Computer Applications in Electrical Engineering
Vol. 14 *2016*
DOI 10.21008/j.1508-4248.2016.0026

Systemic Evolutionary Algorithm inspired by the methods of quantum computing to improve the accuracy of the model on the neuronal motion the end of the robot arm PR–02

Łukasz Wołynka, Jerzy Tchórzewski
Siedlce University of Natural Sciences and Humanities
08–110 Siedlce, 3–Maja 54, e–mail: Jerzy.Tchorzewski@uph.edu.pl, luccmase@gmail.com

The article contains selected results of research on the design Systemic Evolutionary Algorithm inspired by quantum informatics methods and description how to implement it in Matlab language in order to use for improve parameters neural model on example robot robot PR–02 arm motion. Initial population was based on weights matrix of artificial neural network. Randomly selected population of individual chromosomes in both the initial and in the following parent population have been converted to binary values, and these to quantum values by using created for this purpose *quatization()* function. Quantum gene value was determined on the basis of stonger pure state represented by different chromosomes, to which *dequantization()* function was used. Selection of individuals was conducted based on the model of neural robot PR–02 motion implemented in Matlab language using c*alculationsNeuralNetworks()* function.

KEYWORDS: Evolution algorithms, Quantum computing, Modeling systems, Robot PR–02, Artificial Neural Networks, Environment MATLAB and Simulink

1. Neuronal model motion PR–02 robot arm

The artificial neural network (ANN) is designed as a perceptron network, it is a multilayer one–way neural network with an input layer consisting of four neurons, which are given input values describing the parameters of degrees of freedom of the robot PR–02, one hidden layer and output layer consisting of three neurons, which the output values describing the spatial coordinates robot arm end positions of PR–02 are outputted. Learning file was build on the basis of the above four input values and three output values of PR–02 robot which are marked [21, 26]:

θ_1 – angle of the entire PR–02 robot platform (values in the range –300 ° – 0 °),
λ_2 – high of the platform suspension PR–02 robot arm (values between 0 – 0.200 m),
λ_3 – eject length PR–02 robot arm (values between 0.376 – 0.676 m),
θ_4 – the angle of rotation end of robot arm (values in the range 0 ° – 360 °), the angle of rotation end of the robot arm (values in the range 0 ° – 360 °),

so the element can make a full rotation around its own axis, but has no influence on the final position of the end of robot arm,

X,Y,Z – coordinates of the end PR–02 robot arm.

By changing the above. four parameters D–H may directly control the movement of end of PR–02 robot arm (coordinates X, Y, Z) by:

- raising the main arm to a height of 0,200 m,
- incomplete rotation about its own axis up to 300 degrees,
- eject arm to 0,300 m,
- full rotation around its own axis end of the arm.

Learning matrix is composed of 360 pairs of input and output vectors of the PPS, it is related to two rotations and two offsets (fig. 1) at the input side and the three spatial coordinates of the position of the end of robot arm on the output side. The learning Levenberg–Marquardt method with Bayesian regularization was used to learn ANN model of motion PR–02 robot arm [7, 11, 18, 23–24, 42–44]. Matrices value of the input U and output Y quantities, were subjected to cluster analysis, including normalizing by using MapMinMax algorithm built–in MATLAB and Simulink environment (normalization of input quantities values for each state of articulation between [–1 ÷ 1]), the designated parameters were represented by a mapminmax block in Simulink [2].

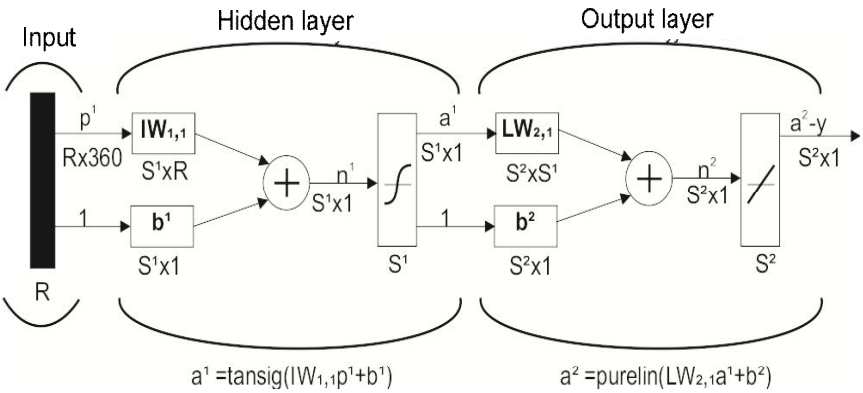

Fig. 1. ANN architecture used to learn neuronal motion the end of the robot arm PR–02. Other descriptions in text. Source: [42]

More important description to Fig. 1:

R – number of input values u_i, in this case R = 4,

R x 360 – matrix size which include input values u_i (in MATLAB environment – learning matrix),

P' – matrix of input values u_i (of dimension R x 360),

$\mathbf{IW}_{1,1}$, $\mathbf{LW}_{2,1}$ – weight matrices first layer (hidden) of dimension S^1 x R and second layer (output) of dimension S^2 x S^1,

$\mathbf{b}^1$,$\mathbf{b}^2$ – bias vectors first layer of dimension S^1 x 1 and second layer of dimension S^2 x 1,

a^1 = tansig($\mathbf{IW}_{1,1}p^1+b^1$) – hyperbolic function which was neurons of the first layer activation function:

$$f(net^1) = a^1(net^1) = \frac{2}{1+e^{-2n}} - 1, \tag{1}$$

a^2 = purelin($\mathbf{LW}_{2,1}a^1+b^2$) – linear function which was neurons of the second layer activation function,

n or net – signal value of first or second weight layer as result of the ANN as adder, this is the summation of the products of the input signals multiplied by the entrance,

y_j – j–th output value.

