BioSystems 150 (2016) 61–72

Contents lists available at ScienceDirect

BioSystems

journal homepage: www.elsevier.com/locate/biosystems

The role of crossover operator in evolutionary-based approach to the problem of genetic code optimization

Paweł Błażej, Małgorzata Wnętrzak, Paweł Mackiewicz*

Department of Genomics, Faculty of Biotechnology, University of Wrocław, ul. Joliot-Curie 14a, Wrocław, Poland

ARTICLE INFO

Article history:
Received 16 March 2016
Received in revised form 20 May 2016
Accepted 11 August 2016
Available online 20 August 2016

Keywords:
Amino acid
Crossover
Evolutionary algorithms
Genetic code
Mutation
Polarity

ABSTRACT

One of theories explaining the present structure of canonical genetic code assumes that it was optimized to minimize harmful effects of amino acid replacements resulting from nucleotide substitutions and translational errors. A way to testify this concept is to find the optimal code under given criteria and compare it with the canonical genetic code. Unfortunately, the huge number of possible alternatives makes it impossible to find the optimal code using exhaustive methods in sensible time. Therefore, heuristic methods should be applied to search the space of possible solutions. Evolutionary algorithms (EA) seem to be ones of such promising approaches. This class of methods is founded both on mutation and crossover operators, which are responsible for creating and maintaining the diversity of candidate solutions. These operators possess dissimilar characteristics and consequently play different roles in the process of finding the best solutions under given criteria. Therefore, the effective searching for the potential solutions can be improved by applying both of them, especially when these operators are devised specifically for a given problem. To study this subject, we analyze the effectiveness of algorithms for various combinations of mutation and crossover probabilities under three models of the genetic code assuming different restrictions on its structure. To achieve that, we adapt the position based crossover operator for the most restricted model and develop a new type of crossover operator for the more general models. The applied fitness function describes costs of amino acid replacement regarding their polarity. Our results indicate that the usage of crossover operators can significantly improve the quality of the solutions. Moreover, the simulations with the crossover operator optimize the fitness function in the smaller number of generations than simulations without this operator. The optimal genetic codes without restrictions on their structure minimize the costs about 2.7 times better than the canonical genetic code. Interestingly, the optimal codes are dominated by amino acids characterized by polarity close to its average value for all amino acids.

1. Introduction

It is worth mentioning that if we take into account the structure of the canonical genetic code with 61 possible codons encoding 20 amino acids and three stop translation codons, then we obtain a huge number of potential alternatives, about 1.51×10^{84}. It makes the question about the 'frozen' canonical genetic code among such enormous number of other possibilities very intriguing (Crick, 1968). There are three main theories trying to explain the origin and structure of the genetic code (see DiGiulio, 2005 for detailed review). However, none of them is unambiguously supported.

The first theory, called stereo-chemical, claims that some structural relationships and interactions between coded amino acids and stretches of RNA (e.g., codons, anticodons and reversed codons) (Dunnill, 1966; Pelc and Welton, 1966) were responsible for the present structure of the genetic code. So far, well confirmed such relationships were found for seven amino acids (see for review: Yarus et al., 2005). According to the physico-chemical (adaptive) theory (Freeland and Hurst, 1998; Gilis et al., 2001; Freeland et al., 2003), the canonical genetic code is optimized to minimize deleterious effects of mutations and errors occurring during protein synthesis (translation). The level of its adaptation can be measured by harmful effects of the replacement of one amino acid to another (Haig and Hurst, 1991). The coevolution hypothesis states that codons in the ancestral genetic code encoded only a small subset of amino acids and later, along with the evolution of biochemical organization of primary cells, newly synthesized amino acids took

* Corresponding author.
E-mail address: pamac@smorfland.uni.wroc.pl (P. Mackiewicz).

http://dx.doi.org/10.1016/j.biosystems.2016.08.008
0303-2647/

over the codons from the amino acids to which they were related in the biosynthetic pathways (Wong, 1975, 2005; Taylor and Coates, 1989; Di Giulio, 1991, 1989, 2016). Since the newly emerged amino acids as well as the taken codons were similar to their precursors this concept also explains why the genetic code can reflect an optimization in respect to translational errors.

The problem of genetic code optimization was investigated by many authors using two approaches: the statistical one (Freeland et al., 2000; Mackiewicz et al., 2008), which compares the canonical genetic code with many randomly generated alternatives, and the engineering method (Di Giulio, 2000), which compares the canonical code with the computationally optimized alternative. However, the large number of possible genetic codes makes it difficult to search the space of potential genetic codes. Therefore, the idea of applying adapted evolutionary-based algorithms (EA) seems very useful in solving this problem and is promising in a further research on general properties of the genetic code (Santos and Monteagudo, 2010, 2011). This proposal allowed for better location of the canonical genetic code in the fitness landscape and calculation of its distance to the optimized code.

