ISSN: 1803-3814 (Printed), 2571-3701 (Online)
https://doi.org/10.13164/mendel.2021.2.012

Advances in Evolutionary Optimization of Quantum Operators

Petr Zufan✉, **Michal Bidlo**

Brno University of Technology, FIT, IT4Innovations Center of Excellence, Brno, Czech Republic

izufan@fit.vut.cz✉, bidlom@fit.vut.cz

Abstract

A comparative study is presented regarding the evolutionary design of quantum operators in the form of unitary matrices. Three existing techniques (representations) which allow generating unitary matrices are used in various evolutionary algorithms in order to optimize their coefficients. The objective is to obtain as precise quantum operators (the resulting unitary matrices) as possible for given quantum transformations. Ordinary evolution strategy, self-adaptive evolution strategy and differential evolution are applied with various settings as the optimization algorithms for the quantum operators. These algorithms are evaluated on the tasks of designing quantum operators for the 3-qubit and 4-qubit maximum amplitude detector and a solver of a logic function of three variables in conjunctive normal form. These tasks require unitary matrices of various sizes. It will be demonstrated that the self-adaptive evolution strategy and differential evolution are able to produce remarkably better results than the ordinary evolution strategy. Moreover, the results can be improved by selecting a proper settings for the evolution as presented by a comparative evaluation.

Keywords: *Evolution Strategy, Differential Evolution, Self-adaptation of Control Parameters, Quantum Operator, Unitary Matrix.*

Received: 06 November 2021
Accepted: 15 December 2021
Published: 21 December 2021

1 Introduction

During recent years, various aspects of the modern computer science are more and more influenced by principles developed at the turn of the 19th and 20th century in the field that is currently known as quantum physics. In the second half of the 20th century, a new concept of computation emerged from its ideas that is referred to as quantum computing. Despite some cardinal obstacles, various advances in modern technology led to the development of a device that can be considered as the first quantum computer. Although only a very simple problem was demonstrated to be solved that time, it has shown as a new (potentially revolutionary) computing paradigm [6]. Nowadays quantum computing is considered as one of the most challenging but very promising area which could have serious impacts in various fields of information technology (e.g. cryptography and secure communication or data compression) [17]. Despite the fact that there are very few quantum computing devices available so far, the underlying principles of quantum computing have been well understood mathematically. One of the key ideas for manipulating states of a quantum system (and hence for describing quantum algorithms) is the concept of unitary matrices. Some of them have been defined rigorously for elementary operations and are referred to as quantum operators (or quantum gates). As an analogy to instructions known from classical computers, the quantum operators constitute building blocks for creating more complex quantum algorithms. Nevertheless, the design of even simple quantum algorithms represents a non-trivial task especially due to the stochastic nature of quantum phenomena and different (quantum) computing paradigm. However, the design of quantum operators (algorithms) in the form of unitary matrices may be considered as an optimization task (performed on ordinary computers) for which various techniques can be applied.

In the previous work [3], a comparative study of designing quantum operators was proposed considering various representations of unitary matrices (the mathematical means of representing quantum operators) the parameters of which were optimized by selected evolutionary algorithms. For example, a technique called QR decomposition [7] was applied in [3] for the first time on designing quantum operators by means of evolutionary algorithms. This approach was compared to the representations of Greenwood et al. [11] and MacKinnon [16] which have also shown a potential for the evolutionary design of quantum operators. The initial study proposed in [3] has shown that genetic algorithm and evolution strategy (both applied in various setups) are able to automatically discover parameters for the given representations providing acceptable solutions for some basic as well as more advanced quantum operators.

In this article, a continuation of this research will be presented utilizing some advanced evolutionary techniques which allow further optimizing the precision of the resulting solutions. Specifically, the adaptive evolution strategy and differential evolution algorithms will be applied for the first time in order to design quantum operators using the given representations. These techniques will be evaluated with a wider range of settings in order to determine their ability to provide solutions of a high precision. It will be shown that a variant of differential evolution is able to significantly outperform most of the other techniques regarding the quality of resulting unitary matrices. Moreover, the adaptive evolution strategy will be demonstrated to be able to generally improve the results in comparison with its ordinary variant. Finally, the results have also highlight the MacKinnon's representation as probably the most suitable technique for generating unitary matrices in combination with most of considered experimental setups which may be beneficial for further research in this area.

2 Related work

In recent years, there has been an ongoing research related to both theoretical and application areas of quantum computing. In this section, we summarize some selected studies related to the utilization of evolutionary techniques for optimizing various aspects of quantum computing.

A survey of evolutionary algorithms applied to evolve quantum algorithms until 2009 can be found in [8]. One of the first Genetic Algorithm-based approach to quantum computing was presented in [22] and focused on designing algorithms as alternative hardware configurations for special purpose quantum computers. Reid presented a method for designing quantum circuits by means of Genetic programming [20]. His experiments discovered simple as well as advanced functions composed of up to several tens of elementary quantum gates. More recently, Drechsler et al. proposed a Genetic Programming-based approach for a multi-objective synthesis of quantum circuits [21]. Hutsell and Greenwood proposed probably for the first time a method for evolutionary generation of quantum operators in the form of unitary matrices [11] (more details about this method will be given in Section 4.2). The authors applied Evolution Strategy to find solution for some elementary quantum gates, quantum oracles as well as more generalized operations. This method was improved by Mackinnon in [16]. The author introduced several modification to the original approach in order to reduce the number of matrices that needed to be multiplied. The goal was to achieve a more flexible representation and to improve the convergence of the evolutionary search (more details will be given in Section 4.3). Later, the MacKinnon's work was an inspiration to Gregor [9] who proposed some new aspects for the evolution of various quantum algorithms, some with the register size up to 6 bits (e.g. identity operator or amplification of element with the maximum amplitude). Krawec followed the work of Hutsell and Greenwood [11] and proposed a Genetic Algorithm with real-coded values to evolve collections of unitary matrices which act on pure or mixed states of arbitrary quantum systems while interacting with fixed, problem specific quantum operators (e.g., oracle calls) and intermediate partial measurements [12]. Bang and Yoo presented a method based on Genetic Algorithm for optimizing unitary transformations with a generalization of the resulting quantum algorithms for a more realistic problem – the one-bit oracle decision problem (also referred to as Deutsch problem) [2]. From a more general perspective, there are other works dealing with evolutionary approaches applied on various aspects of quantum systems or even utilizing quantum principles in order to influence the functioning of evolutionary algorithms themselves. Let's mention some of them here. Various quantum-inspired Differential Evolution algorithms were presented in [10] and [14] for solving minimization problems (e.g. 0–1 knapsack problem). Caires and Noronha applied Genetic Algorithm to synthesize robust circuits based on the Quantum-Dot Cellular Automata concept [5]. Krylov and Lukac presented a Quantum Encoded Quantum Evolutionary Algorithm and successfully evaluated its properties on the design of several reversible and quantum circuits [13]. Szwarcman et al. applied a quantum-inspired algorithm to search for deep neural architectures by assembling substructures and optimizing some numerical hyperparameters [23]. A possibility of integration of quantum entanglement and quantum NOT operator with the well-known Differential Evolution algorithm was studied in [15]. Recently, an approach utilizing the principle of memetic computing was utilized to develop Memetic Quantum Evolutionary Algorithm for solving the global optimization problem [24] and an implementation of an evolutionary optimization framework using a hybrid hardware architecture, where classical processors interact with the family of quantum processors, was presented in [1].