2. Quantum evolutionary algorithm to improve the model parameters

System Evolutionary Algorithm (SEA) which was inspired by quantum computing solutions was called quantum evolutionary algorithm (QEA) [25–26, 29–30, 33–34, 42]. The SEA algorithm was used to improve the parameters of the model neuronal motion PR–02 robot arm [27, 29, 31–32, 35–36]. The QEA algorithm was verified on example of the model neuronal motion PR–02 robot arm implemented in Matlab language [2, 26, 42] (Fig. 2).

SEA algorithm was developed based on control and systems theory [12, 37–37] and for the first time published in 2005 under the title of System Evolutionary Algorithm [35], and then developed in publications among others [29, 31–32, 36]. Important original solutions include creating IP–based control and systems theory. Strict approach of the adaptation function as a robustness function reduced to differences between the individual and the average value in the population.

2.1. Quantum Initial Population

The initial population (IP) QEA was built on the basis of the elements of both weights matrices at set intervals of ($\mathbf{IW}_{1,1}$ and $\mathbf{LW}_{2,1}$) using the quantization operation defined in [25–26]. The process of determining the IP has been described in detail in [42], and the dequantization operation was defined in the block called Evaluation of adaptation (Fig. 2) was used to convert the received quantum states to binary values, and those on the decimal values. For each chromosome consisting of the above. both the matrix weights determined Euclidean distance measured by the absolute value of the difference

value of each chromosome and the average value of the population in a given generation algorithm QEA, which determines a peculiar robustness of individual chromosomes in the course of step QEA algorithm [4, 10, 13, 16, 28, 40, 42]. Adaptation of chromosomes to continue to participate in QEA was based on robustness function of each chromosome. Decision block concerning the end of QEA algorithm criterium was built on basis of the set value of divergence, it is permissible deviations of the best chromosome from the average value of all the chromosomes of a particular era.

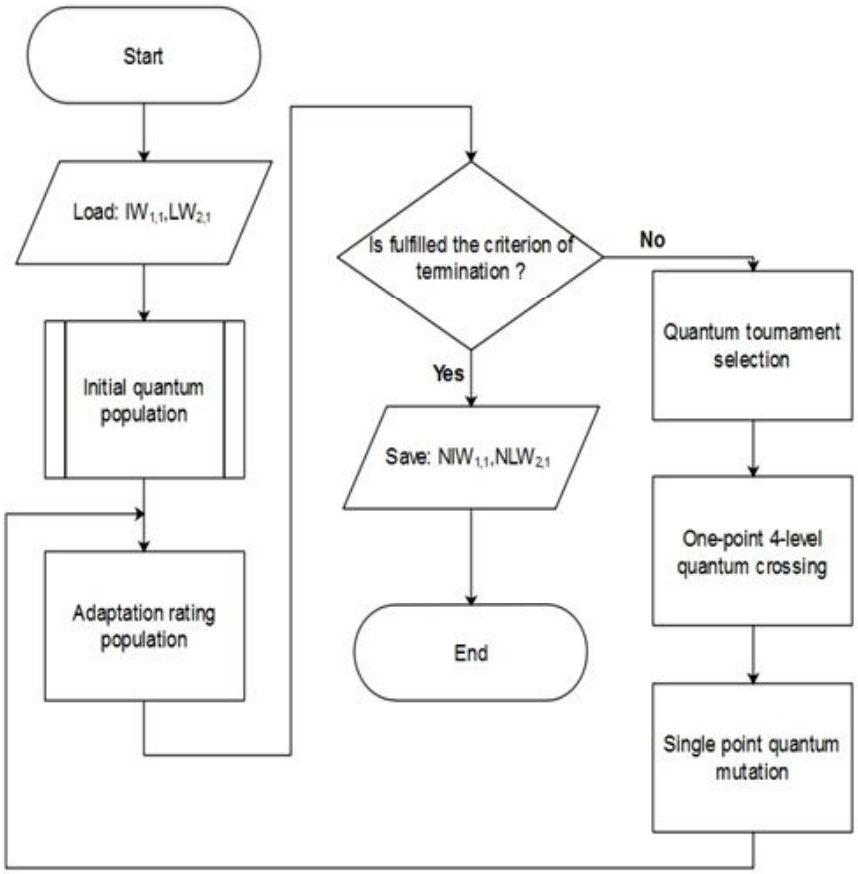

Fig. 2. Quantum Evolutionary Algorithm to improve the parameters model neuronal motion the end of the PR–02 robot arm. Descriptions in text. Source: [42]

The elimination of individuals used in quantum tournament selection with repetition, assuming the operator of single–point crossover and mutation quantum relating to changes of the value selected at random gene to opposite value (e.g. from the state $|0\rangle$ to $|1\rangle$, from the state $l_{k0} = 0{,}6|0\rangle+0{,}8|1\rangle$ to

l_{k1} = 0,8|0>+0,6|1>). In this way, individual blocks QEA performed actions on quantum states. As a result of QEA received more accurate weight values of both layers, it is the matrix: $\mathbf{IW}_{1,1}$ and $\mathbf{LW}_{2,1}$. The quantization operator created possibility of transition from decimal to binary system and further from the binary system on the quantum system. For each chromosome randomly selected an infinite number of its states (e.g. expressed by the number of 1 000 states), which allows to determine the number of occurrences of pure states |0> and the pure states |1> for each qubits, which were used to generate mixed states. It was dictated by the need to obtain classic phenomenon of an infinite number of states for each qubits simultaneously in classic computer.

