
- Protein alphabet (20 letters)

Lyndon Words

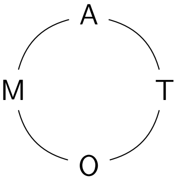
Given Ordered alphabet Σ

Lyndon Word

A finite word $x \in \Sigma^+$ is a Lyndon word if it is least alphabetically amongst all cyclic rotations of the letters.

Example English alphabet with standard lexicographical ordering

ATOM is a Lyndon word since ATOM $<$ OMAT $<$ MATO $<$ TOMA



Other examples: Evolution, Christine, Aberystwyth, Abstract, Amazing, Chicken, Moon

Lyndon Factorisation

Lyndon Factorisation

A factorisation of $x \in \Sigma^+$ into $x = \ell_1 \ell_2 \dots \ell_n$ where

- ℓ_i are Lyndon words and
- $\ell_1 \geq \ell_2 \geq \dots \geq \ell_n$

Example English alphabet with standard lexicographical ordering

$w =$ UNIVERSITY → U $\geq$ N $\geq$ IV $\geq$ ERSITY

Fact Any word $x \in \Sigma^+$ can be uniquely factored into a Lyndon factorisation.

Research Questions

- What impact does the manipulation of the alphabet ordering have on the resulting Lyndon Factorisation, specifically the number of factors?
- Determine an optimal ordering for a number of different objectives.

Applications

Sequence factorisation facilitates useful approaches such as parallelism and block compression to deal with the huge volumes of data.

- Bioinformatics: STAR, an algorithm to search for tandem repeats (approximate and adjacent repetitions of a DNA motif)
- Musicology: Enumerating periodic musical sequences
- Digital geometry
- Two-way string-matching
- Compression: In Suffix arrays + Burrows-Wheeler transform

On the Number of Factors

Example $w = 01^j0^21^{j-1}\dots0^j1$ for $j > 1$

- $0 < 1$: j factors

$$(01^j)\ (0^21^{j-1})\ (\dots)\ (0^j1)$$

- $1 < 0$: 3 factors

$$(0)\ (1^j0^21^{j-1}\dots0^j)\ (1)$$

How can we minimise the number of factors?

Existing approach Greedy Algorithm by Clare & Daykin

How can we maximise the number or balance the length of factors?

Observation Different alphabet sizes and usually no general pattern of characters.

Objectives

Example: bacdbdabbcdbbddbdbdabbacbabacbc

- Minimise the number of factors (a < c < d < b)

  (b) (acdbdabbcdbbddbdbdabbacbabacbc)

- Maximise the number of factors (a < b < c < d)

  (b) (acdbd) (abbcdbbddbdbd) (abbacb) (abacbc)

- Balance the length of the factors (b < a < c < d)

  (bacdbda) (bbcdbbddbdbda) (bbacbabacbc)

  - Standard deviation of the factor length
  - Difference between maximum and minimum length

- Find a specific number of factors (if possible)

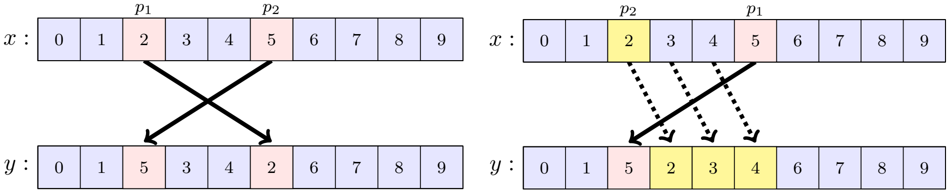
Duval's linear time and constant space algorithm to compute the number of factors.

Evolutionary Algorithm

1. **Initialisation**: Random + based on order of first appearance
2. **While** Exit Criteria Not Met **Do**
    - Evaluate alphabet orderings
    - **Parent Selection:** Select uniformly at random from top half of the population
    - Create offspring using crossover and mutation
    - **Replacement:** Offspring replace lower half of the population

Mutation

Swap Mutation and Insert Mutation



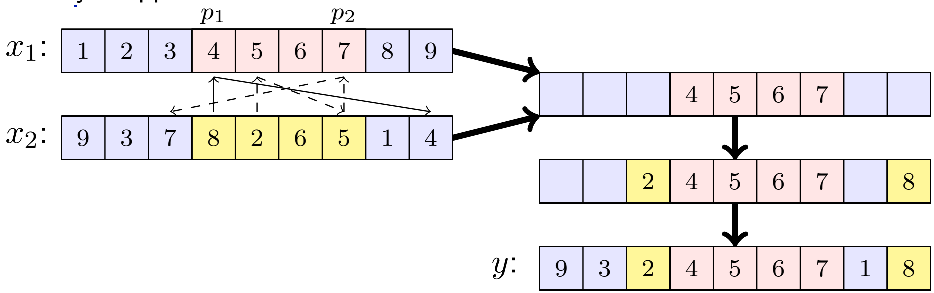
Observation Changes to low ordered characters have higher impact
→ Bias the selection of elements towards low ordered characters

Observation Changing the order of two elements has higher impact
→ Select Swap Mutation with higher probability