3 Quantum computing fundamentals

In classical computing, the state of a system can be represented (in general) by a bit vector the values of which are either 0 or 1 (e.g. the contents of each bit of a computer memory). In quantum computing things are slightly different. The state of a system consists of qubits (quantum bits). A qubit is not only in a state 0 or 1 but in a superposition of both of them. That means that a qubit is in a state 0 with some probability p_0 and at the same time in a state 1 with some probability p_1. Must hold that

$$p_0 + p_1 = 1 \tag{1}$$

More precisely, probabilities of a qubit are represented by complex numbers called probability amplitudes. The probability is given as $|\alpha|^2$ of the given probability amplitude α and then the equation

$$|\alpha_0|^2 + |\alpha_1|^2 = 1 \tag{2}$$

where $|\alpha_0|^2$ respectively $|\alpha_1|^2$ is probability of state 0 respectively 1, must hold.

Using the *ket* notation, that is usual in quantum computing [17], the quantum alternative for classical 0 state is denoted as $|0\rangle$ and for 1 state is denoted as $|1\rangle$. Arbitrary qubit state $|a\rangle$ with probability amplitudes α_0 and α_1 is then denoted as $|a\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$.

For the process of mathematical expression, the *ket* vector $|a\rangle$ is a column vector $(\alpha_0, \alpha_1)^T$ and similarly $\alpha_0|0\rangle = \alpha_0(1,0)^T$ and $\alpha_1|1\rangle = \alpha_1(0,1)^T$.

More complex (n-bit) systems are described by 2^n-element vectors that are, in terms of quantum computing, expressed as *tensor product* ($\otimes$) of state vectors of individual qubits. For example, a 2-qubit register of qubits $|a\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ and $|b\rangle = \beta_0|0\rangle + \beta_1|1\rangle$, can be expressed as

$$|a\rangle \otimes |b\rangle = \begin{pmatrix}\alpha_0\\ \alpha_1\end{pmatrix} \otimes \begin{pmatrix}\beta_0\\ \beta_1\end{pmatrix} = \begin{pmatrix}\alpha_0 \cdot \begin{pmatrix}\beta_0\\ \beta_1\end{pmatrix}\\ \alpha_1 \cdot \begin{pmatrix}\beta_0\\ \beta_1\end{pmatrix}\end{pmatrix} = \begin{pmatrix}\alpha_0\beta_0\\ \alpha_0\beta_1\\ \alpha_1\beta_0\\ \alpha_1\beta_1\end{pmatrix}. \tag{3}$$

For all 2-bit combinations of classical bits we obtain from (3)

for $\alpha_0 = 1, \alpha_1 = 0, \beta_0 = 1, \beta_1 = 0$:

state $|00\rangle = (1,0,0,0)^T$,

for $\alpha_0 = 1, \alpha_1 = 0, \beta_0 = 0, \beta_1 = 1$:

state $|01\rangle = (0,1,0,0)^T$,

for $\alpha_0 = 0, \alpha_1 = 1, \beta_0 = 1, \beta_1 = 0$:

state $|10\rangle = (0,0,1,0)^T$,

for $\alpha_0 = 0, \alpha_1 = 1, \beta_0 = 0, \beta_1 = 1$:

state $|11\rangle = (0,0,0,1)^T$,

Remember that a qubit is in a *superposition* of states $|0\rangle$ and $|1\rangle$, that is interpreted as being in both states simultaneously which is typical for quantum systems. The probabilities then decide in the process of measurement about observing the "logic 0" e.g. with the probability $|\alpha_0|^2$ or "logic 1" with the probability $|\alpha_1|^2$. Therefore, quantum systems are inherently non-deterministic.

A transformation of a state vector from one state to another (i.e. a computation) is mathematically described as multiplication of the vector by a matrix representing an operator. For example, a NOT gate defined by the matrix X as

$$X = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix} \tag{4}$$

transforms state $|0\rangle$ to state $|1\rangle$ and vice versa. The multiplication of a state vector $|a\rangle$ by X results in the negated state. For example, the negation of a qubit $|a\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ is expressed as

$$X|a\rangle = X\left[\alpha_0\begin{pmatrix}1\\ 0\end{pmatrix} + \alpha_1\begin{pmatrix}0\\ 1\end{pmatrix}\right] = X\begin{pmatrix}\alpha_0\\ \alpha_1\end{pmatrix} = \\ = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix}\alpha_0\\ \alpha_1\end{pmatrix} = \begin{pmatrix}\alpha_1\\ \alpha_0\end{pmatrix}. \tag{5}$$

Notice that from state $(\alpha_0, \alpha_1)^T$ we obtained its negation which is $(\alpha_1, \alpha_0)^T = \alpha_1|0\rangle + \alpha_0|1\rangle$. A quantum operator must preserve an equation 2 on outcome quantum state. Therefore the matrix representing quantum operator must be a complex unitary matrix. The unitary matrix U has the property

$$UU^\dagger = I \tag{6}$$

, where $U^\dagger$ denotes a complex conjugate matrix of matrix U and I is an identity matrix. Some other quantum operators will be described in Section 5.1.

4 Evolutionary design of quantum operators

As mentioned in chapter 3, in quantum computing, any operation can be expressed by means of a unitary matrix U which corresponds to mathematical description of a quantum gate. A *quantum algorithm* thus may be described as a sequence of transformations of state vector by a quantum operators. Such transformations are expressed by a multiplication of a state vector and a operator's matrix. Because matrix multiplication is an associative operation, more complex quantum operators may be created by multiplying the matrices corresponding to the basic quantum operators [17]. This approach will be adopted in this paper as the representation of quantum operators for their design by means of evolutionary algorithms because of its simplicity and universality.