The above mentioned algorithm is included in block diagram form on Fig. 3 (a block for generating IP). The draw was used intervals with an accuracy higher by one order of value. Then, as defined ranges of weights randomly selected 1 000 of the individual weights, which was converted to binary form (Fig. 3 – matrix ***Mlb***). Weight values (each weight in the number of 1 000) were generated from above. intervals and converted to binary form by using m–file script developed in Matlab as RandQuantumPopulationGPU [2, 42].

For each such set of bits associated with the designated weight value is the probability of the binary state. The quantization operation consists to the determination of the quantum state where the probability of occurrence of one of the binary states corresponding to the amplitude of the base state created single qubit quantum state, leading to a mixed states [25, 36].

Table 1. Weight values generated a number of 1 000 and converted to binary form. Source: [42]

Bits index	63	62	...	52	51	50	...	2	1	0
Examples of binary values	0	0	...	1	0	1	...	0	0	0
Parts of binary value	Sign bit	Exponent			Mantissa					

2.2. The QEA use to improve the parameters model neuronal motion the end of the PR–02 robot arm

Generated initial population of 100 chromosomes from ranges defined by basis on both matrices weight values in determining by one higher order of magnitude accuracy of individual weight values. On the classic computer conducting such parallel computing is associated the high costs of the algorithm, for the creation of quantum PP, and also by the population Parental (PR) lasted approx. 30 min. Part of the code has been parallelized by using solutions available in Parallel Computing Toolbox (PCT) in order to speed up the computation. Processor performing calculations was the Intel Core i7 3610 QM, 2,3 MHz having 4 cores, which run in parallel were separate and

independent processes of computation, performing parallel above script. The processes working independently of the main process supported by the MATLAB compiler [2].

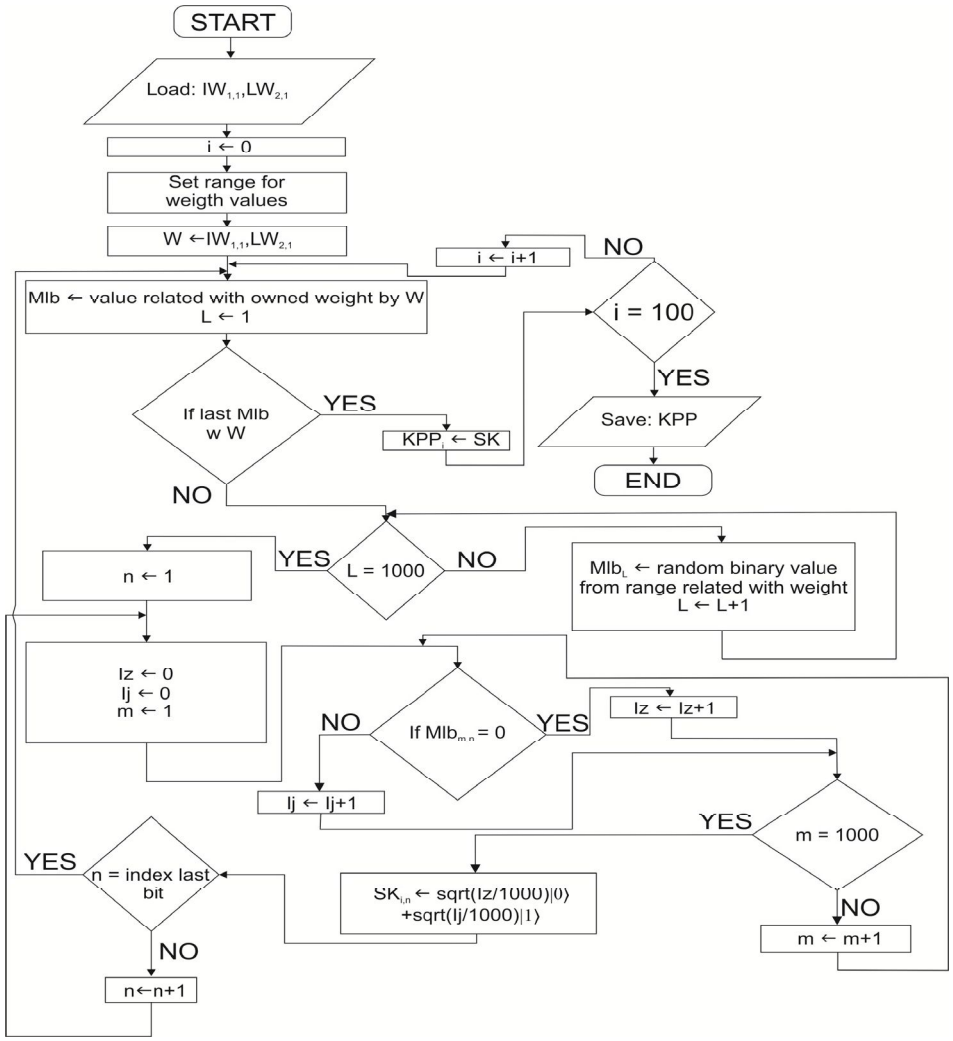

Fig. 3. Block diagram of the generation of the IP in QEA used to improve the parameters model neuronal motion the end of the PR–02 robot arm. Description: SK – quantum state, lz – number of zero, lj – number of one, KPP – Initial quantum population. Descriptions in text. Source: [42]