The EA approaches are based on mutation and crossover operators. The mutation operator is indispensable in every evolutionary based algorithm because it is responsible mainly for introducing new information into the population of candidate solutions. The effectiveness of this algorithm can be improved by applying a crossover operator. This operator is used to create new individuals (offspring) based on existing solutions (parents). As a result, newly created individuals can often inherit good parts of their parents and therefore can be better and quicker adapted. It results from the fact that parent individuals are not random because they are examined by a selection process in the preceding simulation step. Consequently, the crossover mutation operators are jointly responsible for random changes in the population of candidate solutions and drive the computational evolution.

The different properties of these operators makes that each of them introduces its own variation. Therefore, it seems that the inclusion both of them should generally enhance the effectiveness of searching for possible solutions. However, potential benefits of using crossover operator depend on the kind of optimization problem (Fogel and Atmar, 1990; Spears, 1992, 1994; Park and Carter, 1995; Kokosiński, 2005). Thus, it is reasonable to test the influence of crossover operator on the effectiveness of evolutionary algorithm in every considered model and develop operators that are specific for a given problem.

Therefore, in this work, we adapted the position based crossover operator for two models of the genetic code and proposed a new operator for another model. We studied the performance of the algorithms for different combinations of mutation and crossover probabilities. Based on this large item of data, we were able to test with statistical significance the potential impact of these parameters' values on the quality of the optimization process. Thanks to this extensive search we were also able to evaluate the most optimal genetic codes found in these simulations and compare them with the canonical one.

2. Methods

2.1. Mutation and crossover operators

In the previous attempts to solve the problem of the genetic code optimality, different types of mutation operators were used (see Santos and Monteagudo, 2010, 2011 for details). Their usage depended on restrictions on the genetic code structure. However, the authors did not use any type of crossover operator. Furthermore, they emphasized that the classical crossover operators do not guarantee that all amino acids are always represented in the derived genetic codes (offspring) (Santos and Monteagudo, 2010). To deal with this problem, we adapted an already known crossover operator and also proposed a new one. We tested their quality under three restrictions (models) in searching the space of genetic codes:

1. Canonical structure 1 (CS1), which preserves the characteristic structure of codon blocks and degeneracy of the canonical genetic code. To generate potential codes, we permuted the assignment of amino acids between the codon blocks.
2. Canonical structure 2 (CS2), which preserves the number of codons per amino acid as in the canonical code. To generate potential codes, we permuted the assignment of codons to amino acids disregarding the codon blocks structure. By comparing results obtained for CS2 and CS1, we can test the importance of the characteristic codon blocks' structure with maintained degeneracy of the canonical genetic code.
3. Unrestricted structure (US), which has no constraints on the genetic code structure but assumes that every amino acid should be coded by at least one codon. To generate potential codes, we randomly divided 61 codons into 20 non-overlapping sets.

For all the described models, we claimed that stop codons remained invariant during all simulations and stayed the same as in the canonical code.

In the case of CS1, we adapted the position based crossover (POS) operator (Syswerda, 1991). A similar procedure is used in an evolutionary-based approach to the travelling salesman problem (Larrañaga et al., 1999). The POS draws amino acids from the parental codes at random and assigns them to the corresponding codon blocks in the offspring (Fig. 1A). The remaining codon blocks have amino acids assigned in the order of the other parent. When an amino acid is already present in the offspring, the other one is selected according to its position in the vector of amino acids (Fig. 1B). It ensures that every amino acid in the offspring is assigned only to one codon block.

However, this operator cannot be directly used in the CS2 and US models because the possible offspring might not inherit the proper structure of its parents. In this case, the generated genetic codes might not code all 20 amino acids. Therefore, we had to introduce another version of crossover operator (Fig. 2), according to the following procedure:

1. We create offspring O_1 and O_2, which are identical to their parents P_1 and P_2.
2. We select randomly an amino acid a_i, the same for the two parents, coded by parental codon blocks C_1 and C_2, respectively.
3. We compare the blocks and recognize the set of codons present in both parents, i.e., $U = C_1 \cap C_2$ as well as sets of codons present in one parent and absent in the other, i.e., $S_1 = C_1 \setminus U$ and $S_2 = C_2 \setminus U$ such that the condition $S_1 \cap S_2 = \emptyset$ is fulfilled.
4. The codons that are the same in the two parental codon blocks, i.e., $c_i \in U$, are not exchanged (Fig. 2A).
5. In the case of the sets S_1 and S_2, we choose at random codons $c_i \in S_1$ and $c_j \in S_2$ and exchange them between offspring O_1 and O_2 (Fig. 2B). To keep the original set of all codons represented by only one item, the codon exchange is realized by the swap of corresponding codons within a given offspring. Thanks to that, the individual that donated a codon does not loose it, whereas the offspring obtaining the codon has it assigned only once. The exchanged codons c_i and c_j are then removed from S_1 and S_2. This procedure is repeated until there are no codons left in S_1 or S_2 for selection.
6. When, for example, $S_1 = \emptyset$ and $S_2 \neq \emptyset$, there are no codons for mutual exchange. Then a codon, here $c_j \in S_2$, is moved to the