Since quantum algorithms do not comply with usual programming paradigms, their design usually represents a challenging tasks. Therefore, various unconventional approaches can often provide suitable resources to automate this process. In this paper we apply evolutionary algorithms (EA) to perform this task. This section describes the formulation of the problem and techniques used for its solution by means of EA.

4.1 Problem definition, evaluation and simulation of solutions

In this paper, the task to be solved will be defined as follows. Let $|i\rangle$ denote an initial (input) quantum state and $|o\rangle$ denote an output (target) state. The goal is to find an operator U for which $|o\rangle = U|i\rangle$. More generally, let $M = \{(|i_1\rangle, |o_1\rangle), (|i_2\rangle, |o_2\rangle), \ldots (|i_p\rangle, |o_p\rangle)\}$ be a finite set of pairs *(initial state, output state)*. Then $|o_k\rangle = U|i_k\rangle$ is required to hold for all $k = 1, \ldots, p$.

In order to evaluate the quality of candidate solutions, the following objective function is considered to be minimized during evolution ($fitness(U) = 0$ for a perfect solution):

$$fitness(U) = \frac{\sum_{v=0}^{|M|}\sum_{w=0}^{N} \left|\, |o_v\rangle[w] - U|i_v\rangle[w] \,\right|}{|M|N}, \tag{7}$$

where N is the number of elements of the state vector, and $|i_v\rangle[w]$, $|o_v\rangle[w]$ denotes the w-th element of the

v-th state vector from the training set M. Since the elements (probabilities) of the state vectors of quantum systems are composed of complex numbers, it is usually not possible to achieve exact solutions. Hence a threshold value $\epsilon > 0$ is specified and candidate solutions whose $fitness < \epsilon$ are considered as acceptable solutions.

Since the real quantum hardware is still very rare, a simulator running on a common PC has been chosen for our experiments. In this paper, we utilized QuEST – Quantum Exact Simulation Toolkit[1] that provides necessary functions to perform the simulation of a quantum computer for the transformation of given state vectors by means of quantum operators specified in the form of unitary matrices. Moreover, it has a C++ interface allowing easy integration into our evolutionary system.

EA will be applied to find a suitable quantum operator U (considered as phenotype) and several techniques will be utilized to generate the phenotype from parameter vectors encoded in chromosomes of the EA. The following subsections will briefly describe these techniques.

4.2 Representation of Hutsell & Greenwood

This method for generating unitary matrices was originally developed by Zyczkowski and Kus in [26] and later adopted by Hutsell and Greenwood for the evolution of quantum operators in [11].

The method is based on the fact that an $N \times N$ unitary matrix U can be generated by means of $N-1$ composite rotations in complex subspaces by the equation 8.

$$U = e^{i\lambda} E_1 E_2 \dots E_{N-1} \tag{8}$$

where $e^{i\lambda}$ is the complex phase factor, $\lambda \in [0, 2\pi)$ and E_p, for $p = 1, \dots, N-1$ are $N \times N$ rotation matrices. Each such composite rotation matrix E_p is defined as a product of $p-1$ elementary rotation matrices F given by the equation

$$E_p = \prod_{q=1}^{p} F^{j,k}(\phi_{j,k}, \psi_{j,k}, \chi_{j,k}). \tag{9}$$

where $j = p - q + 1$, $k = p + 1$ and $\phi_{j,k}, \psi_{j,k}, \chi_{j,k}$ are parameters (rotation angles) of the elementary rotation matrix $F^{j,k}$. Finally the $F^{j,k}$ is a $N \times N$ matrix defined for each $j, k \in \{1, \dots N\}$ and $\phi \in \langle 0, \pi/2 \rangle$ and $\psi, \chi \in \langle 0, 2\pi \rangle$ as

$$F^{j,k}(\phi, \psi, \chi) = \begin{cases} cos(\phi)e^{i\psi} & \text{for } F[j,j] \\ sin(\phi)e^{i\chi} & \text{for } F[j,k] \\ -sin(\phi)e^{-i\chi} & \text{for } F[k,j] \\ cos(\phi)e^{-i\psi} & \text{for } F[k,k] \\ 1 & \text{otherwise} \end{cases} \tag{10}$$

[1] https://quest.qtechtheory.org/

where $F[j,k]$ denotes element of matrix F in the j-th row and k-th column. The $F^{j,k}$ matrix can be also seen as

$$F^{j,k}(\phi, \psi, \chi) = \begin{pmatrix} 1 & \dots & 1 & \dots & 1 \\ \vdots & cos(\phi)e^{i\psi} & \vdots & sin(\phi)e^{i\chi} & \vdots \\ 1 & \dots & 1 & \dots & 1 \\ \vdots & -sin(\phi)e^{-i\chi} & \vdots & cos(\phi)e^{-i\psi} & \vdots \\ 1 & \dots & 1 & \dots & 1 \end{pmatrix}. \tag{11}$$

Note that if $j \neq 1$, then $\chi_{j,k} = 0$ which reduces the number of χ angles needed. In total N^2 angles are needed: $(N-1)N/2$ of ϕ angles, $(N-1)N/2$ of ψ angles and $N-1$ of χ angles. These angles are the subject of evolution. A candidate solution is represented as a real-valued vector of N^2 values given as:

$$(\phi_{1,2}, \psi_{1,2}, \chi_{1,2}, \dots \phi_{N-1,N}, \psi_{N-1,N}). \tag{12}$$

4.3 Representation of MacKinnon

As the second representation, we adopted MacKinnon's idea proposed in [16]. According to that, the action of an $N \times N$ unitary matrix which is decomposed as such on a N-dimensional state vector can be simulated sequentially as the action of each elementary unitary operator, where the action of an elementary operator on the $j-k$ subspace is simulated by taking the j-th and k-th elements from the input vector, combining these two elements into a 2-dimensional vector, left-multiplying this vector by the corresponding 2×2 special unitary matrix, and then reinserting the updated elements into their respective positions in the input vector. Once this has been done for all of the $N(N-1)/2$ elementary unitary operators, then the entire vector is multiplied by the complex phase factor [16].

Formally [16], any 2×2 special unitary matrix can be written in the form $U = a_0 I + i(a_1 X + a_2 Y + a_3 Z)$ where i is the complex unit, $a_0, a_1, a_2, a_3 \in [0, 1]$ are parameters constituting a real-valued vector of norm 1, I is the identity matrix, and X, Y and Z are the Pauli matrices. The proof can be found in [16]

The chromosome for an evolved unitary operator thus contains $4N(N-1)/2 = 2N(N-1)$ real parameters from $[0, 1]$ and one real parameter from $[0, 2\pi)$ for the complex phase coefficient [16].