Quantization operator was built similar to the dequantization operator, which gives the ability to transition from the quantum form to binary form, and then the binary form to decimal form. At the beginning of each individual are composed of a single chromosome, where each gene is represented by a single

qubit, was set the base state, as a state with a higher probability of occurrence of pure |0> or |1> in the set of 1000 states, within the range of the border between the state |0> and state |1> (acceptance in principle of superposition equal amplitudes $\alpha = \beta$). Then, applying the principle of superposition of states for each of the qubits separately set the second state using the principle of superposition of states [1, 3, 5–6, 8–9, 15, 17, 19–20, 22]. Can also make a random selection of values in the range <0,1> for a stronger pure state, which increases the chance state to obtain which, the probability value of the amplitude is greater than the base state. On the other hand, it is not precluded randomly selected a smaller value of the area defined by the probability of the amplitude of the base state. In such a method may also be designated pure state |0 > or |1>.

Then the amplitude associated with clear conditions are compared with one another condition to extract more probable as a mixed state of the gene in the chromosome. Then the amplitude associated with clear conditions are compared with one another condition to extract more probable as a mixed state of the gene in the chromosome. To simplify the description of mixed states and pure associated with the chromosome it was adopted as subchromosomes. In this case it is possible mapping pure states to bits, respectively: 0 to |0> and 1 to |1> creating one weight value. Then, decoding of 64 bits being one of the chromosome to form the real numbers belonging to using IEEE 754. The structure of chromosome is shown with the first and last chromosome ch_1 and last chromosome ch_{100}, which are provided in Table 2.

Table 2. The structure of a chromosome with the sample weight values generated in a number of 1 000 (chromosome no. 1 and chromosome no. 100). Source: [42]

Indexes	Number of gene		0	1	…	63
chromosome: ch_1	subchromosome pch^1_1	α|0>	0.6	0.4	…	0.9949
		β|1>	0.8	0.9165	…	0.1
		Pure state	|1>	|1>	…	|1>
chromosome: ch_{100}	subchromosome pch^1_{100}	α|0>	0.77659	0.3	…	0.9165
		β|1>	0.63	0.9539	…	0.4
		Pure state	|1>	|0>	…	|0>

The chromosome ch_1 consists of one subchromosome pch^1_1 obtained in the quantizing process. Each of subchromosomes consists of a mixed state and obtained in the pure state dequantization. The chromosome ch_{100} is made up of subchromosome pch^1_{100}. Subchromosomes of chromosome ch_{100} have the same structure as in the case of chromosome ch_1. After the dequantization, the evaluation of adaptation 100 chromosomes representing neuronal model in the form of a two weight matrices, was made. Function which was built as the

vector of the output artificial neural network was used to evaluate the adaptation [7, 13–14, 16, 29, 35]. Values which were generated by neural model in the form of ANN are trajectory points of the end of the robot PR–02 arm. To determine their adaptation, more precisely determine the error in relation to the real trajectory, have to calculate the Euclidean distance of each point received from a simple kinematics for PR–02 robot [21, 25–26, 42] in adaptation process conducted by QEA [25–26, 30–31, 33–34, 42]. Selection method allows to greater exploration of the data through the selection of the *n*– tournaments, random selected *k*–element group, of which best suited one chromosome is showdown which will take part in the crossover [7, 13–14, 16, 43]. In the case of the experiment 100 tournaments were extracted with a group of five chromosomes. 100 chromosomes for crossover operations were obtained by using a selection method.

2.3. Quantum operator of crossover and mutation

The four level single point crossover operator was chosen because of the complex structure of a single chromosome which represent the two matrices ANN (Table 2). With two parents selected randomly, the intersection for each parent called locus chosen in a random manner for each intersection, whose value can not be higher than the length of a chromosome [42], in other words, must not exceed 64 and should belong to the set of natural numbers *N*. In this case, the value is obtained from the interval <1÷64–1>.

Given the structure chromosome must take into account the amplitude of quantum states subchromosomes which will be replaced during the crossover. This will affect the proper operation of the algorithm, since the amplitude of quantum states will result in a greater randomness of values chromosome, causing not only finding the best solutions locally. The four level single point crossover within the better illustrate the essence of the action shown in Fig. 4–6. As has been interpreted in Figure 4 first offspring is created of first parent chromosomes that mean from the first chromosome to the crossover points (locus) m_k and second parent chromosomes of the crossover points m_k the end of *j*. Similarly, the second offspring will be made, namely the chromosome of the second parent will be created from 1,1 to first crossover points and the first parent chromosomes to the end of the crossover points *j*. The place of crossover is selectable separately for each pair of parents. Subchromosomes in Fig. 5 was marked with darker color as the amplitude for the base state |0> and brighter as the amplitude for the base state |1>, which in dequantization process becomes a single gene in chromosome. In one chromosome include 64 subchromosomes marked on Fig.5 from 1 to 64. The dimensions of the individual are presented as *i x j*.

