Advances and Applications in Mathematical Sciences
Volume 22, Issue 1, November 2022, Pages 251-259

A DNA APPROACH TO SOLVE THE ASSIGNMENT PROBLEM WITH INVOLUTIVE FACTOR FREE STRANDS

C. ANNAL DEVA PRIYA DARSHINI, KAJLA BASU and E. MELITA VINOLIAH

Department of Mathematics
Madras Christian College
Chennai, India
E-mail: annaldevapriyadarshini@mcc.edu.in

Department of Mathematics
National Institute of Technology
Durgapur, India
E-mail: kajla.basu@maths.nitdgp.ac.in

Department of Mathematics
SRM Institute of Science and Technology
Kattankulathur, India
E-mail: melitav@srmist.edu.in

Abstract

The field of DNA Computing emerged with the research work of Adleman [2] where he solved the Hamiltonian Path problem through the use of biological molecules. This new connection led to many research studies to solve problems that cannot be solved by silicon-based binary computers. The strong point of using DNA lies in its intrinsic feature of a four letter alphabet with very specific bonding property. This is mathematically modelled as an involution function and formal language theoretic studies of DNA strands like the involutively bordered words have yielded rich mathematical results. We have introduced the involutive factor free words as DNA strands over restricted alphabets to eliminate undesirable properties in the context of DNA-based calculations. This paper is a study on applying the involutive factor free words to encode and utilize the bio-operations to solve the assignment problem. It is an attempt to apply the involutive factor free words in industry-related problem. In [13], Wang et al. have solved the unbalanced assignment problem based on DNA molecular computing. Motivated by

2020 Mathematics Subject Classification: 90C27, 90-08, 92D20.

Keywords: Assignment Problem, involutive factor free words, DNA Computing.

Received May 27, 2022; Accepted June 1, 2022

this study, we have examined whether the assignment problem can be solved by utilizing involutive factor free strands and obtained a positive result. The method is described and illustrated with an example.

1. Introduction

DNA computing has its own significance in many areas. Scientists and Researchers have made a number of theoretical and practical advancements in this field. DNA computing replaces the traditional silicon - based technology with DNA and molecular biology hardware. DNA computers would be able to solve the hardest problems more rapidly than the conventional ones. The physical characteristics of DNA could be used to store information and conduct calculations in molecular computers. These features include incredibly dense data storage, massive parallelism, and exceptional energy efficiency. The DNA computation allows a huge number of processes to be executed at the same time [3, 4, 5].

The function of DNA molecules store information and transfer it to the next generation. As the DNA carries instructions for constructing other components of cells, such as protein molecules, it is frequently compared to a collection of blueprints or a code. DNA comprises long polymers with simple units called nucleotides. Each nucleotide has alternating sugar and phosphate groups with certain bases connected to it. These four types of bases are Adenine (A), Thymine (T), Guanine (G), and Cytosine (C). The genetic code consists of a sequence of these four bases along the backbone that encodes information. Base-to-base bonds can be formed between nucleotides. There are certain specifications in which the bases are paired. A DNA molecule can be thought of as a string composed of the letters A, T, C and G, with the relationship between the strands depicted as an involution mapping [8].

The assignment problem is the largest significant problem in decision making. The assignment problem came to light as the resources on hand had different degrees of effectiveness. Therefore, the quality, cost and profit differ with respect to different resources. Tremendous efforts and methods were put forth to bring in numerous methodologies for solving problems in decision making.

The main objective of the assignment problem is to find the best assignment that minimizes the cost or maximizes the profit of workers with different skills to jobs. It works on the fact that each person should be assigned to only one job.

The assignment problem is mathematically formulated as follows:

$$\text{minimize}\, z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{1}$$

$$\text{Subject to } \sum_{i=1}^{n} x_{ij} = 1, \text{ for } i = 1, 2, \ldots, n,$$

$$\sum_{i=1}^{n} x_{ij} = 1, \text{ for } i = 1, 2, \ldots, n$$

and

$$x_{ij} = \begin{cases} 1, \text{ if the } i^{th} \text{ person is assigned the } j^{th} \text{ task} \\ \qquad\qquad \tilde{0}, \text{ otherwise.} \end{cases}$$

x_{ij} is the decision variable which helps us to know that i^{th} person is assigned the j^{th} job and c_{ij} is the cost for assigning person i to job j.