4.4 Representation using QR decomposition

The last technique considered in this paper for generating unitary matrices is the QR decomposition. It was utilized for the first time by Bidlo and Zufan in [3] for the evolutionary design of quantum operators and in this article it is used in more advanced evolutionary techniques.

Following the definition in [25] and adapting it to square matrices for the purposes of this paper, a $N \times N$ matrix A with complex entries and $rank\,(A) = N$ may be decomposed to $A = QR$ where Q is a unitary matrix and R is an upper triangular matrix. This means that we can generate unitary matrices from an input matrix A by finding Q using this method. Specifically, we encode A as a sequence of $N \times N$ complex numbers in a chromosome that is the subject of evolution, perform the QR decomposition and evaluate Q as a candidate (unitary) quantum operator for a given task.

There exist several algorithms for the QR decomposition, we utilized the Householder method from [19] that works as follows. We can rewrite[2] $A = QR \Rightarrow Q^\dagger A = R$, then $Q^\dagger = H_n H_{n-1} \dots H_1$ where H_i is a Householder matrix that zeroes elements below the i-th row of a column vector $\mathbf{u}$: $H_i \mathbf{u} = (v_1, \dots, v_i, 0, \dots, 0)^T$ (here T is the transposition). Finally Q is obtained simply from its adjoint.

5 Evolutionary system setups

The experimental setups in this paper follow our previous research in this ares that was presented in [3]. Several advance evolutionary techniques are applied in combination with the representations of quantum operators described in Section 4.1 for the design of more complex case studies. In particular, differential evolution (DE) and self-adaptive evolution strategy (SAES) are introduced in this paper in order to show their potential to improve the previous results and to design more complex quantum operators. These evolutionary algorithms and their setups are described in detail in section 5.2. The results are compared with those obtained by means of ordinary evolution strategy (ES) that will be considered as a reference evolutionary algorithm (based on the results obtained in [3]).

The aforementioned evolutionary techniques (ES, DE and SAES) are evaluated on three case studies: the 3-qubit amplitude detector (Max3), the 4-qubit maximum amplitude detector (Max4) and the 3-qubit solver of a logic functions in conjunctive normal form (Cnf3). The description of these case studies is provided in section 5.1. Results of all experiments are presented in chapter 6.

5.1 Case studies

Three quantum problems will be investigated in the experiments. As described in section 4.1 the problem is given by a set of pairs (the input quantum state and a desired output quantum state).

The first one is the **3-qubit maximum amplitude detector** (Max3). In this experiment we want to maximize the highest amplitude of the input state. Let us describe it on a 2-qubit example. Let $\mathbf{v_i} = \left(\frac{1}{2}, 0, \frac{1}{\sqrt{2}}, \frac{1}{2}\right)^T$ be an input state vector in which the values express various amplitudes of a wave function. The goal is to evolve such an operator U by means of which an output vector $\mathbf{v_o}$ is calculated that has 1 at the position where $\mathbf{v_i}$ has the maximum value and with all other components at 0. For the given $\mathbf{v_i}$ we obtain $\mathbf{v_o} = U\mathbf{v_i} = (0, 0, 1, 0)^T$. Analogically this operator works for more qubit systems (in this article we consider the **4-qubit maximum amplitude detector** - Max4).

[2] By multiplying both sides by $Q^\dagger$ which is the adjoint of Q and since $QQ^\dagger = I$ for unitary Q.

The evolution of Max3 and Max4 is performed in such a way that a random input quantum state is generated at the beginning of evolution which, together with the corresponding target states, constitutes the training set M. The candidate solutions are then evaluated according to (7).

The last case study considered in this paper is a **solver of a logic function of three variables in CNF** (Cnf3). We choose the common encoding in which N qubits are needed for N boolean variables as follows. First, encoding of a boolean function into the input state vector is described. Let's begin with a boolean expression of three variables and negation and disjunction operators only, for example $(v_1 \vee \overline{v_2} \vee v_3)$. We can encode this expression as triple of 0s and 1s, where v_i is encoded as 0 when is negated and as 1 otherwise. For example, the expression $(v_1 \vee \overline{v_2} \vee v_3)$ will be represented as (101). Such that triple is then mapped to a corresponding amplitude of 3-qubit vector state (in this case $|101\rangle$).

This way we encode all boolean expressions of a logic function in CNF into set of pure vector states. Finally all such pure vector states are put into superposition. To describe it on example, let have a boolean function

$$f = (a \vee b \vee c) \wedge (\overline{a} \vee b \vee \overline{c}) \wedge (a \vee \overline{b} \vee \overline{c}). \qquad (13)$$

The result set is $\{(111), (010), (100)\}$ and finally the quantum state in *ket* notation is

$$|f\rangle = \frac{1}{\sqrt{3}}\big(|010\rangle + |100\rangle + |111\rangle\big) + 0\big(|000\rangle + |001\rangle + |011\rangle + |101\rangle + |110\rangle\big) \qquad (14)$$

or written as a vector

$$|f\rangle = \big(0, 0, \frac{1}{\sqrt{3}}, 0, \frac{1}{\sqrt{3}}, 0, 0, \frac{1}{\sqrt{3}}\big)^T. \qquad (15)$$

Similarly, a set of solutions of the input boolean function are encoded into the superposition of the corresponding pure quantum states. For the above example function f, the results are

$r = \{(1,1,1), (1,1,0), (1,0,0), (0,1,0), (0,0,1)\}$ and output state is

$$|r\rangle = \frac{1}{\sqrt{5}}\big(|001\rangle + |010\rangle + |100\rangle + |110\rangle + |111\rangle\big) \qquad (16)$$

in *ket* or

$$|r\rangle = \big(0, \frac{1}{\sqrt{5}}, \frac{1}{\sqrt{5}}, 0, \frac{1}{\sqrt{5}}, 0, \frac{1}{\sqrt{5}}, \frac{1}{\sqrt{5}}\big)^T. \qquad (17)$$

as a vector.

At the beginning of the evolution a logic function with output results is given. Together they form a training set. The evolutionary algorithm then searches for a quantum operator providing a solution for the training set.

Table 1: The best fitness value out of 108 runs in each set of experiments. The best in the row is marked bold.