The second level crossover is applied to each row of the weights matrix, which is part of a offspring hereafter referred to as a parent. For each row will be carried out a separate crossover. As shown in Fig. 6 for each row *i–th* index of the parent matrix is randomized intersection point $m_{i,k}$. The second level crossover operation involves the creation first offspring in *i–th* row from the first parent chromosomes from 1,1 to $m_{i,k}$ that mean *j–th* column and second parent from chromosome $m_{i,k}$ for *i–th* row to the end *j–column*.

In the same way is created second offspring of second parent chromosome from the first (1,1) chromosome to $m_{i,k}$ and first parent from chromosome $m_{i,k}$ to the last chromosome of the analyzed *i–th* index and *j–th* column.

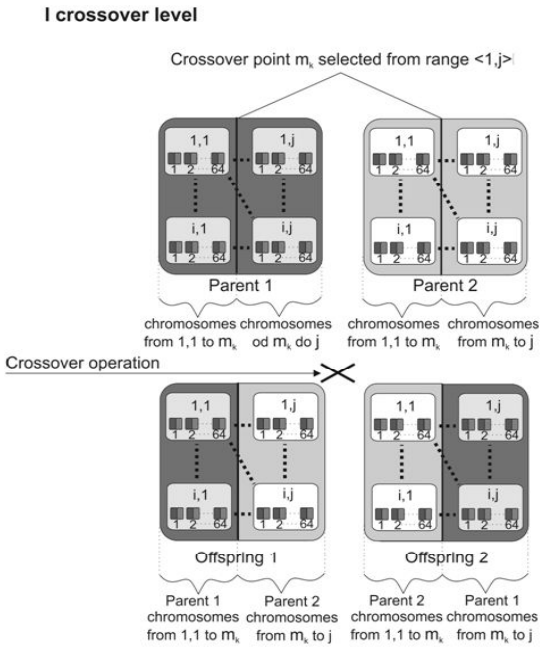

Fig. 4. One–point crossover scheme of the first level. Description in text. Source: [42

The third level crossover is related to the exchange of chromosomes in the *i–th* row accordance with the point of intersection obtained at random from the range from 1 to the last *i–th* row created parent. The process is repeated for each column. The method of crossover is the same as the second level crossover.

The fourth level crossover is associated with crossover into parent subchromosomes creating chromosome. The crossover subchromosomes first and second parents single chromosome in the Fig. 6 is showed.

A first offspring was created from a first parent subchromosomes from 1 to crossover points m_k and second parent subchromosomes from m_k 2 to the last 64 subchromosome.

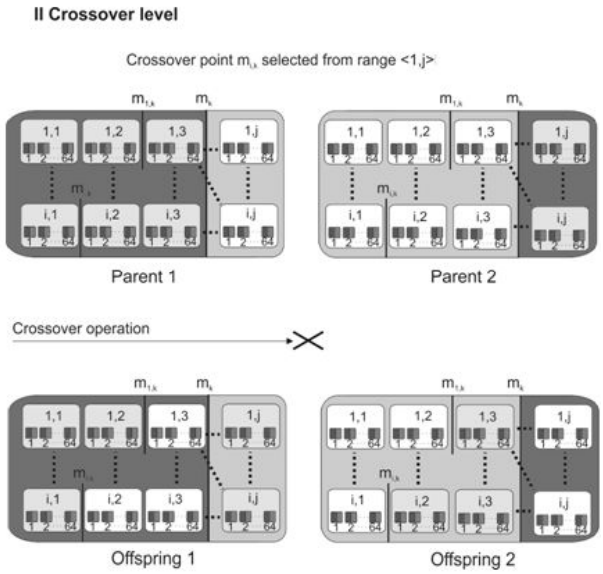

Fig. 5. One–point crossover scheme of the second level. Description in text. Source: [42]

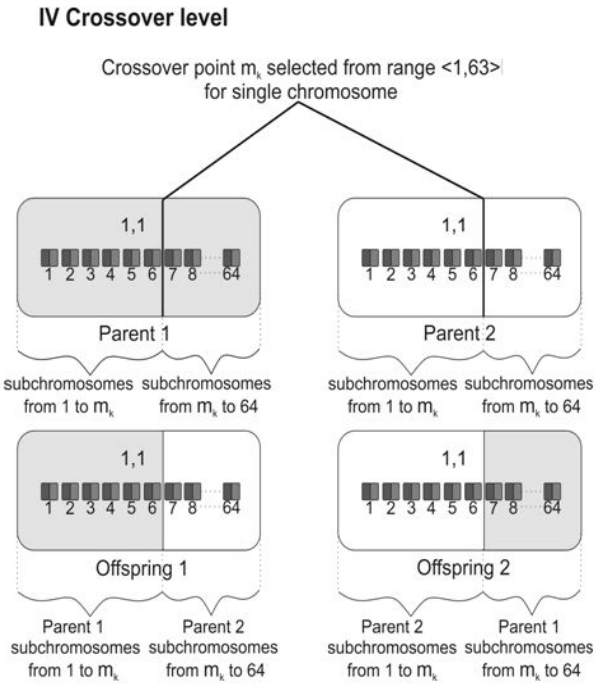

Fig. 6. One–point crossover scheme of the fourth level: Description in text. Source: [42]

As mentioned above the second offspring was created from a second parent subchromosomes, namely from subchromo–somes from 1 to m_k and first parent from crossover point to 64.