The most recognized algorithm to solve the assignment problem was the Hungarian algorithm [1]. In recent years, many authors have used different ways and methods to solve the same. Jayanta and Paul [7] altered the Hungarian algorithm and offered a modified version which reduced computations in 2015. Helena [12] studied the same under uncertainty environment where the concept of binary programming is involved in 2021. Stephen and Dinagar [13] introduced an algorithm which involved less number of iterations in 2021.

In this research, an assignment problem is examined in which each task-related cost is encoded as an involutive factor free word. A theoretical algorithm is presented based on the properties of involutive factor free words. To solve an assignment problem, there are three essential stages in DNA computation that must be achieved. The first is to utilize single-stranded DNA to encode the problem. A series of bio-operations are used to perform the actual computation. The computational result is then sequenced and decoded. For instance, when DNA polymerase is given a molecule with the sequence CATGTC, it will make new molecules with the sequence GTACAG. The polymerase allows DNA to replicate.

The cost matrix associated with an assignment problem is given and the objective is to find the best possible way of assigning task T_i to person P_j. The cost associated with each task T_i, $i = 1, 2, \ldots, n$ is an integer c_{ij}, $j = 1, 2, \ldots, n$. Since a cost matrix associated with a m x_n matrix has m-job values, n-machine values and mn cost values, we require $(m + n + mn)$ unique involutive factor-free words to encode the values. Γ_i^n has $2n$ words of length n. Hence, we choose involutive factor-free words of length k such that $2k = m + n + mn$ to encode a problem involving a $m \times n$ matrix.

2. Preliminaries

We present the basic definitions and results relevant to this study here.

2.1 Notations. The symbol Σ denotes a finite alphabet and Σ^* is the set of all words over Σ including λ, the empty word.

Let $\Sigma^+ = \Sigma^* - \{\lambda\}$. For any word w in this alphabet, $\text{alph}(w)$ is the elements of the alphabet found in w.

The length of a word w is denoted by $|w|$.

Let $w \in \Sigma^+$ be a word over $\Sigma u \in \Sigma^+$ is a factor of w if there are words $p, q \in \Sigma^*$ such that $w = \{u\}$. If $p = \lambda$, then the word u is a prefix of w and if $q = \lambda$ then the word u is a suffix of w. The set of factors of w is denoted by $F(w)$.

Definition 1 [6]. A mapping $\theta : \Sigma^* \to \Sigma^*$ such that the relation $\theta(xy) = \theta(x)\theta(y)$ holds is a morphism on Σ^*. If $\theta(xy) = \theta(x)\theta(y)$, then the mapping θ is an antimorphism on Σ^*.

Definition 2 [6]. The mapping θ is an involution function if $\theta(\theta(x)) = x$

for all $x \in \Sigma^*$. A word $u \in \Sigma^+$ is a θ-factor of w if and only if both the words $u, \theta(u) \in F(w)$. Or in other words, a word $u \in \Sigma^+$ is a θ-factor of w if both u and its θ-image is a factor of w.

Definition 3. Any word $w \in \Sigma^*$ is said to have a factorization if w can be expressed as a product of its factors. w can be written as $w = u_1 u_2 \ldots u_n$ where the u_i, $i = 1, 2, \ldots, n$ are factors of w [6].

Definition 4. A word $w \in \Sigma^*$ is involutively factored if w has at least one factorization $w = u_1 u_2 \ldots u_n$ where the $u_i \in \Sigma^*$, for $i = 1, 2, \ldots, n$ such that w can be expressed as a product of $\theta(u_i)$, $i = 1, 2, \ldots, n$ in some order [6].

Definition 5. A word $w \in \Sigma^*$ is an involutive factor free word if none of its factor is a θ-factor of the word [6].

Theorem 1. *A necessary and sufficient condition for a word $u \in \Sigma^+$ to be involutively factor free is that "for every $a \in \Sigma$ in $alph(u)$, $\theta(a) \notin alph(u)$."*

Definition 6. We define involutive factor free languages as languages such that $L = \{w / \forall v \in F(w), \theta(v) \notin F(w), \theta(v) \notin F(w_1),$ for any other $w_i \in L\}$.