Evolutionary algorithm	**ES 1+20**	**ES 8+16**	**ES 15+100**	**SAES 1+20**	**SAES 8+16**	**SAES 15+100**	**DE 20**	**DE 50**	**DE 100**
Operator + Represent.									
Max3+HG	0.043525	0.050134	0.042193	0.000261	0.000033	0.000034	**0.000000**	0.001082	0.019887
Max3+M	0.023308	0.019021	0.012147	0.000013	0.000004	0.000004	**0.000000**	0.002452	0.008233
Max3+QR	0.040377	0.042927	0.036060	0.059732	0.006954	0.001373	**0.000003**	0.004319	0.011611
Max4+HG	0.096709	0.099597	0.087783	0.136431	0.024238	0.014177	**0.000002**	0.141535	0.122623
Max4+M	0.042302	0.030345	0.029083	0.015321	0.000012	0.000011	**0.000000**	0.005087	0.014415
Max4+QR	0.079707	0.074307	0.075804	0.117730	0.102446	0.095670	**0.000004**	0.010806	0.016960
Cnf3+HG	0.059693	0.056286	0.051927	0.007409	0.000035	0.000038	**0.000004**	0.113955	0.103182
Cnf3+M	0.036600	0.027365	0.026261	0.000019	0.000005	0.000005	**0.000001**	0.011925	0.028561
Cnf3+QR	0.043015	0.036617	0.044502	0.073774	0.015431	0.003695	**0.000009**	0.006686	0.047775

5.2 Evolutionary algorithms

In this study we experimented with the techniques for generating unitary matrices as described in Sections 4.2, 4.3 and 4.4. Each technique was applied to the design of quantum operators described in Section 5.1. The search for the suitable parameters for the generation of the appropriate unitary matrix was performed using three evolutionary algorithms (EA): the ordinary evolution strategy (ES), self-adaptive [18] evolution strategy (SAES) and differential evolution (DE). A wide range of experiments was performed in order to find a suitable EA settings. In this paper we present 3 setups for each ES to show how they affect the result.

The first presented algorithm, the ordinary evolution strategy, was taken from the previous paper for the comparison purposes. The ES was implemented according to [4] considering setups with (1+20), (8+16) and (15+100) individuals. The mutation operator performs on each parameter x_i in a chromosome such that $x_{i,mutated} = x_i + \sigma N(0,1)$ with the mutation control parameter $\sigma = 0.3$.

For the advanced experiments presented in this article, the ES was extended by self-adaptation of the mutation control parameters – here referred to as self-adaptive evolution strategy (SAES). The same population setups are considered, i.e. (1+20), (8+16), (15+100). The mutation in SAES is slightly different. An independent mutation parameter σ_i is introduced for each parameter x_i in a chromosome $(x_1, ..x_n)$ and values of these σs are adapted during evolution. The mutation operator then works in two steps:

$$\sigma_i = \sigma_i \cdot e^{\tau' r' + \tau r} \tag{18}$$

$$x_i = x_i + \sigma_i \cdot r_i, \tag{19}$$

where $\tau' = 1/\sqrt{n}$, $\tau = 1/\sqrt{2\sqrt{n}}$ and r', r and r_i are independent random variables drawn from $N(0,1)$. These steps must be performed in the given order. First, update the σ_i parameter and then update x_i using the new σ_i.

The third evolutionary technique investigated herein is the differential evolution (DE). The DE is implemented according to [4] utilizing the setup DE/rand/1/bin. It means that the base individual is selected randomly and mutated by the addition of the single difference vector (with the scale factor $F = 0.5$). Finally, the binary crossover is utilized with crossover probability $r_c = 0.7$. DE was considered in 3 variants with the population size 20, 50 and 100 individuals.

The termination condition for each of the EAs was considered as a maximum number of fitness evaluations. For the purposes of this article, the maximum number of the fitness evaluations was determined experimentally for each case study and set to 320000 for Max3 and Cnf3 and to 800000 for Max4. If a solution with the fitness value less than 0.05 has been found within the given limit, then the evolutionary run is considered as successful (the statistical evaluation of the experiments with respect to this condition will be given in the next section).

6 Experimental results

This section contains the experimental results and their statistical evaluation. Recall that we performed evolutionary design of quantum operators for three case studies (Max3, Max4 and Cnf3) in the form of unitary matrices generated by means of three different techniques (Hutsell & Greenwood, MacKinnon and QR Decomposition) the parameters of which have been optimized using three evolutionary techniques (ES, SAES and DE). Each of the EAs has been considered in three different settings regarding the population size and the number of offspring. For each of those experimental setups we performed a set of 108 independent experiments (evolutionary runs) with the maximum number of fitness evaluations as given in the last paragraph of Section 5.2.

Table 2: The percentage of successful runs out of 108 runs in each set of experiments. The run is considered successful if a solution has been found with the fitness value less than 0.05. The best in the row is marked bold.

Evolutionary algorithm	**ES 1+20**	**ES 8+16**	**ES 15+100**	**SAES 1+20**	**SAES 8+16**	**SAES 15+100**	**DE 20**	**DE 50**	**DE 100**
Operator + Represent.									
Max3+HG	0.93	0	2.78	17.59	67.59	**79.63**	41.66	10, 19	6.48
Max3+M	58.33	96.29	**100**	67.59	72.22	85.19	90.74	87.04	58.33
Max3+QR	2.78	4.63	10.19	0	14.81	27.78	80.55	**82.41**	17.59
Max4+HG	0	0	0	0	0.93	2.77	**5.56**	0	0
Max4+M	1.85	19.44	24.07	18.52	78.70	**79.63**	39.81	25.93	18.52
Max4+QR	0	0	0	0	0	0	**96.30**	44.44	23.15
Cnf3+HG	0	0	0	5.47	78.13	**94.53**	25	0	0
Cnf3+M	25.78	95.31	91.41	85.94	91.41	95.31	**100**	66.41	5.47
Cnf3+QR	6.25	14.84	8.59	0	7.81	25.78	**100**	60.16	0.78

First, we analyzed the best results obtained from each set of experiment in order to determine which of the settings is able to provide the most accurate unitary matrix for the given case study (i.e. the quantum operator). These results are summarized in Table 1. As can be seen, the differential evolution with 20 individuals in population provides the best results for all the considered quantum operators and representations. Its precision significantly outperforms the results from ES and also most of the results obtained from SAES. Although the differences in the achieved fitness values are very small for DE20, Table 1 also shows that from a more general point of view considering the representation techniques, the MacKinnon's representation exhibits the best results for nearly all of the quantum operators (regardless of the EA used). The *only* exception is Cnf3 and DE50 where the QR Decomposition provided the most accurate result. Therefore, it is possible to conclude that the MacKinnon's representation and differential evolution could be the most suitable evolutionary setup for the design of quantum operators in a wider sense (at least for initial experiments).