For each pair of parents assigned one crossover. Reference to the construction of the individual for the experiment, crossover will be one for the two sets of weights (parents). Delving into the structure of the parents, subchromosomes chromosomes, according to a designated crossover will create new offsprings. The mutation is also one point, namely amplitude as part of single qubit quantum states creating the structure subchromosome are exchanged between the base states |0> and |1> [7, 13–14, 16]. Bit creating value of chromosome gene that is obtained during the process of dequantization, it will also be subjected to mutations.

For example, if the bit is set to 1 will replace the 0. The probability of mutation is identical for each subchromosome. For each gene, creating chromosome is determined from the interval value <0.1>, then is compared with the probability of mutation, and checked it is relative or less equal. If the criterion is fulfilled then carried a mutation in this gene chromosome. The probability of mutations can take a value from the same <0,1> range. The action method of mutation operator in the case of this example is reported in Table 3.

For this example (Table 3) presents a single point mutation process in the following steps:

Stage I

- adoption of values for ratio 0.5,
- randomly selected from each chromosome value a random number between <0,1>,
- checking the random value corresponding to the chromosome value if is less or equal to the number of mutations ratio,
- if the random value corresponding gene fulfills the condition then check if there are more genes then meet the criterion:
  - when there are more – all operations are subject to mutation,. In the example, two genes 1 and 63 meet the conditions, the operation will be carried,
  - if there is one – only he will be subjected to a process of mutation.

Stage II

Gen meets the criterion will be mutated in several:

- Gene bit – when the value 0 is converted into 1,
- Subchromosome – value of base states amplitudes are converted within a state.

Single–point crossover operator and the process of crossover was carried out in the form of a cross with four–level immersion (the highest level concerned the entire neuronal model, second level – the matrix weights, level three – rows in a weights matrix and the lowest level –element stored in the form of quantum numbers) [1, 9, 20].

Table 3. Example of a two–step single–point mutation performed on a single chromosome. Description in text. Source: [42]

Stage I							
Chromosome ch_1	**No. gene**		**0**	**1**		**63**	
	Gene value:		0	1		1	
	Subchromosome pch_1^1	α|0>	0.6	0.4		0.9949	
		β|1>	0.8	0.9165		0.1	
		Pure state	|1>	|1>		|1>	
Random selected number of range <0,1>			0.4	0.5	...	0.45	
Mutation probability: 0.5							
Stage II a							
Chromosome: ch_1	**No. gene:**		**0**	**1**	**...**	**63**	
	Gene value:		0	1	0	1	
	Sub chromosome pch_1^1	α|0>	0.6	0.4	...	0.9949	
		β|1>	0.8	0.9165	...	0.1	
		Pure state	|1>	|1>	...	|1>	
Random selected number of range <0,1>			0.4	0.5	...	0.45	
Mutation probability: 0.5							
Stage II b							
Chromosome ch_1	**No. gene:**		**0**	**1**	**...**	**63**	
	Gene value:		0	0	0	0	
	Sub chromosome pch_1^1	α|0>	0.6	0.9165	...	0.9949	
		β|1>	0.8	0.4	...	0.1	
		Pure state	|1>	—	...	—	
Random selected number of range <0,1>>			0.4	0.5	...	0.45	
Mutation probability: 0.4							

Mutation operator was also used as the operator of single–point, which was consists to change one pure state to another corresponding and conversion of mixed state on a mixed state with him engaged.

3. Analysis of the possibility of QEA algorithm implementation

3.1. Research the QEA sensitivity

Susceptibility testing of the algorithm QEA confirmed its high robustness (effectiveness and efficiency) in improving the accuracy value scales, which resulted in improving the quality of the model neuronal motion the end of the PR–02 robot arm.

This algorithm was sensitive to the change in the probability of mutations, as summarized in Table 4. The best results were obtained using the mutation probabilities 0.001, and the accuracy of $5{,}0000055*10^{-10}$, wherein the chromosomes average arithmetic errors are best suited for 100 generations was 0.09.

Table 4. The results of the QEA susceptibility on the change in the probability of mutation. Source: [42]

Number of turns QEA algorithm for 100 epochs	The probability of mutation	The arithmetic average of the error result for the best individuals (elite
50	0,5	2.34
	0,1	1.66
	0.01	0.23
	0.001	0.09
	0.0001	0.27
	0.00001	0.65
	0.000001	0.76
	0.0000001	0.92

QEA algorithm gave better results than for the same parameters EA classic. Value of error for the elite QEA was $0{,}17575*10^{-11}$ (Fig. 7) a EA classical was $0{,}1757286*10^{-8}$ (increase accuracy by three orders).