Let $\Gamma_1 = \{A, C\}, \Gamma_2 = \{A, G\}, \Gamma_3 = \{T, C\}$ and $\Gamma_4 = \{T, G\}$. Then any $L \subseteq \Gamma_i$ will have words which are θ-factor free and concatenation of any two words in L will not result in involutively bordered words or a word with θ-factors.

3. DNA Based Technique to Solve the Assignment Problem using Involutive Factor-free Strands

Step 1. Designate an involutive factor free word of length k for every job and machine involved in the problem.

Step 2. Encode each cost in the given cost matrix by an involutive factor free word of length k repeated p times if the associated cost is p.

Step 3. Generate a string for every feasible assignment with the words separated by a marker symbol #. A feasible assignment is one where each row/column has a unique assignment.

Step 4. Find the length of each string.

Step 5. The shortest string gives the feasible solution.

Step 6. The optimum cost associated with the assignment problem is the sum of the costs corresponding to this shortest string.

4. Numerical Example

Consider the problem of assigning 4 jobs to 4 machines. The associated cost matrix is given below [14].

Table 4.1. The cost Matrix.

Machines→ Jobs↓	M_1	M_2	M_3	M_4
J_1	8	6	4	8
J_2	6	5	5	8
J_3	9	10	11	12
J_4	7	6	8	10

Since this is a 4×4 cost matrix, we need 24 unique words to represent the jobs/machines/costs. We can choose words in Γ_i^5. The involutive factor free words corresponding to the jobs and machines are represented as follows:

J_1 : *AAAAC*, J_2 : *AAAAA*, J_3 : *AAACA*, J_4 : *AACAA*, M_1 : *ACAAA*,

M_2 : *CAAAA*, M_3 : *ACCAA*, M_4 : *CCAAA*

The associated costs with respect to the machines and jobs are given below:

C_{11} : *CCCCACCCCACCCCACCCCACCCCACCCCACCCCACCCCA*,

C_{12} : *CCCCCCCCCCCCCCCCCCCCCCCCCCCCC*,

C_{13} : *CCCACCCCACCCCACCCCAC*

C_{14} : *CCCAACCCAACCCAACCCAACCCAACCCAACCCAA*

C_{21} : *CCCACCCCACCCCACCCCACCCCACCCCAC*

C_{22} : *CACCACACCACACCACACCACACCA*

C_{23} : *CACCCCACCCCACCCCACCCCACCC*

C_{24} : *ACCCCACCCCACCCCACCCCACCCCACCCCACCCCACCCC*

C_{31} : *CCAACCCAACCCAACCCAACCCAACCCAACCCAACCCAACCCA*

C_{32} : *CAAACCAAACCAAACCAAACCAAACCAAACCAAACCAAACCAAA CCAAAC*

C_{33} : *CAACCCAACCCAACCCAACCCAACCCAACCCAACCCAACCCAAC CCAACCCAACC*

C_{34} : *AAACCAAACCAAACCAAACCAAACCAAACCAAACCAAACCAAAC CAAACCAAACCAAACC*

C_{41} : *AACACAACACAACACAACACAACACAACACAACAC*

C_{42} : *ACAACACAACACAACACAACACAACACAAC*

C_{43} : *AACCAAACCAAACCAAACCAAACCAAACCAAACCAAACCA*

C_{44} : *ACCCAACCCAACCCAACCCAACCCAACCCAACCCAACCCAACCC AACCCA*

Note that we have repeated the words associated with the costs times if the cost entry is p. We have assigned involutive factor – free words for the given problem. Since this is a 4×4 matrix, we have 24 possible feasible assignments.

We create strings for each feasible assignment by catenating the words assigned with the words separated by the marker #. We also insert a marker at the beginning and end of the string so that a marker represents the start and end of a word associated with the table – job/machine/cost.

The optimal assignment is $J_1 \rightarrow M_3$, $J_2 \rightarrow M_4$, $J_3 \rightarrow M_1$, $J_4 \rightarrow M_2$

The string corresponding to the optimal assignment is given below:

#AAAAC#ACCAA#AAAA#CCAAA#AAACA#ACAAA#AACAA#CAAAA#CC CACCCCACCCCACCCCAC#ACCCCACCCCACCCCACCCCACCCCACCCCA CCCC#CCAACCCAACCCAACCCAACCCAACCCAACCCAACCCAACCCAAC# ACAACACAACACAACACAACACAACACAAC#

The string is of length 188.