The second evaluation of the EAs was performed from the point of view of the success rates, i.e. the percentage of runs from each experimental set that are able to provide a solution with the fitness better than 0.05 – we considered this value as a threshold for the acceptable operator accuracy. The results of this evaluation are summarized in Table 2. As evident, a significant number of setups (specifically 20 out of 81) has not been able to provide any acceptable solution. The worst results can be observed in case of ordinary ES which has failed in 10 setups. However, this might be expectable because the ES was chosen as a basic EA with the goal to improve it. This improvement has been achieved by differential evolution which exhibits the lowest number of unsuccessful setups and also the best (most accurate) results obtained in most of the experiments.

Further, a statistical evaluation using box-plots has been performed for all the case studies, representations and evolutionary algorithms. The results are presented separately for each quantum operator Max3, Max4 and Cnf3 in Figure 1, 2 and 3, respectively. It may be observed that both the median and the best values of the fitness decrease (improve) in the considered setups of SAES and DE. However, whilst DE exhibit better results for the lowest population size – see right columns (parts c, f and i) of Figures 1, 2 and 3, this trend is rather opposite in case of SAES – see the middle columns (parts b, e and h) of Figures 1, 2 and 3. The left columns (parts a, d and g) of Figures 1, 2 and 3 also show that the results of ordinary ES are the worst in comparison with the other evolutionary algorithms.

Finally, we performed a comparison of the best results from the best performing setups of each of the EAs as shown in Figure 4. Notice that the green bar, corresponding to the best EA – DE20, is almost invisible as the best fitness values for this EA are very close to 0 (i.e. DE20 provides very precise solutions; the improvement against most of the other setups are by several orders of magnitude).

7 Conclusions

This article presented a comparative study of designing quantum operators by means of various representations (allowing generation of unitary matrices) and evolutionary algorithms. The goal was to obtain as precise unitary matrices as possible for various quantum transformations. The results showed that differential evolution is able to remarkably outperform the considered evolution strategies in solving this task. It proved to provide very precise solutions for all considered quantum operators and using all representations for generating their unitary matrices. Specifically, the difference of the best results provided by the differential evolution with 20 individuals in the population is several orders of magnitude in comparison with most of the other evolutionary setups. This indicates that the utilization of the differential evolution may be suitable for designing quantum operators in a wider sense. Moreover, the results suggested a suitability of the MacKinnon representation for designing the quantum operators (mainly by means of adaptive evolution strategy and differential evolution). Therefore, these advanced evolutionary algorithms, the study of which was the main goal of this work, may be beneficial for further research in this area. For example, the design of more advanced quantum operators or the introduction of novel representations can be considered for further studies. These issues will represent the main ideas in our next research.

Acknowledgment: This work was supported by the internal BUT project FIT/FSI-J-21-7432 and by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ project (ID:90140).

References

[1] Acampora, G., and Vitiello, A. Implementing evolutionary optimization on actual quantum processors. *Information Sciences 575* (2021), 542–562.

[2] Bang, J., and Yoo, S. A genetic-algorithm-based method to find unitary transformations for any desired quantum computation and application to a one-bit oracle decision problem. *Journal of the Korean Physical Society 65*, 12 (2014), 2001–2008.

[3] Bidlo, M., and Zufan, P. On comparison of some representations for the evolution of quantum operators. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)* (2020), IEEE, pp. 2101–2108.

[4] Brabazon, A., O'Neill, M., and McGarraghy, S. *Natural computing algorithms*, vol. 554. Springer, 2015.

[5] Caires, L. F. V., Neto, O. P. V., and Noronha, T. F. Evolutionary synthesis of robust qca circuits. In *2013 IEEE Congress on Evolutionary Computation* (2013), IEEE, pp. 2802–2808.

[6] Chuang, I. L., Gershenfeld, N., and Kubinec, M. Experimental implementation of fast quantum searching. *Physical review letters 80*, 15 (1998), 3408.

[7] Gander, W. Algorithms for the qr decomposition. *Res. Rep 80*, 02 (1980), 1251–1268.

[8] Gepp, A., and Stocks, P. A review of procedures to evolve quantum algorithms. *Genetic programming and evolvable machines 10*, 2 (2009), 181–228.

[9] Gregor, C. *Construction of Unitary Matrices and Bounding Minimal Quantum Gate Fidelity Using Genetic Algorithms*. PhD thesis, The University of Guelph, Guelph, Ontario, CA, 2018.

[10] Hota, A. R., and Pat, A. An adaptive quantum-inspired differential evolution algorithm for 0–1 knapsack problem. In *2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC)* (2010), IEEE, pp. 703–708.

[11] Hutsell, S. R., and Greenwood, G. W. Applying evolutionary techniques to quantum computing problems. In *2007 IEEE Congress on Evolutionary Computation* (2007), IEEE, pp. 4081–4085.

[12] Krawec, W. O. An algorithm for evolving multiple quantum operators for arbitrary quantum computational problems. In *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation* (2014), pp. 59–60.

[13] Krylov, G., and Lukac, M. Quantum encoded quantum evolutionary algorithm for the design of quantum circuits. In *Proceedings of the 16th ACM International Conference on Computing Frontiers* (2019), pp. 220–225.

[14] Li, B., Li, P., et al. Quantum inspired differential evolution algorithm. *Open Journal of Optimization 4*, 02 (2015), 31.

[15] Li, K., Elsayed, S., Sarker, R., and Essam, D. Quantum differential evolution: an investigation. In *2019 IEEE Congress on Evolutionary Computation (CEC)* (2019), IEEE, pp. 3022–3029.

[16] MacKinnon, D. *Evolving Quantum Algorithms with Genetic Programming*. PhD thesis, University of Guelph, Guelph, Ontario, CA, 2017.

[17] Nielsen, M. A., and Chuang, I. *Quantum computation and quantum information*. Cambridge University Press, 2011.

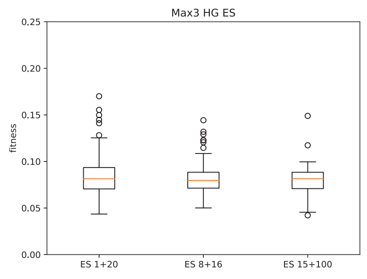
[18] Omran, M. G., Salman, A., and Engelbrecht, A. P. Self-adaptive differential evolution. In *International conference on computational and information science* (2005), Springer, pp. 192–199.

[19] Press, W. H., Flannery, B. P., Teukolsky, S. A., Vetterling, W. T., et al. Numerical recipes, 2007.

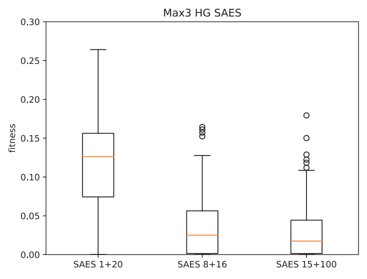
[20] Reid, T. On the evolutionary design of quantum circuits. Master's thesis, University of Waterloo, Ontario, CA, 2005.