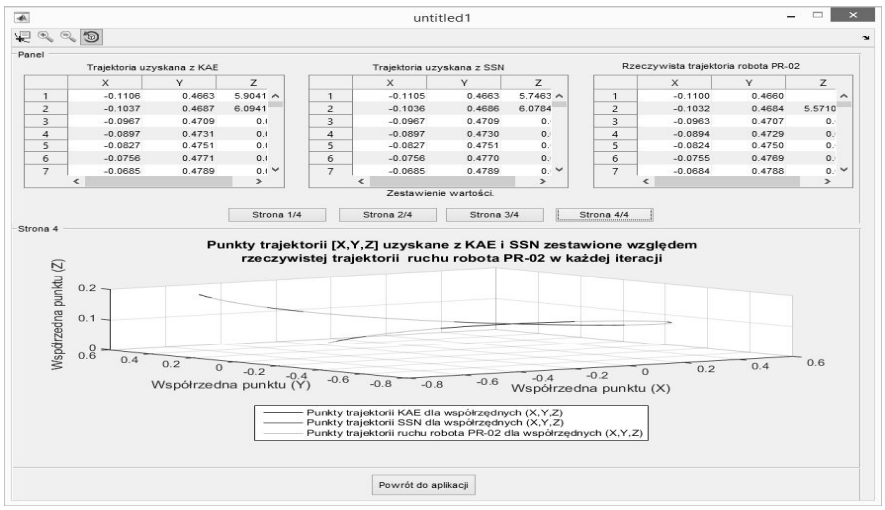

Fig. 7. The results of QEA. Description: Punkty trajektorii KAE dla współrzędnych – QEA points trajectory for the coordinates, Punkty trajektorii SSN dla współrzędnych – ANN points trajectory for the coordinates, Punkty trajektorii ruchu robota PR–02 dla współrzędnych – Points trajectory robot PR–02 for the coordinates. Description in text. Source: [42]

4. Conclusions

This article is a continuation of the work of the same title and the main subtitled Part 1. Quantum evolutionary algorithm [8].

The results obtained for QEA implemented in Matlab language confirmed the possibility of using quantum computing to build a practical model of quantum evolutionary algorithm and use it to improve traffic parameters the end of the PR–02 robot arm.

Susceptibility testing showed that a quantum evolutionary algorithm improved the accuracy of the model neuronal motion of the PR–02 robot arm by three orders accuracy.

References

[1] Bellac L. M., Wstęp do informatyki kwantowej (in Polish), WN PWN, Warszawa 2011.
[2] Embedded MATLAB™ User's Guide © COPYRIGHT 2007 by The MathWorks, Inc. Natick, MA 01760–2098 (USA).
[3] Giaro K., Kamiński M., Wprowadzenie do algorytmów kwantowych (in Polish), AOW EXIT, Warszawa 2003.
[4] Giergiel J., Hendzel Z., Krzepkie algorytmy sterowania ruchem nadążnym mobilnych robotów kołowych (in Polish), Materiały: Warsztaty mobilnych robotów, IPPT PAN, Warszawa, 1997, 65–72.
[5] Greenberg Y.S., Kochmanski M., Podstawy informatyki kwantowej. Elementarne wprowadzenie (in Polish). Wyd. URz., Rzeszów 2011.
[6] Han K., Kim J., Genetic quantum algorithm and its application to combinatorial optimization problem, Evolutionary Computation, 2000. Proceedings of the 2000 Congress on, 2000, Vol. 2, pp. 1354–1360.
[7] Helt P., Parol M., Piotrowski P., Metody sztucznej inteligencji w elektro-energetyce (in Polish). OW PW. Warszawa 2000.
[8] Heller M., Elementy mechaniki kwantowej dla filozofów (in Polish), OBI, Wyd. DT Biblios, Kraków 2011.
[9] Hervensalo M., Algorytmy kwantowe (in Polish), WSiP, Warszawa 2004.
[10] Hendzel Z., Żylski W.: Krzepki obserwator uogólnionych prędkości mobilnego robota kołowego (in Polish), Materiały VI Krajowej Konferencji Robotyki, Wrocław 1998.
[11] Korbicz J., Sztuczne sieci neuronowe i ich zastosowanie w elektrotechnice i energetyce (in Polish). Przegląd Elektrotechniczny nr 9/2009, s. 194–200.
[12] Kaczorek T., Dzieliński A., Dąbrowski W., Łopatka R., Podstawy teorii sterowania (in Polish). WNT, Warszawa 2007.
[13] Kwaśnicka H., Obliczenia ewolucyjne w sztucznej inteligencji(in Polish). PWr., Wrocław 1999.
[14] Kłopotek M. A., Techniczne zastosowanie strategii ewolucyjnych (in Polish). IPI PAN, Warszawa 1999.