Crossover

Observation Need operator that preserves large parts of the ordering

Partially Mapped Crossover



Experimental Setup

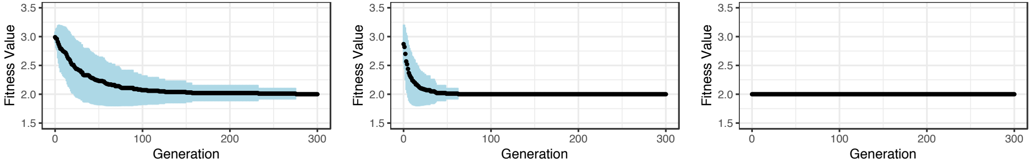
Parameters

- Generations: 1000
- Population size: 16
- Mutation bias:
  - Select one of the 3 lowest ordered elements with probability at least 0.3.
  - Select Insert Mutation with probability 0.9

Experiments

- **Random Sequences:**
  10 random sequences of length 300 over an alphabet of size 20
- **Biosequences:**
  573 protein sequences from a bacterial genome (Buchnera aphidicola)

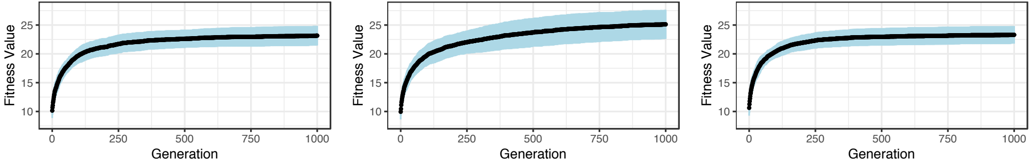
Random Sequences: Minimisation



Best individual in initial population has already good fitness
→ heuristic provides good results

Fitness converges to 2 for all random sequences considered.

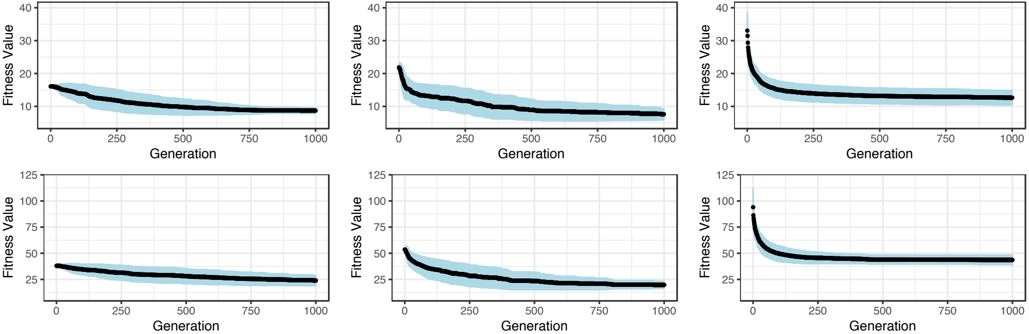
Random Sequences: Maximisation



Maximisation problem appears to be more difficult

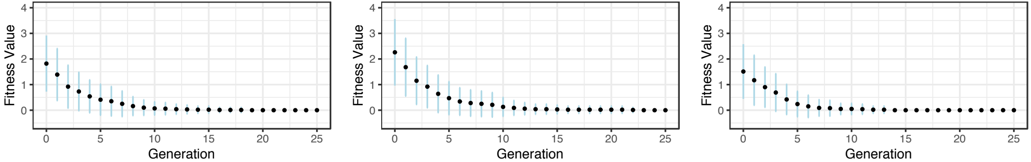
Maximal fitness reached across different sequences very similar

Random Sequences: Balanced



Balance problem also appears to be more difficult

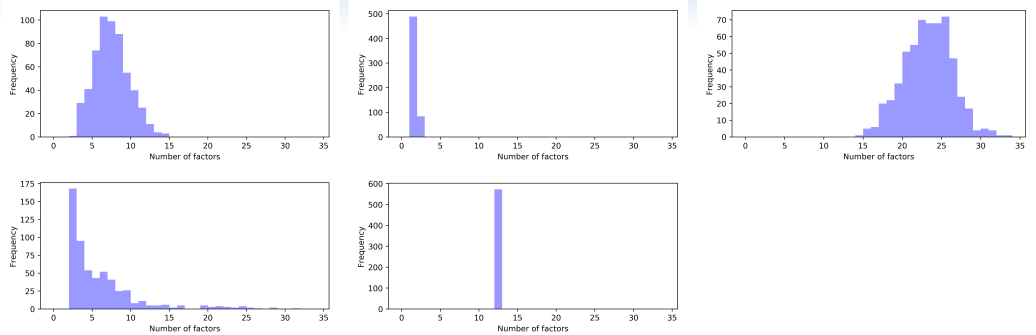
Random Sequences: Specific



Target 12 seems to be relatively easy to reach

More investigations needed to understand how the target influences the difficulty.

Biosequences



- **Lexicographic:** 4053 factors in total (mean 7, standard deviation 2.25).
- **Minimisation:** most cases just 1 factor, at most 2 factors
- **Maximisation:** Appears to follow a normal distribution, with mean of 22.7
- **Balanced:** Range of factors from 2 to 31
- **Specific:** Achieved for all sequences

Conclusions and Future Work

Evolutionary algorithm for finding an optimal alphabet ordering for the Lyndon factorisation problem

Future Work

- Consider different ways to initialise the population
- More detailed analysis of different operators for permutation problems and the underlying fitness landscape
- Investigate the solutions for the minimisation problem as they capture information about the protein sequences