[21] Sarvaghad-Moghaddam, M., Niemann, P., and Drechsler, R. Multi-objective synthesis of quantum circuits using genetic programming. In *International Conference on Reversible Computation* (2018), Springer, pp. 220–227.

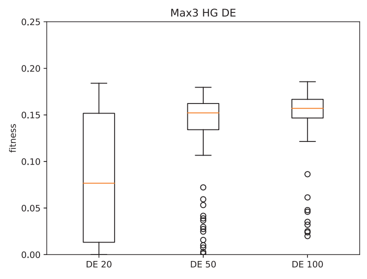
[22] Surkan, A. J., and Khuskivadze, A. Evolution of quantum computer algorithms from reversible operators. In *Proceedings 2002 NASA/DoD Conference on Evolvable Hardware* (2002), IEEE, pp. 186–187.



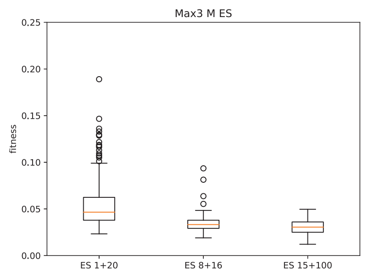
(a) ES on the HG representation



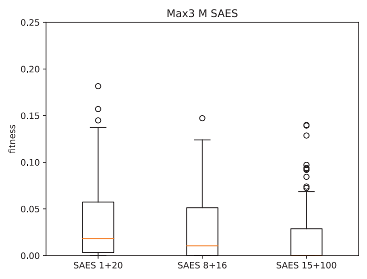
(b) SAES on the HG representation



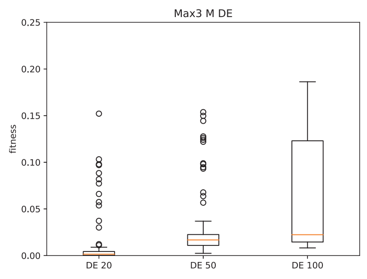
(c) DE on the HG representation



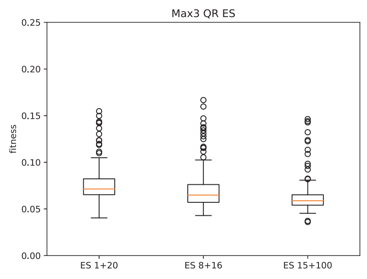
(d) ES on the M representation



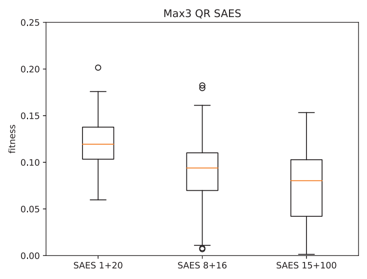
(e) SAES on the M representation



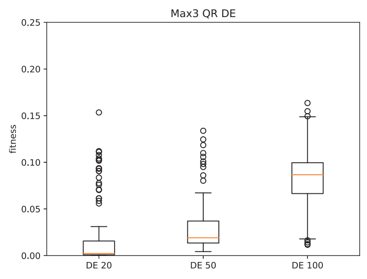
(f) DE on the M representation



(g) ES on the QR representation



(h) SAES on the QR representation



(i) DE on the QR representation

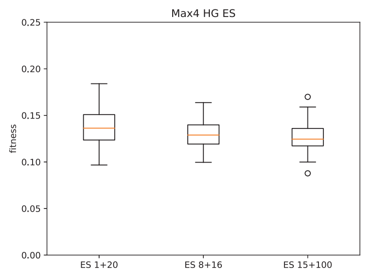
Figure 1: Statistical results from the evolution of the Max3 experiment.

[23] Szwarcman, D., Civitarese, D., and Vellasco, M. Quantum-inspired neural architecture search. In *2019 International Joint Conference on Neural Networks (IJCNN)* (2019), IEEE, pp. 1–8.

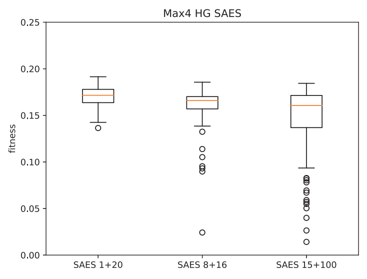
[24] Tang, D., Liu, Z., Zhao, J., Dong, S., and Cai, Y. Memetic quantum evolution algorithm for global optimization. *Neural Computing and Applications* (2019), 1–31.

[25] Van Loan, C. F., and Golub, G. Matrix computations (johns hopkins studies in mathematical sciences).

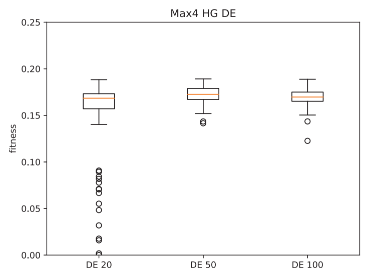
[26] Zyczkowski, K., and Kus, M. Random unitary matrices. *Journal of Physics A: Mathematical and General 27*, 12 (1994), 4235.