This is a theoretical attempt at solving the classical assignment problem. Implementation in a lab may present some challenges. However, we have addressed the problem of avoiding inter – molecular and intra – molecular hybridizations which will not be disturbed under laboratory conditions.

5. Conclusion and Future Studies

The balanced assignment problem was encoded as involutive factor free words in this research, and the proposed methodology has been used to solve it. In practice, however, the parameters in the assignment problem are not precise. As a result, this method can be implemented to a fuzzy environment. The fuzzy assignment problem [9, 10, 11] is a real-world problem with many applications all across the world. Some of the special instances of a fuzzy assignment problem are generalised fuzzy assignment problems and multi-objective fuzzy assignment problems for which our proposed methodology is applicable.

References

[1] H. W. Kuhn, The Hungarian method for the assignment problem, Naval Research Logistics 2(1-2) (1955), 83-97. Available online at https://doi.org/10.1002/nav.3800020109

[2] L. M. Adleman, Molecular computation of solutions to combinatorial problems, Science, 266(187) (1994), 1021-1024. Available online at doi:10.1126/science.7973651

[3] A. Luca, On the combinatorics of finite words, Theoretical Computer Science 218(1) (1999) 13-39. Available online at https://doi.org/10.1016/S0304-3975(98)00248-5

[4] L. Kari, S. Konstantinidis, E. Losseva and G. Wozniak, Sticky free and overhang free DNA languages, Acta Informatica 40 (2003), 119-157. Available online at https://doi.org/10.1007/s00236-003-0118-7.

[5] L. Kari and K. Mahalingam, Involutively bordered words, International Journal of

Foundations of Computer Science 18 (2007), 1089-1106. Available online at doi:10.1142/S0129054107005145

[6] C. Annal Deva Priya Darshini, V. Rajkumar Dare, Ibrahim Venkat and K. G. Subramanian, Factors of word under an involution, Journal of Mathematics and Informatics 1 (2013-2014), 52-59. Available online at www.researchmathsci.org

[7] Jayanta Dutta and S. C. Pal, A note on Hungarian Method for solving assignment problem, Journal of Information and Optimization Sciences 36(5) (2015), 451-459; Available online at https://doi:10.1080/02522667.2014.926711

[8] Zhaocai Wang, Jun Pu, Liling Cao and Jian Tan, A parallel biological optimization algorithm to solve the unbalanced assignment problem based on DNA molecular computing, International Journal of Molecular Sciences 16 (2015), 25338-253352. Available online at doi:10.3390/ijms161025338.

[9] E. Melita Vinoliah and K. Ganesan, Fuzzy optimal solution for a fuzzy assignment problem with octagonal fuzzy numbers, Journal of Physics: Conference Series, 1000(1) (2018), 5 pages; Available online at doi:10.1088/1742-6596/1000/1/012016

[10] Supriya Kar, Aniruddha Samanta and Kajla Basu, Solution of fuzzy multi-objective generalized assignment problem, International Journal of Mathematics in Operations Research 15(1) (2019), 33-54. Available online at doi:10.1504/IJMOR.2019.101611

[11] E. Melita Vinoliah and K. Ganesan, A solution approach for a fully fuzzy assignment problem, IOP Conference Series: Materials Science and Engineering (2020), 7 pages; Available online at doi:10.1088/1757-899X/912/6/062046.

[12] Helena Gaspars-Wieloch, The assignment problem in human resource project management under uncertainty, Risks 9(21) 25 (2021) 1-17. Available online at doi:10.3390/risks9010025

[13] D. Stephen Dinagrand and B. Christopher Raj, Stephen's Algorithm for solving assignment problems, Advances and Applications in Mathematical Sciences 20(5) (2021), 887-894. Available online at www.mililink.com

[14] C. Annal Deva Priya Darshini, J. Jeba Jesintha and Sherlyne Grace, DNA Languages avoiding involutively bordered factors, Proceedings of the National Conference on Mathematics and Computer Applications, (2020).