[15] Liao G., Using chaotic quantum genetic algorithm solving environmental economic dispatch of Smart Microgrid containing distributed generation system problems, Power System Technology, POWERCON, 2010 International Conference on, 2010, pp.: 1 – 7.
[16] Michalewicz Z., Algorytmy genetyczne + struktury danych = programy ewolucyjne (in Polish). WNT, Warszawa 1999.
[17] Narayanan A., Moore M., Quantum–inspired genetic algorithms, Proceedings of IEEE International Conference on Evolutionary Computation, pp. 61–66, IEEE, 1996.
[18] Osowski S., Sieci neuronowe do przetwarzania informacji (in Polish). PW, Warszawa 2000.
[19] Pittenger A. O., An Introduction to Quantum Computing Algorithms, Birkhauser, Boston 2000.
[20] Sawerwain M., Wisniewska J., Informatyka kwantowa. Wybrane obwody i algorytmy (in Polish). WN PWN SA, Warszawa 2015.
[21] Szkodny T.: Kinematyka robotów przemysłowych (in Polish). Wyd. PŚ., Gliwice 2013.
[22] Susskind L., Friedeman A., Mechanika kwantowa. Teoretyczne minimum (in Polish). Prószyński i S–ka, Warszawa 2016.
[23] Tadeusiewicz R., Sztuczne sieci neuronowe (in Polish). AOWRW, Warszawa 1993.
[24] Tadeusiewicz R., Szaleniec M., Leksykon sieci neuronowych (in Polish). Wyd. Fundacji „Projekt Nauka", Wrocław 2015.
[25] Tchórzewski J., Możliwości informatyki kwantowej do poprawy dokładności modelowania. Część 1. Kwantowy algorytm ewolucyjny (in Polish). Poznan University of Technology. Academic Journal. Wyd. PP, Poznań 2016.
[26] Tchórzewski J., Wołynka Ł., Możliwości informatyki kwantowej do poprawy dokładności modelowania. Część 2. Kwantowy algorytm ewolucyjny ruchu robota PR–02 (in Polish). Poznan University of Technology. Academic Journal. Wyd. PP, Poznań 2016.
[27] Tchórzewski J., Chyży E., Researching the Development of the Electrical Power System Using Systemically Evolutionary Algorithm. International Journal of Soft Computing and Software Engineering [JSCSE], Vol. 6, No. 1, pp. 1–10, 2016.
[28] Tchórzewski J., Rozwój systemu elektroenergetycznego w ujęciu teorii sterowania i systemów (in Polish). OW PWr. Wrocław 2013.
[29] Tchórzewski J., Systemic Evolution Algorithms for the design of the development of the electricity market, 10th International Conference on the European Energy Market, EEM 2013, IEEE Xplore Digital Library, 2013, pp. 1–8.
[30] Tchórzewski J., Jurkowski B., Stan i możliwości implementacyjne informatyki kwantowej na przykładzie kwantowego algorytmu ewolucyjnego (in Polish). [w:] Postępy w Elektrotechnice Stosowanej. Materiały VII Ogólnopolskiej Konferencji Naukowo–Technicznej PES–7. OW PTETiS. Wydz. Elektryczny PW, Ośrodek Promocji Badań z/z Energoelektroniki PW. PS Sekcja IEEE. Warszawa–Kościelisko 2009.

[31] Tchórzewski J., Wąsowski A., Systemowy algorytm ewolucyjny SAE w generowaniu rozwoju rynku energii elektrycznej (in Polish). Materiały V Międzynarodowej Konferencji Naukowo–Technicznej ENERGETYKA. PWr., Wrocław 2008.
[32] Tchórzewski J., Systemic Method of Structure and Parameters Researching for Beginning Population Building of Evolving Algorithm SAE on the Example of Electricity Market, Polish Journal of Environmental Studies, Vol. 17, No 2A, 2008, pp. 86–89.
[33] Tchórzewski J., Owczarczyk P., Model and Implementation of Quantum Neural Network for Transfer Knowledge about Electricity Market, Studia Informatica, Vol 3–4 (7) 2007, AP, Siedlce 2007.
[34] Tchórzewski J., Owczarczyk P., Model i implementacja kwantowej sieci neuronowej do przetwarzania wiedzy o rynku energii elektrycznej (in Polish). Computer Application in Electrical Engineering, Politechnika Poznańska, Poznań 2007.
[35] Tchórzewski J., Systemowy Algorytm Ewolucyjny SAE. Bio–Algorithms and Med–Systems (in Polish). Journal Edited by Collegium Medicum. Vol. 1. No. 1/2/2005. UJ. Kraków 2005, pp. 61–64.
[36] Tchórzewski R. J, Ruciński A., Model i implementacja systemowego algorytmu ewolucyjnego do poszukiwania nowego stanu systemu elektroenergetycznej sieci przesyłowej. Studia Informatica, Systemy i technologie informacyjne. UPH, Siedlce 2005, nr 1 (5), s. 59–67.
[37] Tchórzewski J., Systemowe badanie prawidłowości rozwoju systemu sterownia na przykładzie rozwoju elektroenergetycznej sieci przesyłowej (in Polish). Rozprawa naukowa nr. 58. AP. Siedlce 1997.
[38] Tchórzewski J., Cybernetyka życia i rozwoju systemów (in Polish). Monografie nr. 22. Wyd. WSRP w Siedlcach. Siedlce 1992.
[39] Tchórzewski J., Inżynieria rozwoju systemów (in Polish). Monografie nr 18. Wyd. WSRP w Siedlcach. Siedlce 1990.
[40] Węgrzyn S, Klamka J, i inni, Nano i kwantowe systemy informatyki (in Polish). Wyd. PŚ, Gliwice 2004.
[41] Wierzchoń S., Kłopotek M., Cluster analysis. Monografie. Wyd. IPI PAN. Warszawa 2015.
[42] Wołynka Ł., Model i implementacja kwantowego algorytmu ewolucyjnego na przykładzie ruchu robota PR–02 (in Polish). Praca magisterska napisana pod kierunkiem prof. nzw. dr hab. inż. Jerzego Tchórzewskiego w Zakładzie Modelowania i Projektowania Systemów Informatycznych Instytutu Informatyki na Wydziale Nauk Ścisłych, UPH, Siedlce 2015.
[43] Zieliński J. [red.], Inteligentne systemy w zarządzaniu. Teoria i praktyka (in Polish). WN PWN. Warszawa 2000.
[44] Żurada J., Barski M., Jędruch W., Sztuczne sieci neuronowe. Podstawy teorii i zastosowania(in Polish). PWN, Warszawa 1996.

(Received: 12. 10. 2016, revised: 21. 11. 2016)