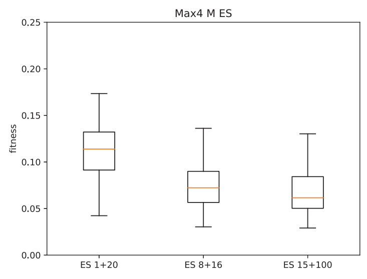
(a) ES on the HG representation



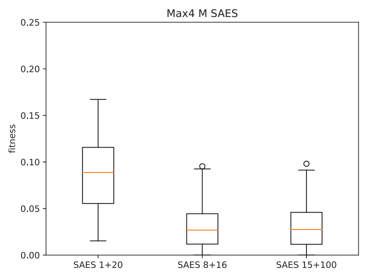
(b) SAES on the HG representation



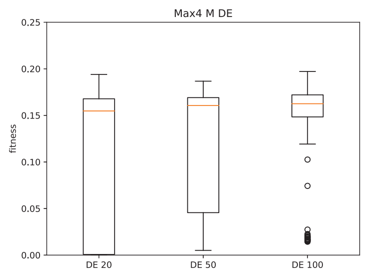
(c) DE on the HG representation



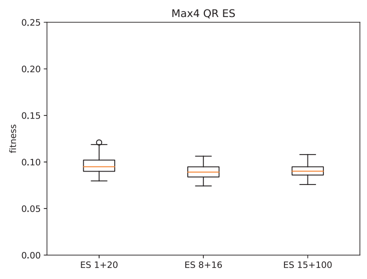
(d) ES on the M representation



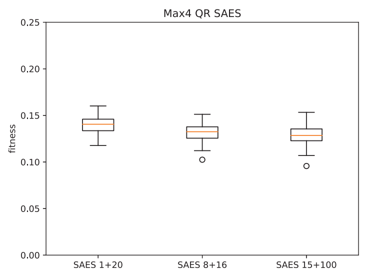
(e) SAES on the M representation



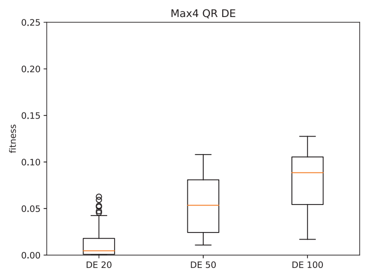
(f) DE on the M representation



(g) ES on the QR representation

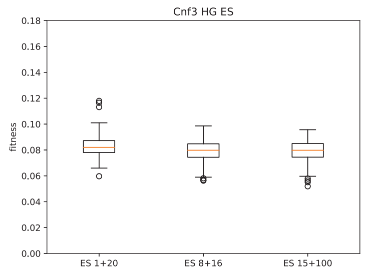


(h) SAES on the QR representation

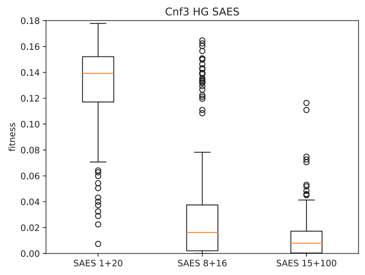


(i) DE on the QR representation

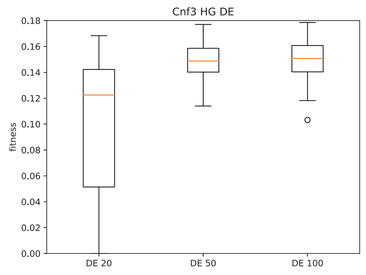
Figure 2: Statistical results from the evolution of the Max4 experiment.



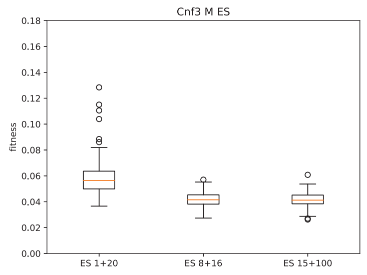
(a) ES on the HG representation



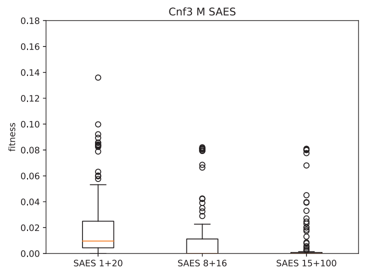
(b) SAES on the HG representation



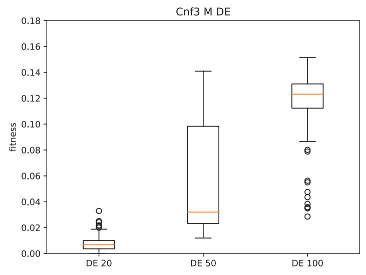
(c) DE on the HG representation



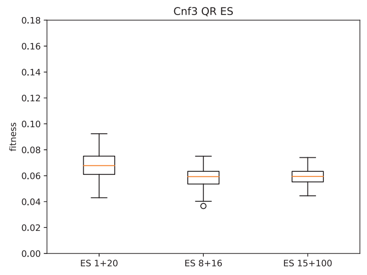
(d) ES on the M representation



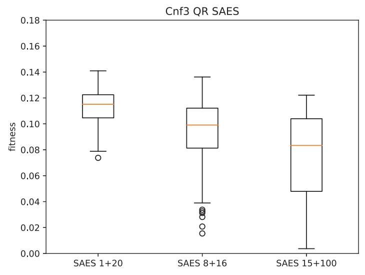
(e) SAES on the M representation



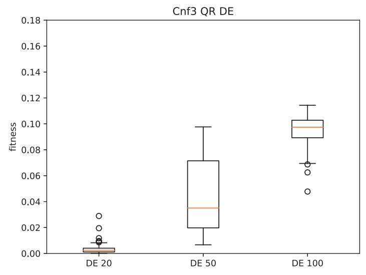
(f) DE on the M representation



(g) ES on the QR representation



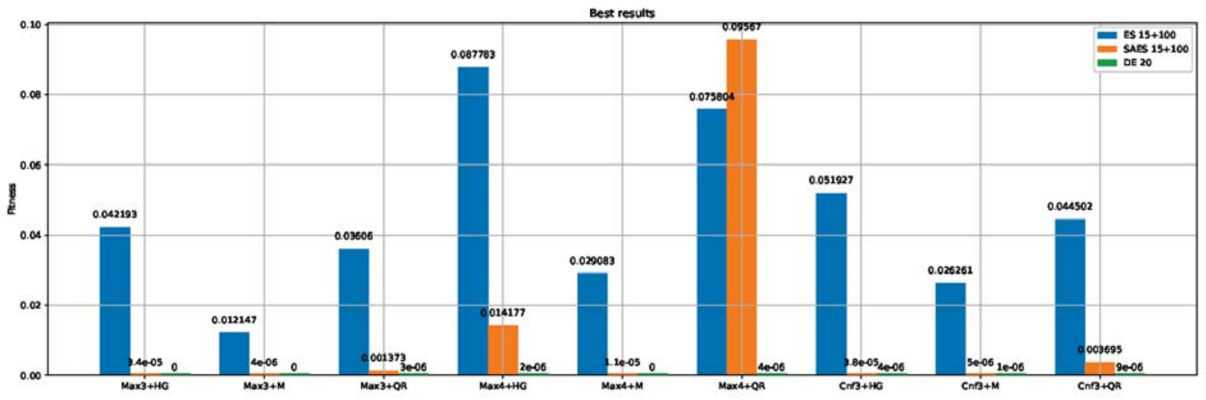
(h) SAES on the QR representation



(i) DE on the QR representation

Figure 3: Statistical results from the evolution of the Cnf3 experiment.

Figure 4: Comparison of best results from best performing setups of all three algorithms.


Best results
ES 15+100
SAES 15+100
DE 20
Fitness
3.4e-05
4e-06
3e-06
2e-06
1.1e-05
4e-06
3.8e-05
4e-06
5e-06
1e-06
9e-06
Max3+HG
Max3+M
Max3+QR
Max4+HG
Max4+M
Max4+QR
Cnf3+HG
Cnf3+M
Cnf3+QR